Firewalls in Hochgeschwindigkeitsnetzen

Untersuchungen zu Firewalls in ATM-Netzen

Dissertation

zur Erlangung des Doktorgrades
am Fachbereich Informatik
der Universität Hamburg

vorgelegt von

Uwe Ellermann

Hamburg 1999

Genehmigt vom Fachbereich Informatik der Universität Hamburg auf Antrag von

Prof. Dr. Klaus Brunnstein, Universität Hamburg
Dr. Hans-Joachim Mück, Universität Hamburg

Hamburg, den 12. Juli 1999
(Datum der Disputation)

Prof. Dr. L. Dreschler-Fischer
(Dekanin des Fachbereichs Informatik)

Copyright © 1998–1999 Uwe Ellermann, Hamburg

Alle Rechte vorbehalten. Die vorliegende Publikation ist urheberrechtlich geschützt. Kein Teil des Werkes darf in irgendeiner Form ohne schriftliche Genehmigung reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder weiterverarbeitet bzw. weiterverbreitet werden.

Bezeichnungen von in dieser Arbeit genannten Erzeugnissen, die zugleich eingetragene Warenzeichen sind, wurden nicht besonders kenntlich gemacht. Es kann aus dem Fehlen der Markierung `tm` nicht geschlossen werden, daß die Bezeichnung ein freier Warenname ist. Ebensowenig ist zu entnehmen, ob ein Patent oder ein Gebrauchsmusterschutz vorliegt.

Der Verfasser hat Texte und Abbildungen mit größter Sorgfalt erarbeitet. Dennoch können Fehler nicht ausgeschlossen werden. Es wird weder eine juristische Verantwortung noch eine Garantie für die Informationen und Abbildungen in bezug auf Qualität, Durchführbarkeit oder Verwendbarkeit für einen bestimmten Zweck übernommen. In keinem Fall ist der Verfasser für direkte, indirekte oder gefolgte Schäden haftbar, die aus der Anwendung der Arbeit resultieren.

Inhaltsverzeichnis

Inhaltsverzeichnis III
Vorwort VII

1 Einleitung und Motivation **1**
1.1 Firewalls in neuen Umgebungen 1
1.2 Überblick über die folgenden Kapitel 3

2 Überblick über den Problembereich **7**
2.1 Überblick über netzbasierte Angriffe 8
2.1.1 Konzeptionelle Schwächen 9
2.1.2 Implementationsfehler 19
2.1.3 Konfigurationsfehler 21
2.2 Absicherung durch kryptographische Verfahren 21
2.3 Firewalls 27
2.3.1 Firewall-Komponenten 29
2.3.2 Firewall-Architekturen 36
2.3.3 Nicht durch Firewalls abgedeckte Sicherheitsprobleme . . . 44
2.4 Überblick über die Techniken von ATM-Netzen 45
2.4.1 Kommunikation in ATM-Netzen 48
2.4.2 “Classical IP” (CLIP) 55
2.4.3 “Next Hop Resolution Protocol” (NHRP) 56
2.4.4 “Multicast Address Resolution Server” (MARS) 59
2.4.5 “Label-Switching” 59
2.4.6 “LAN Emulation” (LANE) 62
2.4.7 “Multi-Protocol over ATM” (MPOA) 63
2.4.8 Unterschiede zwischen ATM und herkömmlichen Netzen . 64
2.5 Neue Herausforderungen für Firewalls 68

3 Firewalls in ATM-Netzen **71**
3.1 Firewall-Aspekte von ATM-Netzen 73

3.2 Neue Schwachstellen in ATM-basierten Netzen 80
3.2.1 Schwachstellen von ATM-Netzen 80
3.2.2 Schwachstellen von CLIP-Netzen 88
3.2.3 Schwachstellen bei NHRP 94
3.2.4 Schwachstellen bei MARS 97
3.2.5 Schwachstellen von "LAN Emulation" 98
3.2.6 Schwachstellen bei MPOA 99
3.3 ATM-spezifische Firewall-Komponenten 99
3.3.1 Kontrolle der Signalisierung 100
3.3.2 Packet Screens für ATM-Netze 142
3.3.3 Kontrollkomponenten auf Anwendungsebene 156
3.3.4 Kontrolle des Netzwerkmanagements 178
3.4 Firewall-Architekturen für ATM-Netze 179
3.4.1 Firewall-Architekturen für CLIP-Netze 180
3.4.2 Einsatz virtueller Netze für Firewall-Architekturen 182
3.4.3 Kombination von Bastion und Signalisierungskontroller . . 187
3.4.4 Asymmetrische Kontrolle von Verbindungen 190
3.4.5 Zentrale Kontrolle und virtuelle Netze 192
3.4.6 Einsatz virtueller Schnittstellen 200
3.4.7 "Native"-ATM-Firewalls 202
3.5 Zukünftige Firewall-Lösungen für ATM-Netze 204

4 Performanz von Firewalls 209
4.1 Performanzkriterien für Firewalls 212
4.2 Das Testnetz 215
4.2.1 Meßwerkzeug "Netperf" 217
4.2.2 Datendurchsatz 218
4.2.3 Nachrichtenlaufzeit 247
4.2.4 Verbindungsaufbauzeit 251
4.2.5 HTTP-Meßwerkzeug 258
4.2.6 Maximale Anzahl paralleler Verbindungen 263
4.2.7 Transaktionsdurchsatz bei parallelen Transaktionen 269
4.2.8 Transaktionsmessung mit längeren Transaktionen 277
4.2.9 Tuning von TCP-Parametern 281
4.3 Performanz von Packet Screens 287
4.3.1 Datendurchsatz über eine Packet Screen 290
4.3.2 Datagrammdurchsatz über eine Packet Screen 294
4.3.3 Messungen ohne Reduzierung der MTU 301

4.3.4 Nachrichtenlaufzeit über eine Packet Screen 303
4.3.5 Verbindungsaufbauzeit über eine Packet Screen 305
4.3.6 Abschließende Bewertung der Performanz von Packet Screens 307
4.4 Performanz von Proxy-Servern 312
4.4.1 Messungen mit "Socks" 314
4.4.2 Messungen mit "MGproxy" 321
4.4.3 Abschließende Bewertung der Performanz von Proxy-Servern 356
4.5 Performanz von Kombinationen aus Packet Screen und Bastion . 362
4.5.1 Datendurchsatz über Packet Screen und Bastion 363
4.5.2 Nachrichtenlaufzeit über Packet Screen und Bastion 365
4.5.3 Verbindungsaufbauzeit über Packet Screen und Bastion . . 366
4.5.4 Abschließende Bewertung der Performanz von Kombinationen aus Packet Screen und Bastion 367
4.6 Abschließende Bewertung der Firewall-Performanz 368

5 Konzepte paralleler Firewalls und weitere Optimierungsansätze 371
5.1 Parallelisierungskonzepte 373
5.2 Parallele Bastionen 375
5.2.1 Funktionale Trennung 376
5.2.2 Zentrale Verteilung von Verbindungen 377
5.2.3 Dezentrale Verteilung von Verbindungen 395
5.3 Parallele Packet Screens 404
5.4 Verringerung der Verbindungsaufbauzeiten 411
5.4.1 Offenhalten von Verbindungen und "Pipelining" 412
5.4.2 Leichtgewichtiges TCP 413
5.4.3 "Multiplexing" von Verbindungen 415
5.5 Abschließende Bewertung 417

6 Zusammenfassung und Ausblick 423
6.1 Zusammenfassung der Ergebnisse 423
6.1.1 Firewalls in ATM-Netzen 423
6.1.2 Performanz von Firewalls 428
6.2 Ausblick 431

Anhang 433

A Literaturverzeichnis 435

B Zusätzliche Abbildungen **456**

C Ergänzende Tabellen und Daten **470**
C.1 Durchschnittliche Paket- und Nachrichtenlängen 470
C.2 Packet Screen Konfigurationen 471

D Abkürzungsverzeichnis **475**

E Abbildungsverzeichnis **479**

F Tabellenverzeichnis **486**

G Glossar **487**

Vorwort

Als ich 1992 im Rahmen meiner Diplom-Arbeit mit Untersuchungen zum Thema Firewalls begann, waren noch Internet-Anschlüsse mit einem Datendurchsatz von 9,6 KBit/s üblich. Die Überwachung dieser geringen Datenmengen war damals selbst beim Einsatz von veralteten Systemen als Firewall noch ohne signifikante Performanzverluste möglich.

Mit wachsender Bedeutung des Internets für die tägliche Arbeit stiegen die Durchsätze der Netzanschlüsse in den folgenden Jahren über 64 KBit/s, 2 MBit/s und 34 MBit/s auf 155 MBit/s. Weitere Steigerungen mit Datendurchsätzen im Gigabit-Bereich stehen bevor. Demgegenüber fielen die Leistungssteigerungen bei den Endsystemen deutlich geringer aus, so daß absehbar ist, daß herkömmliche Firewall-Konzepte in Hochgeschwindigkeitsnetzen einen Engpaß darstellen werden.

Zusätzlich zu den Performanzproblemen ergeben sich durch ATM-Netze auch technische Probleme für Firewalls, da sich die Technik von ATM-Netzen wesentlich von herkömmlichen Netzen unterscheidet. Herkömmliche Firewall-Konzepte müssen daher zunächst an ATM-Netze angepaßt werden. Eine Ausnutzung der sich in ATM-Netzen bietenden Möglichkeiten ist aber erst mit neuen Firewall-Konzepten möglich.

Diese Arbeit untersucht detailliert die technischen Aspekte der Integration von Firewalls in ATM-Netze und die Performanz von Firewalls in Hochgeschwindigkeitsnetzen. Darüber hinaus werden verschiedene Lösungsansätze zur Beseitigung der identifizierten Performanzprobleme vorgestellt.

Danksagung

Die vorliegende Arbeit entstand während meiner Tätigkeit als wissenschaftlicher Mitarbeiter im Drittmittelprojekt "Firewall-Labor für Hochgeschwindigkeitsnetze" (DFN-FWL) am Fachbereich Informatik der Universität Hamburg.

An dieser Stelle möchte ich allen danken, die mich bei der Durchführung dieser Arbeit unterstützt haben. Mein besonderer Dank gilt Prof. Dr. Klaus Brunnstein, der diese Arbeit betreut hat, meinem Kollegen Carsten Benecke, der insbesondere durch Diskussionen und die Implementierung von Meßwerkzeugen das Gelingen dieser Arbeit unterstützt hat, und Herrn Dr. Hans-Joachim Mück, der als Leiter des Projekts die Durchführung dieser Arbeit ermöglicht hat. Ebenfalls bedanken möchte ich mich bei allen Kollegen vom DFN-CERT, DFN-PCA und dem Rechenzentrum des Fachbereichs Informatik sowie allen anderen, die mich bei der Entstehung dieser Arbeit unterstützt haben.

Hamburg, den 1. Dezember 1998 Uwe Ellermann

1 Einleitung und Motivation

Inhalt

1.1 Firewalls in neuen Umgebungen 1
1.2 Überblick über die folgenden Kapitel 3

1.1 Firewalls in neuen Umgebungen

Die Bedeutung von Netzwerken hat sich in den letzten Jahren stark gewandelt. Insbesondere der Erfolg des Internets, das zunächst als reines Forschungsnetz geplant war und heute eine zunehmende Kommerzialisierung erfährt, macht dies deutlich. Mit dem Erfolg einhergehend hat sich auch die Art der Nutzung wesentlich verändert. Zusätzlich zu den einfachen Diensten etwa zur Übertragung von E-Mail (SMTP) oder zum Transfer von Dateien (FTP) stehen heute eine Reihe weiterer Dienste (WWW, MBone usw.) im Internet zur Verfügung. Diese zusätzlichen Dienste sind zum Teil deutlich komplexer und erfordern auch einen höheren Durchsatz als die einfachen Dienste, die es schon seit den Anfangsjahren des Internets gibt.

Zusammen mit der steigenden Beliebtheit des Internets nehmen auch die Sicherheitsprobleme zu, die sich in den rasant ansteigenden Vorfallszahlen der Computer-Notfall-Teams widerspiegeln. Zur Abwehr von Angreifern aus dem Internet werden seit Anfang der 90er Jahre Firewalls eingesetzt. Firewalls kontrollieren die Zugriffe am Übergang zwischen zwei Netzen und lassen nur den Datenverkehr passieren, der vorab definierten Sicherheitsrichtlinien entspricht. Insbesondere Unternehmen wagen es inzwischen kaum noch, ihre Unternehmensnetze ohne Firewalls direkt mit dem Internet zu verbinden. Darüber hinaus werden Firewalls auch innerhalb von Unternehmensnetzen eingesetzt, um besonders sensible Bereiche vom übrigen Netz abzutrennen.

Die anfängliche Haltung, Firewalls als "Allheilmittel" der Netzwerksicherheit anzusehen, hat zunächst heftige Kontroversen ausgelöst. Inzwischen ist jedoch allgemein anerkannt, daß ein Firewall ein Zugriffskontrollmechanismus in Netzen ist

und als Zugriffskontrollmechanismus einen wichtigen Beitrag zum Schutz gegen Angriffe leistet. Gleichzeitig ist aber auch klar, daß nicht alle Sicherheitsprobleme allein durch einen Zugriffskontrollmechanismus gelöst werden können. Ein Sicherheitskonzept muß daher neben Firewalls noch weitere Sicherheitsmechanismen wie z.B. kryptographische Verfahren enthalten, um die verbleibenden Risiken abzudecken.

Um eine Kontrolle am Übergang zwischen zwei Netzen durchführen zu können, bilden Firewalls einen Engpaß, den der zwischen den beiden Netzen ausgetauschte Verkehr passieren muß. Dieser Engpaß ermöglicht einerseits erst die Kontrolle der übertragenen Daten, droht aber andererseits auch hinsichtlich der Performanz zum Engpaß zu werden. Daher stellt sich bereits seit Beginn des Einsatzes von Firewalls die Frage nach der Performanz.

Als Anfang der 90er die ersten Firewalls eingesetzt wurden, waren Unternehmens- und Universitätsnetze noch über 9,6 KBit/s Leitungen mit dem Internet verbunden. Eine Überwachung dieser geringen Datenmengen war damals selbst mit veralteten Systemen noch ohne Performanzverluste möglich. Mit wachsender Bedeutung des Internets mußte auch der Durchsatz des Netzanschlusses über 64 KBit/s und 2 MBit/s auf 34 MBit/s oder sogar 155 MBit/s erhöht werden. Heute sind bereits weitere Steigerungen bis zu Durchsätzen im Gigabit-Bereich absehbar.

Im gleichen Zeitraum stieg zwar auch die Leistungsfähigkeit durchschnittlicher Workstations erheblich, trotzdem blieben aber diese Leistungssteigerungen hinter den Steigerungen des Netzdurchsatzes zurück. Daher ist bereits absehbar, daß Firewalls, die üblicherweise auf Workstations basieren, bei steigenden Netzdurchsätzen einen Performanzengpaß bilden werden.

Die Performanzprobleme bei Firewalls verschärfen sich zusätzlich dadurch, daß neben den geforderten Durchsätzen auch die Aufgaben von Firewalls zugenommen haben. Heute zählen zu den Aufgaben von Firewalls die "Network Address Translation" (NAT), die Verschlüsselung übertragener Daten, aufwendige kryptographische Authentisierungsverfahren, die Untersuchung übertragener Daten auf Viren und auf andere unerwünschte ausführbare Inhalte ("Executable Content Scanner"). Diese zusätzlichen Aufgaben sind zum Teil sehr rechenintensiv, so daß es zusammen mit der gleichzeitigen Steigerung der Netzdurchsätze offensichtlich ist, daß Firewalls zum Performanzengpaß werden müssen.

Zusätzlich zu den Performanzproblemen entstehen durch den Einsatz von Hochgeschwindigkeitsnetzen teilweise auch technische Probleme für Firewalls. Während sich beispielsweise Fast- und Gigabit-Ethernet konzeptionell kaum von der herkömmlichen Ethernet-Technologie unterscheiden und daher auch keine konzeptionellen Schwierigkeiten für Firewalls darstellen, unterscheidet sich die Technik von ATM-Netzen erheblich von herkömmlichen Netzen. Die an herkömmliche

datagrammorientierte Netze angepaßten Firewall-Konzepte eignen sich daher nur unzureichend für verbindungsorientierte ATM-Netze. Abgesehen von diesen technischen Schwierigkeiten bietet die ATM-Technik aber auch eine Reihe von Ansatzpunkten für verbesserte Zugriffskontrollmechanismen, so daß die ATM-Technik nicht nur ein Problem, sondern auch eine Chance für neue Firewall-Konzepte darstellt. Eine Ausnutzung der sich in ATM-Netzen bietenden Möglichkeiten ist nur durch neue Firewall-Konzepte möglich.

1.2 Überblick über die folgenden Kapitel

Das Ziel der vorliegenden Arbeit ist die Untersuchung der Auswirkungen von Hochgeschwindigkeitsnetzen auf Firewalls. Wie einleitend dargestellt, sind Hochgeschwindigkeitsnetze hinsichtlich der Performanz und im Falle von ATM auch aufgrund der unterschiedlichen Technologie eine Herausforderung für Firewalls. Beide Aspekte der Auswirkungen von Hochgeschwindigkeitsnetzen auf Firewalls sollen in dieser Arbeit untersucht werden.

Kapitel 2 führt in den Problembereich von Firewalls in Hochgeschwindigkeitsnetzen ein. Dafür werden zunächst die heute im Internet auftretenden Angriffe vorgestellt und kategorisiert. Die Kenntnis dieser Angriffe ist eine Voraussetzung für die Entwicklung von Sicherheitsmechanismen. Die beiden wichtigsten Sicherheitsmechanismen, die netzbasierten Angriffen entgegenwirken, sind kryptographische Verfahren und Firewalls. Nach einem kurzen Überblick über die Anwendung kryptographischer Verfahren zur Absicherung übertragener Daten folgt eine Einführung in die Konzepte von Firewalls. Die Diskussion von Firewalls beschreibt zunächst Firewall-Komponenten, die einzelne wichtige Teilaufgaben eines Firewalls implementieren, und anschließend Firewall-Architekturen, die aus diesen Firewall-Komponenten aufgebaut werden.

Da sich ATM-Netze stark von herkömmlichen Netzen unterscheiden, ist ebenfalls eine Einführung in die Techniken von ATM-Netzen erforderlich. Auch in ATM-Netzen wird die Übertragung von IP-Dateneinheiten weiterhin eine wichtige Rolle spielen. Daher werden mit "Classical IP", NHRP, "LAN Emulation" und MPOA die wichtigsten Ansätze zur Übertragung von Dateneinheiten herkömmlicher Protokolle über ATM-Netze vorgestellt. Das Kapitel schließt mit einer Untersuchung der Unterschiede zwischen ATM und herkömmlichen Protokollen.

Kapitel 3 und Kapitel 4 bilden den Kern dieser Arbeit. In Kapitel 3 werden die neuen Möglichkeiten und die zusätzlichen Probleme beim Einsatz von Firewalls

in ATM-Netzen untersucht. Kapitel 4 analysiert die Performanz von Firewalls in Hochgeschwindigkeitsnetzen.

Für den Einsatz von Firewalls ist in Kapitel 3 zunächst eine Untersuchung der in ATM-basierten Netzen vorhandenen Risiken erforderlich. Da sich ATM-Netze stark von herkömmlichen Netzen unterscheiden, sind in ATM-Netzen auch andere Schwachstellen vorhanden als in herkömmlichen Netzen. Dies gilt insbesondere auch für die zusätzlichen Dienste, die für die Übertragung von IP-Dateneinheiten bei CLIP und NHRP erforderlich sind. Die Schwachstellen von ATM, CLIP und NHRP werden ausführlich diskutiert. Dabei wird gezeigt, daß ATM-basierte Netze nicht sicherer sind als herkömmliche Netze.

Als verbindungsorientiertes Netz erfordert ATM vor jeder Kommunikation zunächst den Aufbau einer virtuellen Verbindung. Die Signalisierung über PNNI bzw. UNI, die zum Aufbau einer virtuellen Verbindung verwendet wird, eignet sich hervorragend für den Einsatz von Zugriffskontrollmechanismen. Abschnitt 3.3.1 untersucht die Möglichkeiten für eine Zugriffskontrolle auf Signalisierungsnachrichten. Neuartige Möglichkeiten für den Einsatz von Zugriffskontrollmechanismen ergeben sich durch die Manipulation von Signalisierungsnachrichten, da es durch diese Manipulation möglich wird, virtuelle Verbindungen umzuleiten, um dadurch zusätzliche Sicherheitskomponenten wie z. B. Proxies einzubinden. Diese Sicherheitsdienstgüte-basierte Wegwahl eröffnet neue Möglichkeiten für den Einsatz von Zugriffskontrollmechanismen. Diese Möglichkeiten bestanden in herkömmlichen Netzen nicht.

Zur Überwachung der auf einer virtuellen Verbindung übertragenen Daten können die bekannten Firewall-Komponenten Packet Screen und Proxy verwendet werden. Für Packet Screens stellt der Einsatz von ATM ein besonderes Problem dar, weil die von Packet Screens gefilterten IP-Dateneinheiten in einem ATM-Netz auf mehrere Zellen aufgeteilt werden. Abschnitt 3.3.2 untersucht die verschiedenen Möglichkeiten zur Implementation einer Packet Screen in einem ATM-Netz.

Abschnitt 3.3.3 analysiert den Einsatz von Proxies in ATM-Netzen. Für den Einsatz von Proxies stellt ATM weitaus geringere technische Probleme dar als für Packet Screens. Die Art der Übertragung hat nur geringen Einfluß auf die Verarbeitung der Daten in einem Proxy, da Proxies Daten auf Anwendungsebene verarbeiten. Herkömmliche Proxies können daher unverändert auch in CLIP- oder NHRP-Netzen eingesetzt werden. Der Einsatz von Proxies für "Native"-ATM-Anwendungen wird am Beispiel der Anwendung "Video on Demand" (VoD) untersucht. Ein besonders wichtiger Aspekt für "Native"-ATM-Proxies stellt die Anforderung von Dienstgüte-Eigenschaften durch die Anwendungen dar. Eingehend wird untersucht, welche Dienstgüte-Eigenschaften von "Native"-ATM-Proxies unterstützt werden müssen.

Ein neues Konzept stellen transparente Proxies für NHRP-Netze dar, die im Abschnitt 3.3.3.3 vorgestellt werden. Das Konzept des transparenten Proxies ermöglicht das Einbinden eines Proxies in eine virtuelle Verbindung, ohne eine explizite Unterstützung der Kommunikationspartner zu erfordern. Dadurch können Proxies beim Aufbau einer virtuellen Verbindung über eine Sicherheitsdienstgütebasierte Wegwahl eingebunden werden. Die Realisierbarkeit eines transparenten Proxies für NHRP-Netze wird durch die Implementation eines Prototyps demonstriert.

Abschnitt 3.4 untersucht Firewall-Architekturen für ATM-Netze, die aus den im Abschnitt 3.3 vorgestellten Firewall-Komponenten aufgebaut werden. Ausgangspunkt ist dabei die direkte Migration herkömmlicher Firewall-Architekturen, die allerdings starke Einschränkungen zur Folge hat. Erst durch den Einsatz der Zugriffskontrolle auf Signalisierungsnachrichten können leistungsfähige Firewall-Architekturen für ATM-Netze entwickelt werden. Vorgestellt werden Firewall-Architekturen für "Native"-ATM-Anwendungen sowie das neue Konzept des "virtuell zentralen Firewalls", durch das ein ATM-basiertes Netz sehr flexibel in unterschiedliche Sicherheitsbereiche aufgeteilt werden kann.

In Kapitel 4 wird die Performanz von Firewalls in Hochgeschwindigkeitsnetzen detailliert untersucht. Durch Messungen an Testkonfigurationen in einem ATM-Netz werden dabei die Engpässe unterschiedlicher Firewall-Komponenten analysiert.

Nach einer kurzen Einführung der Performanzkriterien, die bei Firewalls untersucht werden müssen, folgt die Vorstellung der Testumgebung. Die genaue Beschreibung des Testnetzes ist erforderlich, um Vergleichswerte für spätere Messungen der Performanz von Firewalls zu ermitteln. Dabei erfolgt gleichzeitig eine Beschreibung der Meßmethodik. In Ergänzung zum Standard-Meßwerkzeug "Netperf" muß für die Performanzmessungen bei Proxies ein spezielles Meßwerkzeug entwickelt werden, mit dem auch Messungen mit parallelen Verbindungen durchgeführt werden können.

Bei den Messungen zur Performanz von Firewalls werden die Firewall-Komponenten Packet Screen und Proxy unabhängig voneinander untersucht, wodurch die Engpässe der einzelnen Firewall-Komponenten jeweils exakt bestimmt werden können. Schwerpunkt der Untersuchungen bildet die Performanz von Proxy-Servern, da diese aufgrund der Verarbeitung der übertragenen Daten auf Anwendungsebene besonders starke Auswirkungen auf die Performanz haben. Ziel der Messungen ist dabei nicht die Ermittlung der Performanz bestimmter Firewall-Produkte. Stattdessen sollen die konzeptionsbedingten Engpässe von Firewall-Komponenten ermittelt und bewertet werden. Zu diesem Zweck wird ein prototypischer Proxy-Server entwickelt und hinsichtlich der Performanz optimiert. Durch

die leichte Erweiterbarkeit bietet dieser Proxy-Server die Möglichkeit die Auswirkungen unterschiedlicher Implementationsalternativen experimentell zu untersuchen. Darüber hinaus bildet dieser Proxy-Server auch eine flexible Basis für Untersuchungen der Auswirkungen von kryptographischen Authentisierungsverfahren, von Verschlüsselung und von "Executable Content Scannern".

Kapitel 5 zeigt verschiedene Möglichkeiten zur Beseitigung der im Kapitel 4 identifizierten Performanzengpässe von Firewall-Komponenten auf. In erster Linie bieten sich hier Parallelisierungskonzepte an, um die Last auf parallele Firewall-Komponenten zu verteilen. Für den Einsatz bei Proxies werden unterschiedliche Verfahren zur zentralen oder dezentralen Verteilung von Verbindungen auf parallele Bastionen untersucht. Für Packet Screens eignen sich ausschließlich dezentrale Verteilungsverfahren, da hier ein zentraler Verteiler schnell zum Engpaß wird. Eine Bewertung der Leistungsfähigkeit der unterschiedlichen Ansätze erfolgt dabei anhand der im Kapitel 4 gewonnenen Erkenntnisse über die Engpässe bei Firewall-Komponenten.

Ansätze zur Verringerung der Verbindungsaufbauzeiten bei Proxies werden ebenfalls im Kapitel 5 untersucht. Da die parallele Verarbeitung von Verbindungen auf parallelen Bastionen hier keine Verringerungen der Verbindungsaufbauzeiten ermöglicht, müssen andere Lösungsansätze wie z. B. effizientere Protokolle untersucht werden.

Im Kapitel 6 werden die Ergebnisse der Untersuchungen zusammengefaßt, und es wird ein kurzer Ausblick auf zukünftige Entwicklungen gegeben.

2 Überblick über den Problembereich

Inhalt

2.1 Überblick über netzbasierte Angriffe 8
2.1.1 Konzeptionelle Schwächen 9
2.1.1.1 Fehlen von Vertraulichkeit 10
2.1.1.2 Authentizität nicht überprüfbar 12
2.1.1.3 Keine Gewährleistung der Integrität 15
2.1.1.4 Schwächen bei der Gewährleistung der Verfügbarkeit 17
2.1.2 Implementationsfehler 19
2.1.3 Konfigurationsfehler 21
2.2 Absicherung durch kryptographische Verfahren 21
2.3 Firewalls 27
2.3.1 Firewall-Komponenten 29
2.3.1.1 Packet Screen 29
2.3.1.2 Proxy-Server 33
2.3.1.3 Anwendungs-Gateway 35
2.3.1.4 Weitere Firewall-Komponenten 36
2.3.2 Firewall-Architekturen 36
2.3.2.1 Packet Screens 37
2.3.2.2 Kombinationen von Packet Screen und Bastion . . . 39
2.3.2.3 Gateway-Firewalls 42
2.3.2.4 Firewall-Ergänzungen 43
2.3.3 Nicht durch Firewalls abgedeckte Sicherheitsprobleme 44
2.4 Überblick über die Techniken von ATM-Netzen 45
2.4.1 Kommunikation in ATM-Netzen 48
2.4.2 "Classical IP" (CLIP) 55

2.4.3 "Next Hop Resolution Protocol" (NHRP) 56
2.4.4 "Multicast Address Resolution Server" (MARS) 59
2.4.5 "Label-Switching" 59
2.4.6 "LAN Emulation" (LANE) 62
2.4.7 "Multi-Protocol over ATM" (MPOA) 63
2.4.8 Unterschiede zwischen ATM und herkömmlichen Netzen . . . 64
2.5 Neue Herausforderungen für Firewalls 68

Dieses Kapitel gibt einen Überblick über den Problembereich von Firewalls in Hochgeschwindigkeitsnetzen. Ziel ist eine Beschreibung des aktuellen Stands der Technik, um dabei die für die folgenden Kapitel notwendigen Grundlagen zu schaffen. Zunächst wird ein Überblick über netzbasierte Angriffe gegeben. Darauf folgend werden kryptographische Verfahren und Firewalls vorgestellt, die als Sicherheitsmechanismen gegen die beschriebenen Angriffe eingesetzt werden können. Im Abschnitt 2.4 folgt dann eine Einführung in die Techniken von ATM-Netzen, die sich deutlich gegenüber herkömmlichen Netzen unterscheiden. Abschließend werden die zukünftigen Herausforderungen für Firewalls vorgestellt um aufzuzeigen, in welchen Bereichen weitere Entwicklungen notwendig sind.

2.1 Überblick über netzbasierte Angriffe

Netzwerke haben in den letzten Jahren ein enormes Wachstum erfahren. Gleichzeitig ist damit auch die Verletzlichkeit gestiegen, da Netze wie das Internet heute nicht mehr als reine "Spielwiese" der Wissenschaftler dienen. Einhergehend mit dem Wachstum im Bereich der Netzwerke stellen Computer-Notfall-Teams wie das DFN-CERT eine starke Zunahme der netzbasierten Angriffe fest. Dieser Abschnitt soll einen Überblick über die netzbasierten Angriffe im Internet geben, um eine Grundlage für die im Abschnitt 3.2 durchgeführten Untersuchungen der Schwachstellen von ATM-Netzen zu bilden und um den für die Analyse von Sicherheitsmechanismen erforderlichen Hintergrund zu liefern. Die Kenntnis der Angriffe, die ein Netz bedrohen, ist eine Voraussetzung für den Entwurf von Sicherheitsmechanismen und das Erstellen von Sicherheitsrichtlinien. Untersucht werden im folgenden Angriffe auf ein Netz und die angeschlossenen Komponenten. Diese Angriffe können sich gegen die Infrastruktur (Router, Switches) richten, oder es werden Schwachstellen in den Endsystemen ausgebeutet. Nicht diskutiert werden Angriffe auf die Host-Sicherheit, mit denen z. B. Angreifer, die bereits eine Benutzerberechtigung erlangt haben, höhere Privilegien erzielen.

Die Sicherheitsprobleme im Internet — damals noch "ARPAnet" genannt — wurden bereits früh erkannt [Metcalfe 73, Salus 95]. 1973 hatte das ARPAnet

erst 31 (einunddreißig !) Hosts [NSF 95]. Die damals identifizierten Probleme — ratbare Paßworte und Studenten, die das Einbrechen in fremde Rechner als Sport betreiben — sind auch heute noch im Internet zu finden. Zusätzlich dazu sind jedoch neue Attacken hinzugekommen. Trotzdem kann ein Übergang von diesen einfachen Angriffen zu neuen, "intelligenteren" Angriffsformen nicht von CERTs beobachtet werden. Vielmehr bleiben die einfachen Angriffe zusätzlich zu den neu hinzugekommenen Angriffsformen weiterhin erhalten.

Das Auflisten aller bekannten Angriffe ist heute kaum noch möglich, so daß eine Klassifikation der Angriffe erforderlich wird. Inzwischen gibt es bereits eine Reihe von Ansätzen zur Klassifikation von Angriffen [Howard 97, Chapter 6].

In [Chapman et al. 95] erfolgt eine Klassifikation nach den zu schützenden Werten (Daten, Ressourcen oder der Ruf eines Unternehmens) sowie eine Klassifikation nach der Art der Angriffe (Systemeinbrüche, "Denial of Service"-Angriffe, Informationsdiebstahl usw.). Häufig werden Angriffe auch nach den Angreifern (interne vs. externe Angreifer) oder nach den Folgen des Angriffs (Preisgabe von Informationen, Manipulation von Daten, Verlust der Verfügbarkeit) klassifiziert. Beim Entwurf und bei der Auswahl von Sicherheitsmechanismen hat es sich als vorteilhaft erwiesen, nach den bei den Angriffen ausgebeuteten Schwachstellen vorzugehen, die durch geeignete Sicherheitsmechanismen abgesichert werden müssen. Schwachstellen werden zunächst grob unterteilt in konzeptionelle Schwächen, Implementationsfehler und Konfigurationsfehler. Die konzeptionellen Schwächen werden zusätzlich weiter unterteilt nach der Art der Schwachstelle in Schwächen bei Gewährleistung von Vertraulichkeit, Integrität, Authentizität und Verfügbarkeit.

Implementations- und Administrationsfehler wurden bereits in diversen Büchern und Artikeln ausgiebig diskutiert.[1] Auch konzeptionelle Schwächen in den Netzwerkprotokollen sind zwar schon lange bekannt [Bellovin 89, Morris 85], werden aber erst ausgebeutet, seit sogenannte "Hackertools" verfügbar sind. Auch bei den Implementations- und Administrationsfehlern zeigt sich, daß Angriffe auf bekannte Schwächen erst dann stattfinden, wenn einfach zu bedienende "Hackertools" eine entsprechende Verbreitung gefunden haben. Nach den Erfahrungen des DFN-CERTs und anderer Computer-Notfall-Teams kann die Mehrheit der Angreifer weder "Hackertools" selbst entwickeln, noch haben sie das notwendige Wissen, um die Angriffe im einzelnen zu verstehen.

2.1.1 Konzeptionelle Schwächen

Bei den Angriffen auf die Netzwerkprotokolle wird von konzeptionellen Schwächen oder Konzeptions*lücken*, nicht aber von Konzeptions*fehlern* gesprochen, weil

[1] Eine sehr empfehlenswerte Einführung findet sich in [Garfinkel et al. 96].

die Konzepte nicht im eigentlichen Sinne fehlerhaft sind, sondern lediglich den Anforderungen der heutigen Einsatzbereiche nicht mehr gerecht werden. Dies liegt daran, daß heute wichtige Anforderungen erst im Laufe der Zeit eine stärkere Bedeutung erlangt haben.

In den Anfangsjahren des Internets waren Mechanismen zum Schutz der Vertraulichkeit, Authentizität und Integrität der übertragenen Daten sowie Maßnahmen zum Schutz der Verfügbarkeit nicht erforderlich. Bei der damals geringen Anzahl von Rechnern und Nutzern war zudem die Bedrohung gering. Heute ermöglichen diese Konzeptionslücken Angriffe durch "Sniffer", "IP-Spoofing", "TCP-Hijacking" und zahllose "Denial of Service"-Angriffe. Da diese vier Angriffsformen zu schwerwiegenden Risiken führen, werden sie im folgenden ausführlich diskutiert.

2.1.1.1 Fehlen von Vertraulichkeit

Vertraulichkeit bezeichnet die Eigenschaft, daß eine Nachricht nur durch den vom Sender vorgegebenen Empfängern lesbar ist.[2] Der Fehlen der Vertraulichkeit ist heute eine der wichtigsten konzeptionellen Schwächen im Internet. Die Anzahl der durch fehlende Vertraulichkeit ermöglichten "Sniffer"-Angriffe ist zwar leicht rückläufig aber trotzdem zählen sie heute noch zu den am häufigsten an Computer-Notfall-Teams gemeldeten Angriffen [DFN-CERT 98].

Mit Hilfe von "Sniffern" ist es möglich, das Netzwerk abzuhören und die dort übertragenen Benutzernamen und Paßworte aufzuzeichnen. Diese Informationen ermöglichen dem Angreifer anschließend ein Einloggen auf dem Zielrechner.

"Sniffer" werden seit Jahren von Administratoren benutzt, um Netzwerkprobleme zu analysieren. Seit Anfang 1994 sind "Sniffer" auch als "Hackertool" verfügbar — spezialisiert auf das Abhören von Accountnamen und Paßworten [CERT 94:01]. Seitdem bereiten "Sniffer"-Angriffe Anwendern, Netzwerkbetreibern und CERTs[3] Probleme. "Sniffer" nutzen zwei Eigenschaften heutiger Netze:

1. Die Daten werden unverschlüsselt von Sender zum Empfänger übertragen. Somit sind die Daten für jeden verständlich, der sie auffangen kann.

[2] Die Vertraulichkeit kann dabei nicht nur für die Nachricht, sondern auch für die Kommunikationsbeziehung selbst erforderlich sein. Die Vertraulichkeit von Kommunikationsbeziehungen verhindert, daß ein Angreifer Erkenntnisse darüber gewinnt, welche Systeme miteinander kommunizieren.

[3] Für die Netzwerkbetreiber bedeutet ein "Sniffer"-Angriff oft ein langwieriges Säubern der Systeme. Für Anwender ist bei einem "Sniffer"-Angriff oft schwer festzustellen, wer betroffen ist — das Vertrauen in die Rechner ist gestört. Für CERTs stellen "Sniffer"-Angriffe eine große Arbeitslast dar, weil meist hunderte von Rechnern in diversen Domains betroffen sind und alle Administratoren einzeln benachrichtigt werden müssen.

2. In "Broadcast"-Netzen wie dem in LANs gebräuchlichen Ethernet empfängt jede angeschlossene Station alle auf dem Netz übertragenen Daten. Im normalen Betrieb ignoriert eine Station alle Daten, die nicht an sie adressiert sind. Wird die Ethernet-Schnittstelle in den "Promiscuous Mode" umgeschaltet, werden auch die Daten an die anfordernde Applikation weitergeleitet, die an fremde Stationen adressiert sind.

Diese beiden Eigenschaften werden von "Sniffern" ausgenutzt, indem sie gezielt die Daten solcher Verbindungen aufzeichnen, bei denen Paßworte zur Authentisierung übertragen werden (Telnet, FTP und "rlogin"). Da die Paßworte zu Beginn einer Verbindung übertragen werden, reicht es in der Regel aus, nur die ersten etwa 100 Bytes aufzuzeichnen.

In Abbildung 2.1 ist ein typisches Szenario eines "Sniffer"-Angriffs wiedergegeben: Ein Benutzer meldet sich mit Hilfe des Telnet-Protokolls auf dem Rechner "`server.com`" an. Zur Authentisierung wird der Benutzer aufgefordert, Benutzername und Paßwort anzugeben. Diese Benutzereingaben — auch das Paßwort — werden unverschlüsselt übertragen und können vom "Sniffer" aufgezeichnet werden. Sind nach einiger Zeit genügend Accountnamen und Paßworte aufgezeichnet worden, werden die vom "Sniffer" aufgezeichneten Daten vom Angreifer abgeholt. Der Angreifer kann sich nun mit den abgehörten Informationen auf dem betroffenen Rechner (in diesem Fall `server.com`) anmelden. Dort wird dann nach üblichen Schwachpunkten gesucht, um die notwendigen System-Privilegien zu erlangen, um auch hier wieder einen "Sniffer" zu installieren. Wenn ein "Sniffer" entdeckt wird, werden bei der Aufklärung des Vorfalls durch CERTs fast immer eine Vielzahl weiterer "Sniffer" auf anderen Rechnern entdeckt. Die Anzahl der betroffenen Accounts ist daher bei "Sniffer"-Vorfällen erheblich höher als bei anderen Angriffen.

Zur Vermeidung bzw. zur Verringerung der Auswirkungen von "Sniffer"-Angriffen wird in [CERT 94:01] die Verwendung von nicht wiederverwendbaren Paßworten empfohlen. Solche Paßworte können zwar weiterhin von einem "Sniffer" aufgezeichnet werden, jedoch ist ein Einloggen für den Angreifer nicht mehr möglich. Die verschiedenen Systeme für nicht wiederverwendbare Paßworte haben bis heute immer noch eine geringe Verbreitung, da sie nachträglich installiert werden müssen und für Benutzer deutlich umständlicher zu bedienen sind als normale Paßworte. Durch die Verfügbarkeit von "TCP-Hijacking-Tools" (siehe Abschnitt 2.1.1.3) ist der Einsatz von nicht wiederverwendbaren Paßworten allein auch nicht mehr ausreichend zur Authentisierung.

Der Einsatz von nicht wiederverwendbaren Paßworten ist eine Maßnahme, die sich gezielt gegen derzeit stattfindende "Sniffer"-Angriffe richtet, die Paßworte aufzeichnen. "Sniffer" lassen sich jedoch grundsätzlich auch zum Aufzeichnen von anderen interessanten Informationen nutzen: vertrauliche E-Mails, Datenbank-

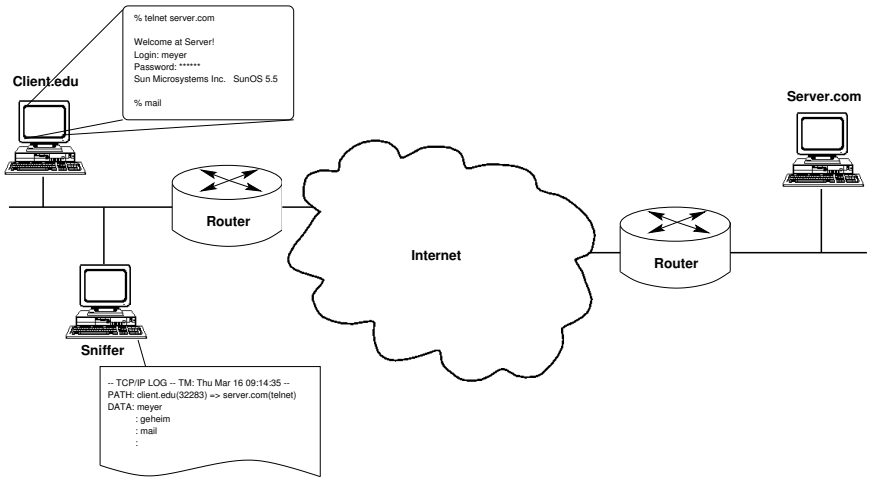


Abbildung 2.1: "Sniffer"-Angriff

Anfragen oder Kreditkarten-Nummern. Als Maßnahme zum Schutz der Vertraulichkeit kommt in diesen Fällen ausschließlich Verschlüsselung in Frage. Die Anwendungen von Verschlüsselungsverfahren zur Sicherung von Kommunikation werden in Kapitel 2.2 vorgestellt.

2.1.1.2 Authentizität nicht überprüfbar

Authentizität bezeichnet die Eigenschaft, daß ein Empfänger einer Nachricht sicher den Sender der Nachricht identifizieren kann. Die Authentizität von übertragenen Daten wird von den meisten Internet-Protokollen nicht gewährleistet. Dieses Problem ermöglicht das Fälschen von Adressen bei sogenannten "Spoofing"-Angriffen. Der Empfänger kann die Adreßinformationen nicht überprüfen und muß daher annehmen, daß die Daten tatsächlich von der angegebenen Adresse stammen. Die beiden wichtigsten "Spoofing"-Angriffe sind das Fälschen von Absender-Adressen bei E-Mails und IP-Datagrammen. Das "Mail-Spoofing" ist sehr einfach durchzuführen, indem der Angreifer eine TCP-Verbindung zum Mail-Server eröffnet[4] und die Nachrichten des Text-basierten "Simple Mail Transfer Protocol" (SMTP) [Postel 82] direkt eingibt, dabei aber eine vom Angreifer gewählte E-Mail-Adresse als Absender-Adresse angibt. Das aufwendigere Fälschen von IP-Absender-Adressen wird im folgenden diskutiert.

Bei einigen Diensten wie zum Beispiel "rlogin" [Kantor 91], "rsh" und NFS [Nowicki 91, Callaghan et al. 95] werden IP-Absender-Adressen für die Authentisierung von Clients verwendet. Daß eine IP-Adresse keine ausreichende Au-

[4] Zum Öffnen der Verbindung eignen sich die Programme `telnet` oder `mconnect`.

thentisierung darstellt, ist bereits lange klar — zu einfach ist es, an dem eigenen System eine andere IP-Adresse einzustellen und sich dadurch erfolgreich zu authentisieren. Wegen der bequemen Bedienung haben Dienste wie "rlogin", "rsh" und NFS trotz dieser Schwäche eine weite Verbreitung gefunden.

Seit Frühjahr 1995 sind "Hackertools" verfügbar, die "IP-Spoofing"-Angriffe auf "rlogin"- und "rsh"-Server ausführen. Die "IP-Spoofing-Tools" generieren die notwendigen Dateneinheiten unter Umgehung der Protokoll-Software und senden die Dateneinheiten anschließend über den direkten Zugriff auf die Ethernet-Schnittstelle. Durch diesen direkten Zugriff entfällt die Notwendigkeit der Umstellung der IP-Adresse des Rechners.

Bevor ein "IP-Spoofing"-Angriff zum Erfolg führen kann, müssen vom Angreifer drei Probleme gelöst werden:

1. Das Opfer sendet seine Antworten an die als Absender-Adresse angegebene IP-Adresse, daher wird der Angreifer die Antworten des Opfers auf die gefälschten Dateneinheiten nie sehen.

2. Der Verbindungsaufbau bei TCP erfolgt über einen "3-Way-Handshake" (vgl. Abb. 2.2). Dabei werden Sequenznummern ausgetauscht, mit deren Hilfe TCP nach dem Verbindungsaufbau kontrolliert, welche der gesendeten Daten bereits von der Gegenseite bestätigt wurden und ob verlorengegangene Daten erneut gesendet werden müssen. Um den "Handshake" erfolgreich abzuschließen, ist es notwendig, die vom Opfer gesendete Sequenznummer zu kennen.

3. Der Rechner, dessen IP-Adresse der Angreifer in den gefälschten Datagrammen als Absender-Adresse verwendet, erhält die Antworten des Opfers. Da die erste Dateneinheit des "Handshakes" aber nicht von diesem Rechner gesendet wurde, kann die empfangene zweite Dateneinheit des "Handshakes" nicht ausgewertet werden. Der Rechner schickt daher ein "Reset" (RST) an das Opfer und fordert es damit auf, die Verbindung abzubrechen. Der Angreifer kann den "Handshake" nie erfolgreich beenden, wenn der Rechner, dessen IP-Absender-Adresse er fälscht, die Verbindung sofort durch ein RST abbricht.

Die analysierten "IP-Spoofing-Tools" lösen die drei genannten Probleme: Sie können einen Angriff auch ohne Erhalt der Antworten erfolgreich durchführen. Die für einen erfolgreichen "Handshake" notwendigen Informationen werden mit hoher Trefferquote erraten, und durch das vorübergehende Lähmen des Rechners, dessen IP-Absender-Adresse gefälscht wird, kann das Senden von RSTs verhindert werden.

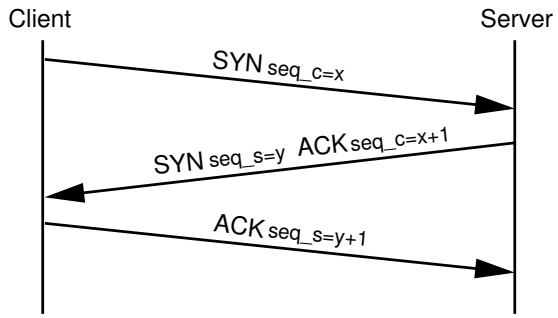


Abbildung 2.2: TCP-Verbindungsaufbau ("3-Way-Handshake")

Ein "IP-Spoofing"-Angriff läuft folgendermaßen ab: Zunächst versucht der Angreifer auszukundschaften, für welche IP-Adresse (im folgenden `Quelle` genannt) ein Zugriff über den "rlogin"- oder "rsh"-Server des angegriffenen Rechners (im folgenden `Ziel` genannt) erlaubt ist. Mit diesen Informationen kann der Angreifer das "IP-Spoofing-Hackertool" starten.

Im ersten Schritt sorgt das "IP-Spoofing-Tool" dafür, daß `Quelle` den Angriff nicht durch RSTs oder andere Dateneinheiten stören kann. Dafür wird 20-mal die erste Dateneinheit des "Handshakes" für einen neuen Verbindungsaufbau gesendet. Als Absender-Adresse wird hierfür eine IP-Adresse gewählt, die nicht erreichbar ist, weil sie z. B. hinter einem Firewall gelegen ist. Dadurch wartet `Quelle` vergebens auf den Abschluß des "Handshakes", die Verbindungen bleiben in einem halb-geöffneten Zustand. Einige Implementationen des TCP-Protokolls können nur eine sehr geringe Anzahl von halb-geöffneten Verbindungen verwalten; bei 20 halb-geöffneten Verbindungen können diese Systeme nicht mehr auf weitere Verbindungsaufbau-Versuche reagieren.[5] Dieses System kann in diesem Zustand nicht mehr auf die Dateneinheiten von `Ziel` reagieren und somit auch keine RSTs an `Ziel` senden.

Im nächsten Schritt werden mehrere reguläre Verbindungen zum `Ziel` aufgebaut. Die von `Ziel` gewählten Sequenznummern werden dabei protokolliert. Bei gebräuchlichen TCP/IP-Implementationen werden sich zwei direkt aufeinander folgende Verbindungen immer um einen festen Betrag voneinander unterscheiden. Dadurch wird es möglich, die nächste Sequenznummer mit hoher Wahrscheinlichkeit vorauszusagen.[6]

Jetzt kann der Angreifer die Verbindung mit der gefälschten Absender-Adresse von `Quelle` nach `Ziel` eröffnen. Das "IP-Spoofing-Tool" sendet die erste Daten-

[5] Betroffen sind unter anderem SunOS 4.x, Ultrix und AIX.

[6] Dieser Schritt schlägt fehl, wenn im selben Zeitraum andere Rechner ebenfalls Verbindungen zu `Ziel` aufbauen.

einheit des "Handshakes" und sendet kurz darauf auch die dritte Dateneinheit des "Handshakes". Dieses dritte Dateneinheit enthält die im vorigen Schritt erratene Sequenznummer, die vermutlich von `Ziel` an `Quelle` zurückgesendet worden ist. Wenn die richtige Sequenznummer erraten wurde, glaubt `Ziel` eine Verbindung mit `Quelle` zu haben.

Über diese offene Verbindung kann der Angreifer jetzt in der nächsten Dateneinheit einen oder mehrere Befehle senden, die auf `Ziel` ausgeführt werden sollen. Sind die Befehle ausgeführt, kann die Verbindung abgebrochen werden.

Dieser Angriff wurde bereits 1985 in [Morris 85] beschrieben und in [Bellovin 89] wieder aufgegriffen. Aber erst 10 Jahre später im Frühjahr 1995 sind in Hacker-Kreisen zwei "IP-Spoofing-Tools" aufgetaucht, die seitdem bei einer Reihe von Angriffen zum Einsatz gekommen sind [CERT 95:01].

In [CERT 95:01] werden mehrere Lösungen zur Abwehr von "IP-Spoofing"-Angriffen vorgeschlagen. Ein Teil der Angriffe kann in Routern erkannt und durch Packet Screens abgewehrt werden. Die Authentisierung durch IP-Adressen in "rlogin" und "rsh" wird vornehmlich in LANs verwendet; daß ein Zugriff für andere Systeme im Internet zugelassen wird, ist die Ausnahme. Daher müssen für "IP-Spoofing"-Angriffe in der Regel die IP-Adressen von Systemen im selben LAN gefälscht werden, in dem auch das angegriffene System angeschlossen ist. Da gefälschte Dateneinheiten über den Router ins LAN gesendet werden müssen, kann der Angriff im Router leicht daran erkannt werden, daß die vom Internet aus empfangenen Dateneinheiten eine Absender-Adresse aus dem LAN enthalten.

In [Bellovin 96] wird außerdem vorgeschlagen, zur Minimierung des Problems unterschiedliche Sequenznummer-Startwerte für unterschiedliche Verbindungen zu verwenden. Das Erraten einer korrekten Sequenznummer wird für den Angreifer entscheidend erschwert. Natürlich bleibt ein Erraten grundsätzlich immer möglich, daher ist eine kryptographische Authentisierung vorzuziehen.

2.1.1.3 Keine Gewährleistung der Integrität

Die Integrität bezeichnet die Eigenschaft, daß eine Nachricht eines Senders nicht von Dritten manipuliert wurde. Zusätzlich zu Angriffen auf die Vertraulichkeit und Authentizität der übertragenen Daten, wie sie mit "Sniffern" und "IP-Spoofing"-Angriffen diskutiert wurden, sind auch Angriffe auf die Integrität der übertragenen Daten möglich. Da die Authentizität übertragener Daten nicht überprüft werden kann, ist es auch nicht möglich, Manipulationen an übertragenen Daten (E-Mails, Web-Seiten usw.) zu erkennen. Angreifer können die fehlende Gewährleistung der Integrität ausnutzen, um Daten auf dem Weg zum Empfänger zu manipulieren. Besonders lohnend sind Angriffe auf die Integrität von TCP-Verbindungen.

Bei den meisten Protokollen wird lediglich zu Beginn einer neuen Verbindung eine Authentisierung der Kommunikationspartner (z.B. über Paßworte oder "Challenge-Response"-Verfahren) durchgeführt, bei allen Daten, die anschließend auf dieser Verbindung gesendet werden, wird davon ausgegangen, daß diese Daten von dem vorher authentisierten Kommunikationspartner gesendet wurden. Eine Authentisierung der einzelnen Dateneinheiten der Verbindung findet nicht statt. Die fehlende Authentisierung ermöglicht es einem Angreifer, eigene Dateneinheiten in eine TCP-Verbindung einzufügen. Dieser Angriff wird auch als "TCP-Hijacking" bezeichnet.

Das untersuchte "TCP-Hijacking-Tool" überwacht zunächst ähnlich wie ein "Sniffer" die Verbindungen, die über den Netzstrang geführt werden. Der Unterschied zum "Sniffer" ist dabei, daß die abgehörten Daten nicht in einer Datei gespeichert werden, sondern dem Angreifer über eine leicht zu bedienende Oberfläche dargestellt werden. Der Angreifer kann zwischen den einzelnen Verbindungen umschalten und erhält dieselbe Ausgabe, die auch der richtige Benutzer auf seinem Bildschirm sieht. Dadurch kann der Angreifer erkennen, wann der Benutzer die Authentisierung abgeschlossen hat. Ähnlich wie bei einem "IP-Spoofing"-Angriff generiert das "TCP-Hijacking-Tool" eine Dateneinheit, die anschließend an den Server gesendet wird. Im Gegensatz zu "IP-Spoofing"-Angriffen ist es jedoch nicht notwendig, die Sequenznummer zu erraten, da diese bereits beim Abhören der Verbindung protokolliert wird. Eine gültige Dateneinheit läßt sich daher für den Angreifer vergleichsweise einfach generieren. Die vom "TCP-Hijacking-Tool" gefälschte Dateneinheit enthält einen oder mehrere Befehle, durch die in der Regel eine Hintertür geöffnet wird, über die der Angreifer anschließend in den Server eindringen kann. Die angegriffene Verbindung bricht kurz nach dem Senden der gefälschten Dateneinheit zusammen, da durch die gefälschte Dateneinheit die Sequenznummern zwischen Server und Client desynchronisiert werden. Nachdem der Versuch fehlschlägt, wieder eine Synchronisierung zu erreichen, brechen die Kommunikationspartner die Verbindung ab. In dem Moment, in dem die Verbindung abbricht, sind die Befehle des Angreifers jedoch bereits ausgeführt.

In [Joncheray 95] wird ein ähnlicher Angriff beschrieben, der zwar aufwendiger ist, aber dem Angreifer ermöglicht, eine Verbindung über eine längere Zeit zu manipulieren. Die Verbindung wird dabei nicht nach dem Ausführen eines Kommandos abgebrochen, so daß der Angriff schwerer für das Opfer zu erkennen ist. Jedoch ist es fraglich, in wie weit dieser Angriff wirklich eine reale Gefahr darstellt, da durch wiederholte ACK-Stürme die Verbindungen häufig zusammenbrechen werden und die Last auf den betroffenen Netzen steigen wird, so daß auch andere Verbindungen in Mitleidenschaft gezogen werden.

"TCP-Hijacking"-Angriffe können nur auf aktive Verbindungen ausgeführt werden. Daher muß ein Angreifer mitunter eine längere Zeit warten, bis ein Benutzer eine Verbindung zum gewünschten Server aufbaut. Da die TCP-Verbindung

nach einem Angriff abbricht, kann der Angriff zudem nicht unbemerkt durchgeführt werden. Schließlich kann dieser Angriff auch nur auf Verbindungen durchgeführt werden, die über den Netzstrang geführt werden, an den das System des Angreifers angeschlossen ist. Aufgrund dieser Einschränkungen haben "TCP-Hijacking"-Angriffe nie eine nennenswerte Verbreitung gefunden. Andererseits ist eine Abwehr durch Packet Screens, wie sie gegen "IP-Spoofing"-Angriffe eingesetzt werden, nicht möglich. Dieser Angriff kann noch nicht einmal durch Proxy-Server verhindert werden, da der Angreifer die Authentisierung beim Proxy abwarten kann, bevor er die gefälschte Dateneinheit sendet. Der einzige wirksame Schutz gegen "TCP-Hijacking"-Angriffe ist die kryptographische Authentisierung aller Dateneinheiten, die zu einer Verbindung gehören, einschließlich der verwendeten Sequenznummern.

2.1.1.4 Schwächen bei der Gewährleistung der Verfügbarkeit

Die Verfügbarkeit bezeichnet die Eigenschaft, daß ein System mit seinen Diensten und den dort gespeicherten Daten von einem autorisierten Benutzer bzw. dessen Prozessen zum gewünschten Zeitpunkt genutzt werden kann und einwandfrei funktioniert. Mit der steigenden Abhängigkeit von der Datenkommunikation wird die Sicherung der Verfügbarkeit des Netzes erforderlich. In jüngster Zeit wurden vermehrt sogenannte "Denial of Service"-Angriffe beobachtet, bei denen Schwächen in den Internet-Protokollen ausgenutzt werden, um die Nutzung des Netzes durch berechtigte Benutzer einzuschränken.

Drei "Denial of Service"-Angriffe, bei denen konzeptionelle Schwächen der Internet-Protokolle ausgenutzt werden, werden im folgenden dargestellt.

ICMP-"Destination Unreachable": IP-Dateneinheiten werden im Internet auf dem Weg vom Sender zum Empfänger von Routern weitergeleitet. Jeder Router ermittelt dabei anhand der ihm vorliegenden "Routing-Informationen" den nächsten Router zum Empfänger, an den die IP-Dateneinheit weitergeleitet werden muß. Eine IP-Dateneinheit, die vom Router nicht weitergeleitet werden kann, weil kein Weg zum Empfänger bekannt ist oder weil der Empfänger nicht erreichbar ist, wird vom Router vernichtet. Damit der Sender keine weiteren IP-Dateneinheiten an diese Adresse sendet, benachrichtigt der Router den Sender mit einer ICMP-"Destination Unreachable" über die fehlende Erreichbarkeit des Empfängers. Durch das Fälschen von ICMP-"Destination Unreachable"-Nachrichten kann ein Angreifer die Erreichbarkeit bestimmter Adressen einschränken. Da die Authentizität einer ICMP-Nachricht nicht überprüft werden kann, besteht für den Empfänger des ICMP-"Destination Unreachables" keine Möglichkeit, die gefälschten Nachrichten zu erkennen.

"Broadcast-Ping": Über einen "Broadcast" kann eine Dateneinheit gleichzeitig an alle Systeme eines Netzes gesendet werden. IP-Dateneinheiten, die an die "Broadcast"-Adresse `192.168.14.255` gesendet werden, erreichen beispielsweise alle Systeme des Subnetzes `192.168.14.0`.

Für die Überprüfung der Erreichbarkeit eines Systems wird üblicherweise das Werkzeug "Ping" verwendet, das eine ICMP-"Echo Request"-Nachricht an das System sendet. Der Empfang einer ICMP-"Echo Request"-Nachricht wird mit einem ICMP-"Echo Reply" beantwortet. Ein an eine "Broadcast"-Adresse gesendeter ICMP-"Echo Request" wird entsprechend von allen Systemen dieses Subnetzes beantwortet. Diese Multiplikation der Dateneinheiten wird bei "Broadcast-Ping"-Angriffen ausgenutzt, indem ICMP-"Echo Requests" mit einer gefälschten Absender-Adresse ("IP-Spoofing", s. Abschnitt 2.1.1.2) an mehrere "Broadcast"-Adressen gesendet werden. Die als Antwort zurückgesendeten ICMP-"Echo Reply"-Dateneinheiten von hunderten oder tausenden von Systemen treffen daraufhin bei dem System ein, dessen IP-Adresse als gefälschte Absender-Adresse angegeben wurde. Durch diesen Angriff ist nicht nur das System betroffen, dessen IP-Adresse gefälscht wurde, sondern es wird auch die Verfügbarkeit der Netze und Router durch diese "Paket-Stürme" beeinträchtigt. [CERT 98:01]

UDP Echo: Die meisten Systeme bieten eine Reihe einfacher Dienste an, die die Uhrzeit liefern ("Daytime", [Postel 83c]), den Zeichensatz ausgeben ("Chargen", [Postel 83b]) oder einfach nur an sie gesendeten Daten unverändert zurücksenden ("Echo", [Postel 83a]). Diese Dienste stehen sowohl in einer verbindungsorientierten TCP-Version als auch in einer datagrammorientierten UDP-Version zur Verfügung. Durch das Fälschen der Adreßinformationen beim "IP-Spoofing" kann nicht nur eine beliebige IP-Adresse als Absender-Adresse eingetragen werden, sondern es ist auch möglich, den Absender-Port des UDP- oder TCP-"Headers" zu fälschen. Ein Angreifer kann dadurch sowohl als Empfänger- als auch als Absender-Port jeweils einen der obengenannten Dienste einsetzen. Da bei UDP-Diensten kein Verbindungsaufbau erfolgt, werden die Antworten des angesprochenen Dienstes an den gefälschten Absender-Port zurückgesendet. Auch der durch diese Antwort angesprochene Dienst schickt wiederum eine Antwort zurück, so daß die beiden Dienste sich gegenseitig ansprechen. Durch die Zugriffe steigt die Last auf den derart miteinander verbundenen Systemen. Durch das Verbinden weiterer Systeme kann auch die Last auf dem dazwischenliegenden Netz ansteigen. Der Angreifer braucht bei diesem Angriff nur das erste Datagramm zu senden, das gegenseitige Ansprechen erfolgt anschließend ohne weitere Eingriffe des Angreifers. [CERT 96:01]

Ein grundsätzliches Problem ist, daß die Zugriffe auf Betriebsmitteln nicht oder nur unzureichend kontrolliert werden. Dadurch ist es für Angreifer leicht, den

Durchsatz des Netzes oder andere Betriebsmittel mit "Denial of Service"-Angriffen zu verschwenden. Durch Zugriffskontrollmechanismen wie Firewalls können "Denial of Service"-Angriffe zum Teil eingedämmt werden. Neben den hier beschriebenen konzeptionellen Schwächen können auch die im folgenden Abschnitt diskutierten Implementations- oder Konfigurationsfehler zu einem Verlust der Verfügbarkeit führen.

2.1.2 Implementationsfehler

Bei der Implementation von Programmen werden häufig Fehler gemacht. Insbesondere bei der Implementation von Netzwerkdiensten können diese Fehler schwerwiegende Folgen haben. Bei der überwiegenden Zahl der den Computer-Notfall-Teams gemeldeten Vorfälle handelt es sich um Angriffe, die Implementationsfehler ausbeuten. Ein Programm, das einen besonderen Ruf für vielfältige Fehler erworben hat, ist `sendmail`. Beginnend mit dem ersten "CERT-Advisory" im Jahr 1988 wurden bis Ende 1997 insgesamt 15 "CERT-Advisories" zu `sendmail` herausgegeben.[7]

Beim Ausbeuten von Implementationsfehlern machen die Puffer-Überläufe den Hauptanteil der Angriffe aus. Bei Puffer-Überläufen wird ausgenutzt, daß die Länge der in einen Pufferspeicher eingelesenen Daten nicht überprüft wird. Die Standard C-Bibliotheks-Funktionen zum Einlesen von Daten `strcpy()` und `gets()` überprüfen im Regelfall nicht die Länge der eingelesenen Daten, stattdessen werden über die Länge des Puffers hinausgehende Daten einfach in nachfolgende Speicherbereiche geschrieben. Dieses Überschreiben der nachfolgenden Speicherbereiche wird durch einen Angreifer in zweifacher Weise ausgenutzt. Zunächst wird ausführbarer Code in diese Speicherbereiche geschrieben. Damit dieser ausführbare Code anschließend ausgeführt wird, muß zusätzlich die auf dem "Stack" gespeicherte Rücksprungadresse einer momentan ausgeführten Funktion ersetzt werden. Anstatt an die Stelle des Funktionsaufrufs zurückzukehren wird nach dem Beenden der Funktion dann der vom Angreifer abgelegte Code ausgeführt. [Cowan et al. 98]

Durch das Ausbeuten von Puffer-Überläufen bei Netzwerkdiensten[8] wird es für Angreifer sehr einfach in fremde Systeme einzubrechen,[9] da lediglich so viele Da-

[7] Siehe [CERT 88:01, CERT 90:01, CERT 92:11, CERT 93:15, CERT 93:16, CERT 94:12, CERT 95:05, CERT 95:08, CERT 95:11, CERT 95:13, CERT 96:04, CERT 96:20, CERT 96:24, CERT 96:25, CERT 97:05]

[8] Siehe [CERT 97:04, CERT 97:09, CERT 98:05, CERT 98:09]

[9] Bereits der "Morris-Wurm" nutzte unter anderem auch einen Pufferspeicher-Überlauf im `finger`-Server aus, um in fremde Systeme einzudringen. [Ellermann et al. 91, Kossakowski 92, Reynolds 89, Rochlis et al. 89, Spafford 88, Spafford 90, US GAO 89]

ten an den Server gesendet werden müssen, daß über den Puffer hinausgeschrieben wird und der ausführbare Code sowie die Rücksprungadresse die gewünschte Stelle im Speicher überschreiben. Da hierfür zudem sehr einfache "Hackertools" zur Verfügung stehen, können Puffer-Überläufe auch von Angreifern mit nur sehr geringen Erfahrungen ausgebeutet werden. Da der Angreifer durch einen Puffer-Überlauf sofort in das angegriffene System eindringen kann, haben Puffer-Überläufe inzwischen schon zu einer Reduzierung bei der Anzahl der "Sniffer"-Angriffe geführt, da das Auswerten der von einem "Sniffer" aufgezeichneten Daten viel mühsamer ist und zudem nach der Installation eines "Sniffers" mehrere Stunden oder Tage abgewartet werden muß, bevor interessante Benutzernamen und Paßworte aufgezeichnet worden sind.

Zusätzlich zu der fehlenden Überprüfung der Puffer-Länge werden auch die eingelesenen Daten selbst nicht ausreichend überprüft. Durch die Eingabe unerwarteter Daten können häufig Effekte erzielt werden, die bei der Implementation ursprünglich nicht beabsichtigt waren. So werden zum Beispiel "CGI-Skripte" von Web-Servern verwendet, um einfache Funktionen wie ein Gästebuch zu implementieren. Beim "CGI-phf"-Angriff [CERT 96:06] enthalten die vom Angreifer an den Web-Server geschickten Daten spezielle Zeichen, die dazu führen, daß vom Shell-Skript "phf" zusätzliche vom Angreifer vorgegebene Befehle ausgeführt werden.

Neben den genannten Angriffen, die ein unberechtigtes Eindringen in fremde Systeme ermöglichen, können Implementationsfehler auch für Angriffe auf die Verfügbarkeit ausgenutzt werden. Ziel der Angriffe ist es dann meist einen Dienst oder das gesamte System zu blockieren oder zum "Absturz" zu bringen.

Wie bereits im Abschnitt 2.1.1.2 beim "IP-Spoofing"-Angriff beschrieben, können manche Systeme nur eine geringe Anzahl halb-geöffneter Verbindungen verwalten. Durch das Fälschen von Dateneinheiten des TCP-Verbindungsaufbaus (SYN) können sehr einfach genügend halbgeöffnete Verbindungen erzeugt werden, um das System zu blockieren. Das angegriffene System beantwortet ein gefälschtes "SYN" jeweils mit einem "SYN-ACK" — der zweiten Dateneinheit des TCP-Verbindungsaufbaus — und wartet auf die dritte Dateneinheit des Verbindungsaufbaus. Wenn das System keine Antwort erhält, bleibt diese halbgeöffnete Verbindung bis zu einem "Time-Out" (etwa 3-4 Minuten) bestehen. Wird beim Angriff auch die maximale Anzahl halbgeöffneter Verbindungen erreicht, kann das System während dieser Zeit keine neuen Verbindungen mehr annehmen. Dieser Angriff wird als "SYN-Flooding" bezeichnet. [CERT 96:21, Schuba et al. 97]

Manche Systeme können durch übermäßig große ICMP-"Echo Request"-Nachrichten zum Absturz gebracht werden [CERT 96:26]. Abstürze verursachen bei bestimmten Systemen auch fragmentierte IP-Datagramme, bei denen sich die einzelnen Fragmente überlappen, so daß einzelne Teile des IP-Datagramms gleichzeitig in mehreren Fragmenten enthalten sind (s. "Teardrop"-Angriff [CERT 97:28]).

Beim "Land"-Angriff werden schließlich TCP-SYN-Segmente generiert, bei denen die Absender- und Empfänger-Adresse gleich sind. Auch dies hat den Absturz des angegriffenen Systems zur Folge [CERT 97:28].

Durch moderne Programmiersprachen werden viele Implementationsfehler vermieden. Gänzlich ausschließen lassen sich Programmierfehler jedoch nie. Daher ist es wichtig, vorhandene Fehler möglichst schnell zu erkennen (durch Tests) und zu beheben (durch "Patches"). Angriffe auf Implementationsfehler in Netzwerkdiensten können durch Firewalls abgewehrt werden, indem Zugriffe auf potentiell fehlerhafte interne Systeme vom Firewall verhindert werden und nur Zugriffe auf die wirklich erforderlichen Dienste zugelassen werden.

2.1.3 Konfigurationsfehler

Probleme, die ebenfalls von Angreifern häufig ausgenutzt werden, sind vom Administrator zu großzügig vergebene Zugriffsrechte. So werden Dateisysteme dem gesamten Internet zur Verfügung gestellt, anstatt den Zugriff auf einzelne Rechner zu beschränken. Ähnliche Problem treten bei `.rhosts` und `/etc/hosts.equiv` Dateien auf, die die Zugriffskontrolle für die "rlogin"- und "rsh"-Dienste regeln.

Die Konfiguration von Systemen ist eine komplexe Aufgabe. Konfigurationsfehler lassen sich ähnlich wie Implementationsfehler nicht vollständig ausschließen. Eine regelmäßige Überprüfung des Netzes und der Systeme kann Konfigurationsfehler aufdecken. Angriffe, die Konfigurationsfehler ausnutzen, können über Firewalls abgewehrt werden, indem der Firewall nur Zugriffe auf wenige ausgewählte und sehr sorgfältig administrierte Systeme zuläßt.

2.2 Absicherung durch kryptographische Verfahren

Dieser Abschnitt gibt einen kurzen Überblick über kryptographische Verfahren, ihre Aufgaben und Einsatzbereiche.[10] Abschließend wird dargestellt, was kryptographische Verfahren nicht leisten können und welche zusätzlichen Sicherheitsmechanismen zur Absicherung von Netzen benötigt werden.

"Kryptographie" bezeichnet in direkter Übersetzung der griechischen Worte das "geheime Schreiben". Die Aufgabe kryptographischer Verfahren war entsprechend in erster Linie die Sicherung der Vertraulichkeit. Heute ist der Einsatz

[10] Als weiterführende Literatur zu kryptographischen Verfahren wird [Schneier 96] empfohlen. Die Anwendung von kryptographischen Verfahren zur Absicherung von Kommunikation ist sehr gut in [Kaufman et al. 95] erklärt.

kryptographischer Verfahren zur Sicherung der Authentizität und Integrität ebenso wichtig wie die Sicherung der Vertraulichkeit.[11] Kryptographische Verfahren verwandeln Daten ("Klartext") mit Hilfe eines Schlüssels beim *Verschlüsseln* in "Geheimtext". Der Geheimtext kann anschließend nur mit dem passenden Schlüssel wieder in Klartext verwandelt werden. Wird zum Verschlüsseln und zum Entschlüsseln derselbe Schlüssel verwendet, wird von symmetrischen Verfahren gesprochen. Verfahren, die unterschiedliche Schlüssel zur Ver- und Entschlüsselung verwenden, werden als asymmetrische Verfahren oder "Public-Key"-Verfahren bezeichnet.

Symmetrische Verfahren werden weiter unterteilt in Block-Chiffren ("Block Cipher") und Strom-Chiffren ("Stream Cipher"). Das am weitesten verbreitete Verschlüsselungsverfahren ist DES ("Data Encryption Standard"), eine Block-Chiffre, die mit einem 56 Bit langen Schlüssel 64 Bit lange Datenblöcke verschlüsselt. Symmetrische Verfahren erfordern weniger Rechenaufwand als asymmetrische Verfahren, können somit sehr viel mehr Daten in derselben Zeit ver- bzw. entschlüsseln.

Nachdem die gestiegene Leistungsfähigkeit durchschnittlicher Workstations das etablierte DES mit einer festen Schlüssellänge von nur 56 Bits nicht mehr ausreichend erscheinen lassen [Blaze et al. 96] und in Netzwerken die Berechnungen für das Brechen eines Schlüssels leicht auf viele Prozessoren verteilt werden kann,[12] wird zunehmend "Triple-DES" (3DES) als neuer Standard zur Absicherung der Kommunikation angesehen. Anstelle der einmaligen DES-Verschlüsselung tritt bei 3DES das Verschlüsseln, Entschlüsseln und erneute Verschlüsseln mit unterschiedlichen Schlüsseln. Der Arbeitsaufwand ist gegenüber einer normalen DES-Verschlüsselung etwa dreimal so hoch.

---

[11] Die Unleugbarkeit ("Nonrepudiation") ist eine weitere Eigenschaft, zu denen Sicherung kryptographische Verfahren dienen können. Mit Hilfe kryptographischer Verfahren kann sichergestellt werden, daß ein Sender nicht nachträglich das Senden einer Nachricht leugnen kann. Entsprechendes ist auch für den Nachweis des Empfangens einer Nachricht möglich.

[12] RSA schrieb 1997 einen Wettbewerb (`http://www.rsa.com/rsalabs/97challenge/`) aus, bei dem ein DES-Schlüssel gefunden werden sollte, wobei nur ein Teil des Klartextes in der verschlüsselten Nachricht bekannt war. Die DESCHALL-Gruppe (`http://www.frii.com/~rcv/deschall.htm`) entwickelte hierfür eine verteilte Applikation, bei der jeweils Teile des Schlüsselraums von einzelnen Systemen im Internet zum Entschlüsseln der Nachricht ausprobiert wurden. Bis zu 14.000 Systeme im Internet beteiligten sich zeitweise gleichzeitig an diesem "Brute-Force"-Angriff. Nach etwa 4 Monaten und fast $18 * 10^{15}$ Schlüsseln konnte der korrekte Schlüssel zur Entschlüsselung der Nachricht gefunden werden. (`http://www.frii.com/~rcv/despr4.htm`)
Im Sommer 1998 benötigte ein speziell zum Brechen von DES-Schlüsseln entworfenes System nur noch weniger als drei Tage bis der korrekte Schlüssel gefunden wurde. Dieses System basiert auf etwa 1.500 "Deep-Crack"-Chips, von denen jeder sehr effizient mögliche Schlüssel ausprobieren kann. (s. `http://www.eff.org/descracker/`)

Strom-Chiffren haben noch nicht dieselbe Verbreitung gefunden wie Block-Chiffren. Bei Strom-Chiffren können im Gegensatz zu Block-Chiffren Nachrichten beliebiger Länge verschlüsselt werden, so daß ein Anhängen von Fülldaten zum Erreichen einer vorgegebenen Blocklänge nicht erforderlich ist. Die bekannteste Strom-Chiffre ist RC4 von Ron Rivest.[13]

Asymmetrische Verfahren verwenden zwei unterschiedliche Schlüssel: einen öffentlichen Schlüssel zum Verschlüsseln und einen privaten Schlüssel zum Entschlüsseln. Beide Schlüssel hängen mathematisch voneinander ab, es ist aber trotzdem nicht möglich, den privaten Schlüssel aus dem öffentlichen Schlüssel zu berechnen. Im Gegensatz zu symmetrischen Verschlüsselungsverfahren, bei denen der Schlüssel geheimgehalten werden muß und daher die vertrauliche Übermittlung des Schlüssels an den Kommunikationspartner ein Problem darstellt, kann der öffentliche Schlüssel eines asymmetrischen Verfahrens bekanntgegeben werden. Asymmetrische Verfahren sind sehr viel rechenintensiver als symmetrische Verfahren. Daher ist es in der Regel nicht akzeptabel, ein asymmetrisches Verfahren direkt zur Verschlüsselung von Daten einzusetzen. Stattdessen werden asymmetrische Verfahren praktisch nur für den vertraulichen Austausch von Schlüsseln symmetrischer Verfahren eingesetzt. Dadurch werden die Vorteile asymmetrischer Verfahren (Verwendung öffentlicher Schlüssel) und symmetrischer Verfahren (effizientere Verschlüsselung) miteinander kombiniert. Die beiden wichtigsten asymmetrischen Verfahren sind das "Diffie-Hellman"-Verfahren sowie RSA, das nach seinen drei Entwicklern Rivest, Shamir und Adleman benannt ist.

Prüfsummen-Verfahren sichern die Integrität einer Nachricht, indem mit der Prüfsumme Informationen zur Nachricht hinzugefügt werden, die es dem Empfänger ermöglichen, Veränderungen der Nachricht zu erkennen. Bei *kryptographischen* Prüfsummen-Verfahren ist es darüber hinaus nicht durch einfache Berechnungen möglich, eine zweite Nachricht zu finden, die dieselbe Prüfsumme ergibt. Gebräuchliche kryptographische Prüfsummen-Verfahren sind MD5 [Rivest 92, Kaliski et al. 95] und SHA-1 [NIST 95].

Kryptographische Prüfsummen-Verfahren reichen nicht allein zur Abwehr von Angriffen auf die Integrität einer übertragenen Nachricht aus, da eine Angreifer nach der Manipulation der Nachricht zusätzlich auch die Prüfsumme ersetzen kann. Um eine Authentisierung des Senders zu ermöglichen und auch eine Manipulation der Nachricht mitsamt der Prüfsumme zu verhindern, wird vor der Berechnung der Prüfsumme ein Schlüssel zur Nachricht hinzugefügt, der nur dem Sender und dem Empfänger bekannt ist. Dieser Schlüssel wird

[13] Ron Rivest ist einer der Entwickler des RSA-Verfahrens (s. u.) und Gründer des Unternehmens "RSA Data Security, Inc.".

nach der Berechnung der Prüfsumme wieder entfernt, so daß der Schlüssel nicht zusammen mit der Nachricht übertragen wird. Erst der Empfänger fügt den Schlüssel zur Überprüfung der Prüfsumme wieder zur Nachricht hinzu. Bei diesem Verfahren kann der Empfänger sicher sein, daß die Prüfsumme von einem Sender erzeugt wurde, der in Besitz des Schlüssels ist. Dieses Verfahren wird z.B. als "Keyed MD5" [Metzger et al. 95a, Krawczyk et al. 97] bzw. "Keyed SHA-1" [Metzger et al. 95b] beim "Authentication Header" von IPSEC (s. u.) zur Sicherung der Authentizität und Integrität von IP-Dateneinheiten verwendet.

Einsatz kryptographischer Verfahren in Protokollen: Kryptographische Verfahren werden zur Sicherung von Vertraulichkeit, Integrität und Authentizität von Daten bei der Speicherung oder bei der Übertragung eingesetzt. Für den Einsatz zur Absicherung von Daten bei der Übertragung sind bereits sehr viele unterschiedliche Protokolle verfügbar. Einen Überblick über die derzeit verfügbaren kryptographischen Protokolle gibt Abbildung 2.3. Der Einsatz zur Sicherung gespeicherter Daten wird im folgenden nicht weitergehend betrachtet.

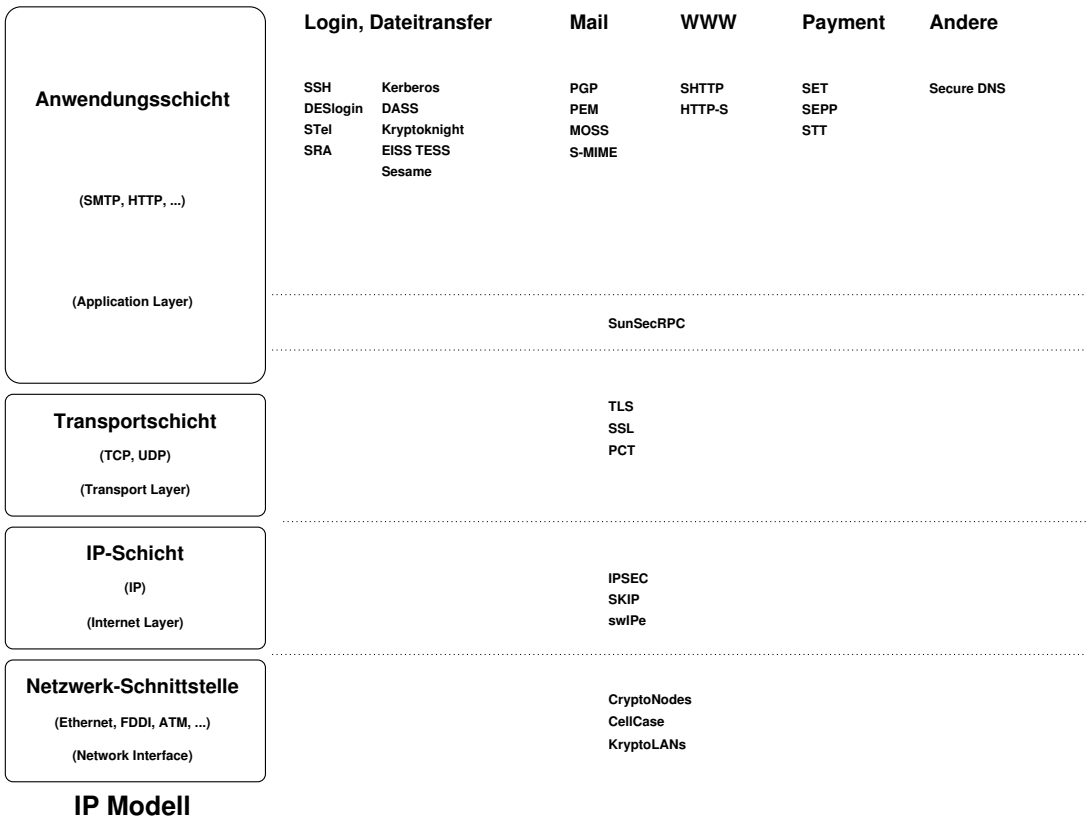


Abbildung 2.3: Einsatz kryptographischer Verfahren in Protokollen

Wie Abbildung 2.3 zeigt, sind die kryptographischen Protokolle unterschiedlichen Schichten des Kommunikationssystems zugeordnet. Abhängig von der Schicht, auf der das jeweilige Protokoll eingesetzt wird, werden unterschiedliche Eigenschaften erreicht, so daß jedes Protokoll seinen eigenen Einsatzbereich hat. Kryptographische Protokolle auf der Anwendungsschicht sind eng mit der jeweiligen Anwendung verbunden, während Protokolle auf den unteren Schichten zwar die Daten unterschiedlicher Anwendungen absichern können, jedoch dadurch auch nicht so exakt die Anforderungen der jeweiligen Anwendung erfüllen können. Die wichtigsten kryptographischen Protokolle werden im folgenden kurz aufgeführt. Eine ausführlicher Dokumentation findet sich in [Benecke et al. 98c] und in [Kaufman et al. 95].

SSH ("Secure SHell") ermöglicht ein sicheres "Login" und eine sichere Dateiübertragung [Ylönen 96a] und soll die als unsicher bekannten Dienste "rlogin" ("Remote Login") [Kantor 91], "rsh" ("Remote Shell") und "rcp" ("Remote Copy") ersetzen. Nach einer Authentisierung mit RSA-Schlüsseln sichert SSH die Vertraulichkeit und Integrität der Verbindung.

"Kerberos" ist ein Protokoll, das auf einem zentralen Schlüsselverteiler ("Key Distribution Center", KDC) basiert und ebenfalls "Login" und die Übertragung von Dateien ermöglicht. "Kerberos" kommt dabei ohne asymmetrische Verschlüsselungsverfahren aus. [Steiner et al. 88, Kohl et al. 93, Kaufman et al. 95, Bellovin et al. 91]

PGP und S-MIME dienen zur Sicherung der Authentizität, Integrität und Vertraulichkeit von E-Mail. Bei beiden Verfahren handelt es sich strenggenommen nicht um Protokolle, da sie außerhalb des Protokollstapels lediglich ein Format für das Verschlüsseln von Daten und das Anbringen von Signaturen definieren. Die auf diese Weise gesicherten Daten können anschließend in herkömmlicher E-Mail übertragen werden. [Garfinkel 95, Atkins et al. 96, Dusse et al. 98a, Dusse et al. 98b]

SET wird gemeinsam von Visa und Mastercard für sichere "Electronic Commerce"-Transaktionen entwickelt.[14]

"Secure DNS" ergänzt DNS Einträge ("Resource Records", RR) um kryptographische Informationen, damit Clients die Authentizität der von DNS-Servern gelieferten Informationen überprüfen können. Eine Sicherung der Vertraulichkeit ist nicht vorgesehen und auch nicht erforderlich, da DNS-Informationen öffentlich sind. [Eastlake et al. 97, Gilmore 97]

TLS und SSL sind zwei Protokolle, die auf Transportebene angesiedelt sind. Sie setzen auf TCP auf und sichern nach einer Authentisierung des Servers beim

[14] [Mastercard et al. 97a, Mastercard et al. 97b, Mastercard et al. 97c]

Client die Integrität und Vertraulichkeit der Verbindung. Eine Authentisierung des Clients beim Server ist optional möglich, wird von den Protokollen aber nicht gefordert. SSL ("Secure Socket Layer") [Freier et al. 96, Hudson et al. 95, Wagner et al. 96] ist zur Absicherung von HTTP-Transaktionen entwickelt worden und hat auch heute noch hier den Haupteinsatzbereich. TLS ("Transport Layer Security") [Dierks et al. 97] ist eine Weiterentwicklung von SSL und soll als Sicherheitsprotokoll auf der Transportebene zur Absicherung unterschiedlicher Protokolle dienen.

IPSEC ergänzt das Internet Protokoll (IP) um den "Authentication Header" (AH) [Atkinson 95b], der die Integrität und Authentizität eines IP-Datagramms sichert, und um die "Encapsulated Security Payload" (ESP) [Atkinson 95c], die eine Verschlüsselung der in einem IP-Datagramm enthalten Daten ermöglicht. Zusätzlich wird mit ISAKMP/Oakley [Orman 98, Harkins et al. 98, Maughan et al. 98] ein Schlüsselverwaltungsprotokoll für IPSEC definiert, mit dem die für die symmetrischen Verfahren benötigten Sitzungsschlüssel zwischen Kommunikationspartnern vereinbart werden. [Atkinson 95a]

Grenzen kryptographischer Verfahren: Kryptographische Verfahren leisten durch die Sicherung von Vertraulichkeit, Authentizität und Integrität einen wichtigen Betrag in einem Sicherheitskonzept. Jedoch können durch kryptographische Verfahren nicht alle der in Abschnitt 2.1 diskutierten Schwachstellen von Netzen abgesichert werden.

Schwächen bei der Gewährleistung der Verfügbarkeit: Es ist zwar möglich, einige der auf "IP-Spoofing" basierende "Denial of Service"-Angriffe durch den Einsatz von IPSEC zu erkennen und abzuwehren, aber auf der anderen Seite können kryptographische Verfahren selbst das Ziel von "Denial of Service"-Angriffen werden. Kryptographische Verfahren sind sehr rechenaufwendig. Dies gilt insbesondere für die häufig zur Authentisierung eingesetzten RSA-basierten Verfahren. Dadurch können selbst Zugriffe, die die Authentisierung nicht passieren, erhebliche Ressourcen binden.

Implementationsfehler: Der Einsatz kryptographischer Verfahren kann Implementationsfehler nicht beseitigen. Ein Angreifer erhält beim Einsatz kryptographischer Verfahren lediglich eine *sichere* Verbindung zu einem *unsicheren* Server und kann somit im Server vorhandene Fehler weiterhin ausbeuten. Zusätzlich muß auch bei kryptographischen Verfahren selbst mit Implementationsfehlern gerechnet werden.

Konfigurationsfehler: Ähnlich wie Implementationsfehler können auch Konfigurationsfehler nicht durch kryptographische Verfahren beseitigt werden.

Vielmehr erschweren die kryptographischen Verfahren die Konfiguration noch zusätzlich, da bereits die zugrundeliegenden Konzepte schwer zu verstehen sind und bestehende Implementationen meist unzählige Parameter anbieten, deren Auswirkungen einem unerfahrenem Administrator häufig verborgen bleiben. Der zunehmende Einsatz kryptographischer Verfahren zur Absicherung der Kommunikation wird Konfigurationsfehler somit zukünftig noch fördern, statt sie zu reduzieren.

Zur Absicherung dieser aufgeführten Probleme sind weitere Sicherheitsmechanismen erforderlich. In Ergänzung zu kryptographischen Verfahren werden in erster Linie Zugriffskontrollmechanismen benötigt. Teilweise können Zugriffskontrollmechanismen direkt in kryptographische Verfahren integriert werden, wie z. B. IPSEC und "Kerberos" zeigen. Gegen Angriffe auf Implementations- und Konfigurationsfehler ist jedoch eine möglichst frühzeitige Ablehnung unberechtigter Zugriffe bereits durch einen Firewall am erfolgversprechendsten. Ein Firewall verhindert Angriffe auf Implementations- und Konfigurationsfehler, da die Zugriffe von Angreifern gar nicht erst zu möglicherweise fehlerhaften Systemen durchgelassen werden. Der Firewall selbst muß natürlich besonders gründlich auf Implementationsfehler hin untersucht werden und auch besonders sorgfältig administriert werden, damit der Firewall nicht selbst angreifbare Implementations- oder Konfigurationsfehler enthält. Die Kombination von Firewalls mit kryptographischen Verfahren ist derzeit noch ein Forschungsbereich [Benecke et al. 98c].

2.3 Firewalls

Firewalls sind Zugriffskontrollmechanismen in Netzen. Sie werden eingesetzt, um eine Trennung der Netze nach Sicherheitsgesichtspunkten zu erreichen. Üblicherweise wird durch einen Firewall ein internes, sicherheitskritisches Netz vor Angriffen aus einem externen Netz geschützt.[15] Der Firewall setzt vorher spezifizierte Sicherheitsrichtlinien ("Security Policy") durch. Dem externen Netz gegenüber stellt der Firewall nur eine kleine Anzahl gut gesicherter und streng überwachter Dienste zur Verfügung. Damit der Firewall die gesamte Kommunikation zwischen internem und externem Netz überwachen kann, muß durch organisatorische Maßnahmen gewährleistet werden, daß der Firewall den alleinigen Übergang zwischen beiden Netzen bildet. Die Kommunikation zwischen Systemen innerhalb des internen Netzes sowie zwischen Systemen im externen Netz wird durch den Firewall nicht kontrolliert.

[15] Für das durch den Firewall geschützte Netz wird im folgenden die Bezeichnung "*internes Netz*" verwendet. Das Netz auf der anderen Seite des Firewalls wird als "*externes Netz*" bezeichnet.

Firewall: "*Ein Firewall ist eine Schwelle zwischen zwei Netzen, die überwunden werden muß, um Systeme im jeweils anderen Netz zu erreichen. Es wird dafür gesorgt, daß jede Kommunikation zwischen den beiden Netzen über den Firewall geführt werden muß. Auf dem Firewall sorgen Zugriffskontrolle und Audit dafür, daß das Prinzip der geringsten Berechtigung durchgesetzt wird und potentielle Angriffe schnellstmöglich erkannt werden.*" [Ellermann 93]

Der Begriff "Firewall" (Brandschutzmauer) stammt aus der Architektur und wird zum ersten Mal 1975 von [Saltzer et al. 76] in "The Protection of Information in Computer Systems" im Zusammenhang mit Computer-Sicherheitsmechanismen verwendet. Erst 1989/90 werden, ohne den Begriff "Firewall" zu verwenden, die auch heute noch verwendeten Firewall-Konzepte erstmals beschrieben. 1989 stellt [Mogul 89] eine Packet Screen vor, und in [Cheswick 90] wird ein Gateway-Firewall mit einem Proxy beschrieben. Diverse Autoren greifen 1992 diese Konzepte auf und prägen hierfür den Begriff "Firewall".[16] Zunächst wurden Firewalls primär zur sicheren Anbindung von Unternehmensnetzen an das (öffentliche) Internet eingesetzt.[17] Neben diesen "Internet-Firewalls" ist heute die Absicherung besonders sensitiver Abteilungen innerhalb eines Unternehmensnetzes ein wichtiger Einsatzbereich von Firewalls ("Intranet-Firewalls"). Diese beiden Einsatzbereiche unterscheiden sich durch:

Kommunikationsanforderungen: Zwischen den verschiedenen Teilen eines Unternehmensnetzes bestehen in der Regel engere Kommunikationsbeziehungen als mit dem Internet. So ist meist der Zugriff vom abgesicherten Netz auf Datenbanken im Unternehmensnetz notwendig.

Da das durch einen Intranet-Firewall abgetrennte Netz oft nur wenige Systeme umfaßt, ist der Aufwand zur Absicherung selten genutzter Dienste kaum zu rechtfertigen. Die Liste der zu unterstützenden Protokolle wird sich daher für Internet- und Intranet-Firewalls unterscheiden.

Sicherheitsanforderungen: Intranet-Firewalls sind einer höheren Bedrohung ausgesetzt, da die zu schützenden Systeme besonders kritische Informationen enthalten. Auch der Kreis der potentiellen Angreifer unterscheidet sich, da primär mit Angriffen durch firmeninterne Mitarbeiter gerechnet wird. Externe Angreifer werden bereits durch den Internet-Firewall erkannt und aufgehalten.

---

[16] Siehe [Bellovin 92, Carl-Mitchell et al. 92, Chapman 92, Koblas et al. 92, Ranum 92a, Ranum 92b]

[17] Praktisch alle Firewalls werden derzeit für TCP/IP-Protokolle eingesetzt. Eine Anwendung des Firewall-Konzeptes ist prinzipiell aber auch bei anderen Protokollen (ATM, DECNet, ISO-OSI, ...) möglich. (siehe z. B. [Berge et al. 96])

Performanzanforderungen: Hochgeschwindigkeitsnetze werden innerhalb von Unternehmen früher eingesetzt als bei der Internet-Anbindung. Intranet-Firewalls bilden daher früher einen Engpaß.

Für den Aufbau eines Firewalls wird eine Reihe von Komponenten benötigt, die exakt aufeinander abgestimmt sein müssen. Die wichtigsten Firewall-Komponenten — Packet Screen, Proxy-Server und Anwendungs-Gateway — sollen im folgenden Abschnitt 2.3.1 vorgestellt werden. Die wichtigsten Firewall-Architekturen, die sich aus den vorgestellten Firewall-Komponenten aufbauen lassen, werden im Abschnitt 2.3.2 diskutiert. Auf die Vor- und Nachteile jeder Firewall-Architektur wird dabei explizit hingewiesen.

2.3.1 Firewall-Komponenten

Firewall-Komponenten sind Mechanismen, aus denen Firewalls aufgebaut werden. Die Mechanismen werden auf unterschiedlichen Ebenen realisiert. Die erreichte Granularität der Zugriffskontrolle hängt von den auf der jeweiligen Ebene zur Verfügung stehenden Informationen ab.

2.3.1.1 Packet Screen

Packet Screens führen eine Zugriffskontrolle auf IP-Datagrammen durch. Sie gewähren bzw. unterbinden die Kommunikation zwischen mindestens zwei Netzen, indem sie nur die Datagramme weiterleiten, deren Protokoll-Header auf Vermittlungs- und Transportschicht den vom Verwalter konfigurierten Filterregeln genügen.[18] Einen Überblick über Header-Informationen, die von einer Packet Screen ausgewertet werden können, bietet Abbildung 2.4.

Über die Sender- und Empfänger-Adressen des IP-Headers ist eine Beschränkung der Endsysteme möglich, die miteinander kommunizieren dürfen. Dadurch kann zum Beispiel der Zugriff auf besonders kritische Systeme verwehrt werden. Das Protokoll-Feld des IP-Headers gibt an, zu welchem Protokoll die enthaltene Dateneinheit gehört (TCP, UDP, ICMP oder andere). Fragment-Offset und Datagrammlänge — in Abbildung 2.4 schwach gedruckt — werden nur von einigen Packet Screens überprüft, sind aber zur Erkennung von Fragmentierungsangrif-

[18] Häufig werden die Begriffe Paket-Filter ("Packet Filter") und Packet Screen fälschlicherweise synonym verwendet. Ein Paket-Filter ist ein Filter in Endsystemen, mit dem Applikationen wie `tcpdump` empfangene Pakete nach bestimmten Merkmalen selektieren können.

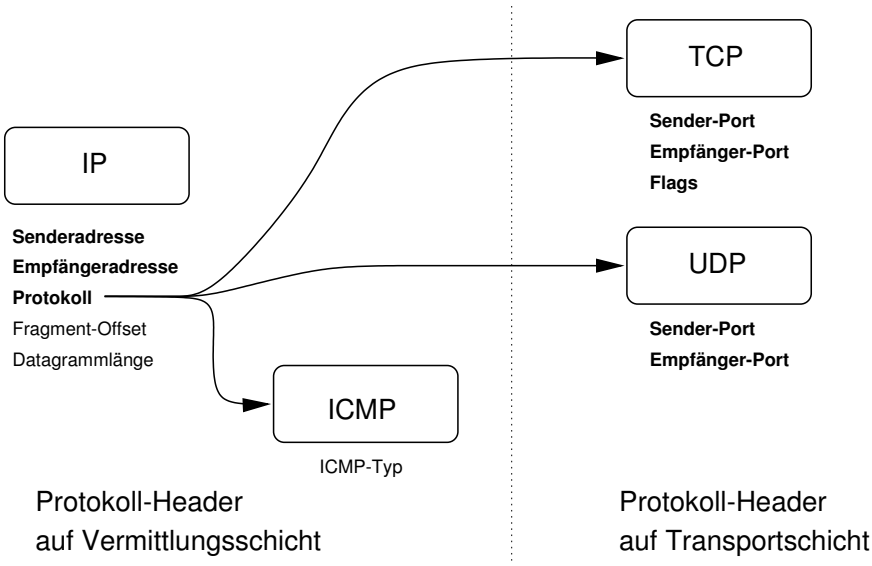


Abbildung 2.4: Von Packet Screens ausgewertete Informationen

fen notwendig.[19] Über Sender- und Empfänger-Ports der Protokoll-Header auf der Transportschicht können die zugegriffenen Dienste identifiziert werden. Die Identifikation der Dienste ist jedoch nicht besonders zuverlässig, da durch Manipulation der Endsysteme auch nicht standardisierte Ports verwendet werden können und zudem für RPC-basierte Protokolle keine feste Zuordnung der Dienste zu bestimmten Portnummern existiert. Über die Auswertung des SYN-Flags im TCP-Header kann zwischen der ersten Dateneinheit einer neuen Verbindung und den nachfolgenden Dateneinheiten unterschieden werden. Die Auswertung des SYN-Flags ermöglicht es der Packet Screen, die Richtung eines Verbindungsaufbaus zu erkennen. Da die Übertragung einer Dateneinheit mit SYN-Flag für den Aufbau einer TCP-Verbindung erforderlich ist, kann unter der Annahme, daß die Dateneinheit mit SYN-Flag von der Packet Screen bereits überprüft wurde, die Prüfung der nachfolgenden Dateneinheiten (ohne SYN-Flag) entsprechend einfacher ausfallen. Die meisten Packet Screens unterstützen keine Filterung von ICMP-Dateneinheiten anhand des ICMP-Typs ("Echo", "Destination Unreachable", "Source-Quench", "Timestamp", "Redirect", ...). Meist kann nur über die Auswertung des Protokoll-Feldes im IP-Header die Übertragung von ICMP-Nachrichten insgesamt unterbunden werden. Dies ist jedoch wenig emp-

[19] Bei fragmentierten Dateneinheiten sind die Transport-Header nur im ersten Fragment enthalten, so daß Packet Screens folgende Fragmente (Offset > 0) ohne vollständige Überprüfung weiterleiten können. Durch nachfolgende Fragmente mit einem kleinen Offset können beim Empfänger Teile des im ersten Fragment enthaltenen Transport-Headers überschrieben werden.

fehlenswert, da ICMP wichtige Kontrollinformationen für die ordnungsgemäße Verarbeitung von IP-Dateneinheiten transportiert.[20]

Dynamische Packet Screens: Im Gegensatz zu den bisher beschriebenen "*Statischen Packet Screens*", bei denen jede eingehende Dateneinheit nach einem festgelegten Satz von Filterregeln unabhängig von bereits verarbeiteten Dateneinheiten ausgewertet wird, kann bei "*Dynamischen Packet Screens*"[21] die Filterentscheidung vom Kontext bereits empfangener Dateneinheiten abhängen.

Die Möglichkeit, Filterentscheidungen anhand des Kontextes bereits empfangener Dateneinheiten durchführen zu können, hat im wesentlichen zwei Vorteile:

- Da bei UDP kein Verbindungsaufbau durchgeführt wird, kann die Richtung des Zugriffs nicht wie bei TCP über das SYN-Flag festgestellt werden. Ohne Kenntnis vorangegangener Dateneinheiten kann nicht entschieden werden, ob es sich bei einem empfangenen UDP-Datagramm um eine Reaktion (Antwort) oder um einen neuen Zugriff handelt. Bei einer "Dynamischen Packet Screen" wird eine Filterregel ergänzt, die Antworten auf übertragene UDP-Datagramme ebenfalls passieren läßt. Nach längerer Inaktivität dieser Kommunikationsbeziehung wird die Filterregel wieder entfernt.

- FTP stellt für Packet Screens ein Problem dar, weil Kontroll- und Datenverbindung von unterschiedlicher Richtung aus geöffnet werden. "Dynamische Packet Screens" können das Zulassen einer FTP-Datenverbindung davon abhängig machen, ob gleichzeitig auch eine Kontrollverbindung zwischen den beiden Systemen existiert.

Granularität: Die Granularität der Zugriffskontrolle ist bei Packet Screens auf die im Protokoll-Header enthaltenen Informationen beschränkt. Somit kann der Zugriff nur anhand der **IP-Adressen** der kommunizierenden Systeme und des zugegriffenen Dienstes (**Port-Nummer**) kontrolliert werden. Eine Authentisierung von Benutzern[22] oder eine Kontrolle der übertragenen Daten ist nicht möglich.

---

[20] Über ICMP kann einem Sender mitgeteilt werden, daß der Empfänger nicht erreichbar ist. Ohne ICMP kann der Sender bei verbindungsorientierten Protokollen erst nach einem mehrere Minuten betragenden "Time-Out" erkennen, daß seine Dateneinheiten nicht bestätigt werden. Bei verbindungslosen Protokollen ist keine Erkennung möglich.

[21] Einige Hersteller verwenden auch die Bezeichnung "Stateful Packetfilter".

[22] Eine Authentisierung von Benutzern wäre zukünftig mit Hilfe von IPSEC denkbar [Ellermann 96], die Verfügbarkeit von entsprechenden Packet Screen Implementationen läßt sich heute noch nicht absehen.

Vorteile einer Packet Screen:

Transparenz: Die Überprüfung von Dateneinheiten kann in der Packet Screen ohne Interaktion mit dem Benutzer erfolgen. Es sind auch keine Konfigurationsänderungen auf den Endsystemen notwendig.

Performanz: Der Filter-Aufwand der Packet Screen ist relativ gering, so daß eine gute Performanz erreicht wird. Die Performanzauswirkungen der Packet Screen werden vom Benutzer nicht bemerkt.[23]

Geringer Installationsaufwand: Die Packet Screen Funktionalität ist in heutigen Routern meist schon vorhanden, so daß nur die Definition geeigneter Filterregeln notwendig ist. Die Konfiguration erfordert nur einen geringen Zeitaufwand.

Kostengünstig: Da Packet Screens meist im Router schon verfügbar sind, ist dank des geringen Zeitaufwands zur Konfiguration der Filterregeln, der Einsatz einer Packet Screen relativ kostengünstig. Wenn kein vorhandener Router als Packet Screen verwendet werden kann, bietet frei verfügbare Packet Screen Software (z. B. "Drawbridge"[24] oder "IP-Filter"[25]), die auf einfacher PC-Hardware eingesetzt werden kann, eine ebenfalls kostengünstige Alternative. Kommerzielle Lösungen wie die dynamische Packet Screen "Firewall-1" sind deutlich teurer.

Nachteile einer Packet Screen:

Granularität der Zugriffskontrolle: Eine Zugriffskontrolle kann bei Packet Screens nur anhand von Informationen im Protokoll-Header erfolgen.

Schlechtes Audit: Die in gebräuchlichen Routern enthaltenen Packet Screens können nur sehr rudimentäre Audit-Daten aufzeichnen. Über das Internet verbreitete Packet Screen Software ("Drawbridge" und "IP-Filter") sowie kommerzielle Lösungen (z. B. "Firewall-1") erlauben eine detailliertere Aufzeichnung von Audit-Daten, jedoch stehen konzeptbedingt bei Packet Screens nur geringe Informationen für das Audit zur Verfügung. (vgl. "Granularität")

---

[23] In Hochgeschwindigkeitsnetzen kann eine Packet Screen jedoch zum Engpaß werden, wie die Analysen im Abschnitt 4.3 zeigen.

[24] `ftp://ftp.cert.dfn.de/pub/firewalls/software/drawbridge/`

[25] `ftp://coombs.anu.edu.au/pub/net/ip-filter/`

Kontrolle der freigegebenen Ports: Eine Packet Screen kann nicht kontrollieren, ob an einer freigegebenen Portnummer tatsächlich der vorgesehene Dienst angeboten wird. Durch einfache Manipulationen an den Endsystemen kann eine Packet Screen leicht umgangen werden.

Korrektheit erstellter Filterregeln: Die korrekte Konfiguration der Filterregeln erfordert größte Sorgfalt, da bereits kleine Fehler zu einem nicht erwünschten Verhalten führen können. Bisher bieten nur wenige Packet Screens dem Verwalter eine Unterstützung bei der Konfiguration an. Wünschenswert wäre auch eine automatische Überprüfung neu erstellter Filterregeln.

2.3.1.2 Proxy-Server

Ein Proxy-Server — kurz Proxy (Stellvertreter) — ist ein Prozeß auf Anwendungsebene, der Anfragen für einen Client an einen Server stellt und die Antworten des Servers an den Client weiterleitet. Bei dem Einsatz von Proxies in Firewall-Konzepten wird durch zusätzliche Maßnahmen die direkte Kommunikation zwischen Client und Server unterbunden, so daß die Kommunikation nur über einen Proxy möglich ist. Da der Proxy Zugriff auf die Anfragen des Clients und auf die Antworten des Servers erhält, wird eine sehr gute Zugriffskontrolle möglich. Der konzeptionelle Aufbau eines Proxies ist in Abbildung 2.5 wiedergegeben.

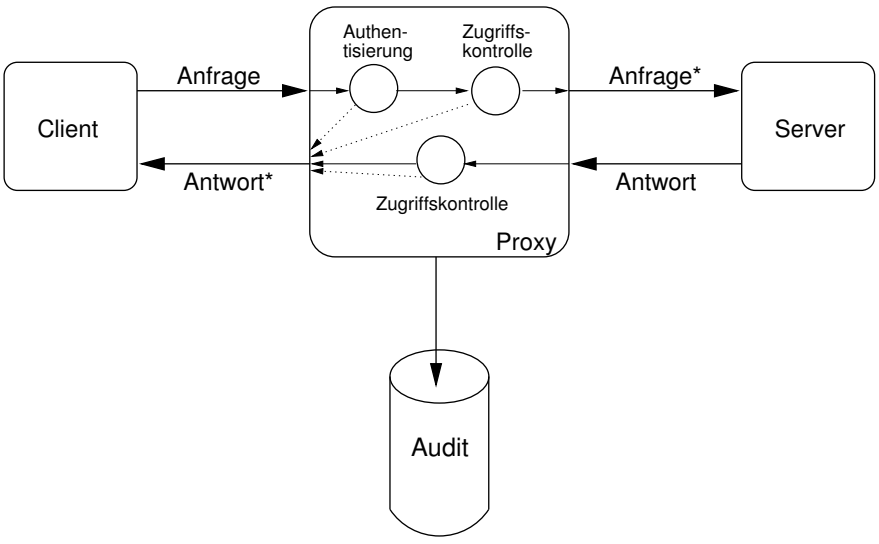


Abbildung 2.5: Konzeptionelle Sicht auf einen Proxy

Eine Anfrage eines Clients muß vom Proxy zunächst authentisiert werden. Hierfür werden kryptographische Verfahren, Einmal-Paßworte, aber auch noch unsiche-

re Verfahren wie die Überprüfung der IP-Absender-Adresse eingesetzt. Je nach Sicherheitsanforderungen kann die Authentisierung einmalig beim Verbindungsaufbau oder wiederholt für jede Anfrage durchgeführt werden. Bei der anschließenden Zugriffskontrolle wird überprüft, ob der Zugriff des Clients den spezifizierten Sicherheitsrichtlinien entspricht. Der Zugriff kann für bestimmte Server, Dienste oder auch bestimmte Uhrzeiten eingeschränkt sein. Es ist auch möglich, den Zugriff auf ausgewählte Bereiche eines Servers zu beschränken. Die vom Proxy an den Server gesendete Anfrage (Anfrage*) kann sich daher durchaus von der ursprünglichen Anfrage des Clients unterscheiden. Die vom Server zurückgesendeten Daten werden vom Proxy ebenfalls zunächst überprüft und gegebenenfalls modifiziert (Antwort*), bevor sie an den Client weitergeleitet werden. Bei der Überprüfung der Antwort kommen hauptsächlich "Scanner" für ausführbare Inhalte (Java-Applets, Java-Script, Makros usw.) zum Einsatz. Alle drei Mechanismen können Zugriffe verweigern und sollten in einem derartigen Falle dem Client dieses über eine Fehlermeldung mitteilen (gestrichelte Pfeile). Das Audit protokolliert normale und verweigerte Zugriffe sowie gegebenenfalls weitere Vorkommnisse. Abhängig von den Sicherheitsrichtlinien kann auf die Implementierung einzelner Mechanismen auch verzichtet werden.

Granularität: Mit Proxies ist eine sehr detaillierte Zugriffskontrolle möglich. Die Kontrolle kann abhängig von **IP-Adressen**, vom **Dienst**, vom **Benutzer**, von **zeitlichen Beschränkungen**, oder vom **Kontext** des Zugriffs erfolgen. Insbesondere stehen der Zugriffskontrolle alle zwischen Client und Server ausgetauschten Daten zur Verfügung.

Vorteile eines Proxies:

Feine Granularität: Größter Vorteil von Proxies ist die **detaillierte Zugriffskontrolle**, durch die auch ein sehr gutes **Audit** möglich wird (s. o.).

Authentisierung: Es ist eine einmalige Benutzerauthentisierung beim Verbindungsaufbau oder eine wiederholte Authentisierung für jede Anfrage möglich.

Nachteile eines Proxies:

Transparenz: Zugriffe über einen Proxy können für den Client nicht transparent erfolgen. Abhängig von der Implementation des Clients kann gegebenenfalls die fehlende Transparenz gegenüber dem Benutzer verborgen werden.

Dafür ist jedoch eine auf den Proxy abgestimmte Implementation und Konfiguration des Clients erforderlich, was einen hohen Installations- und Verwaltungsaufwand nach sich zieht. Authentisierungsverfahren machen oft eine Interaktion mit dem Benutzer erforderlich, so daß eine transparente Lösung hier nicht möglich ist.

Performanz: Die Verarbeitung von Anfragen auf Anwendungsebene erfordert eine höhere Rechenleistung als die Überprüfung der Protokoll-Header in einer Packet Screen. Durch aufwendige Authentisierungs- und Zugriffskontrollmechanismen kann der Proxy schnell zum Engpaß werden.

Installationsaufwand: Proxies können nicht allein eingesetzt werden. Aufbau und Konfiguration ebenfalls benötigter Komponenten wie Packet Screens und Bastionen (s. Abschnitt 2.3.2) stellen einen hohen Aufwand dar.

2.3.1.3 Anwendungs-Gateway

Bei Protokollen, die im Stapelbetrieb ("Batch-Mode") verarbeitet werden, wie zum Beispiel SMTP (E-Mail) und NNTP (Netnews), ist es vorteilhaft, Zugriffe nicht über einen Proxy an einen Server weiterzuleiten, sondern auf dem Firewall bereits wesentliche Teile der Serverfunktionalität zu realisieren.

Die Bezeichnungen Proxy und Anwendungs-Gateway werden häufig synonym verwendet. Anwendungs-Gateways unterscheiden sich von Proxies durch eine höhere Komplexität. Während sich Proxies meist auf eine einfache Authentisierung beschränken und anschließend lediglich Daten zwischen den beiden Verbindungen kopieren, enthalten Anwendungs-Gateways für die jeweilige Applikation spezialisierten Code, der bereits eine Vorverarbeitung durchführt oder einen Teildienst der Anwendung erbringt. Da auch bei Proxies Zugriffskontrollmechanismen eingesetzt werden können, die speziell auf die unterstützte Anwendung ausgerichtet sind, ist die Grenze zwischen den Anwendungs-Gateways und Proxies fließend.

Granularität: Die Granularität der Zugriffskontrolle und des Audits unterscheiden sich nicht von dem eines Proxies.

Vorteile von Anwendungs-Gateways: Zusätzlich zu den bereits vorgestellten Vorteilen von Proxies haben Anwendungs-Gateways folgende Vorteile:

Verdecken der internen Struktur: Durch Erbringen wesentlicher Serverfunktionen kann die interne Struktur des durch den Firewall geschützten Netzes verdeckt werden.

Verzögerungszeiten weniger relevant: Die stapelorientierte Verarbeitung erlaubt aufwendigere Zugriffskontrollverfahren (z. B. das "Scannen" nach Viren in E-Mails), da höhere Verzögerungen als bei interaktiven Protokollen akzeptabel sind.

Nachteile von Anwendungs-Gateways: Zusätzlich zu den bereits vorgestellten Nachteilen von Proxies haben Anwendungs-Gateways folgende Nachteile:

Keine Benutzerinteraktion zur Authentisierung: Stapelorientierte Anwendungen schränken die Liste der einsetzbaren Authentisierungsverfahren ein, da eine Benutzerinteraktion zur Authentisierung nicht möglich ist.

2.3.1.4 Weitere Firewall-Komponenten

Audit: Die von den Zugriffskontrollkomponenten gemeldeten Ereignisse müssen aufgezeichnet und ausgewertet werden. Um Angriffe schnell erkennen zu können, ist eine maschinelle Auswertung der aufgezeichneten Daten notwendig.

Verschlüsselung: In zunehmendem Maße werden Verschlüsselungsverfahren in Firewall-Konzepte integriert. Neben den bereits eingesetzten "Virtual Private Networks" (VPNs) werden zukünftig auch in Proxies und Anwendungs-Gateways kryptographische Verfahren zur Verschlüsselung der übertragenen Daten oder zur Authentisierung angewendet werden.

NAT: Über die "Network Address Translation" (NAT) [Egevang et al. 94] können Packet Screens nicht-öffentliche auf öffentliche IP-Adressen abbilden. Systeme mit nicht-öffentlichen IP-Adressen können so auf Dienste im Internet zugreifen. Zudem wird durch NAT die Struktur des internen Netzes verdeckt.

Systemmanagement: Die Konfiguration und Wartung eines Firewalls stellt eine sehr komplexe Aufgabe dar, die einer Unterstützung durch geeignete Werkzeuge bedarf. Dies können zum Beispiel grafische Benutzeroberflächen für die Erstellung von Filterregeln oder Werkzeuge zur Auswertung von Audit-Daten sein.

2.3.2 Firewall-Architekturen

Firewall-Architekturen werden aus Firewall-Komponenten aufgebaut. Aus der Vielzahl möglicher Kombinationen werden im folgenden die wichtigsten Architekturen mit ihren Vor- und Nachteilen kurz vorgestellt.

2.3.2.1 Packet Screens

Die Konfiguration einer Packet Screen stellt die einfachste Firewall-Architektur dar. Trotz der bereits diskutierten Einschränkungen (s. Abschnitt 2.3.1.1) bei der Granularität der Zugriffskontrolle und beim Audit kann die Konfiguration einer Packet Screen ausreichen, wenn die erkannten Defizite durch zusätzliche Maßnahmen kompensiert werden (s. [Ellermann 95]).

Da die Möglichkeit, Filterregeln zu definieren, bereits in allen gängigen Routern enthalten ist und ein Router zudem zur Anbindung eines Netzes an das Internet oder an ein größeres Firmennetz ohnehin benötigt wird, ist der Einsatz einer Packet Screen eine kostengünstige Lösung. Die konzeptionelle Sicht einer Firewall-Architektur auf Basis einer Packet Screen ist in Abbildung 2.6 wiedergegeben.

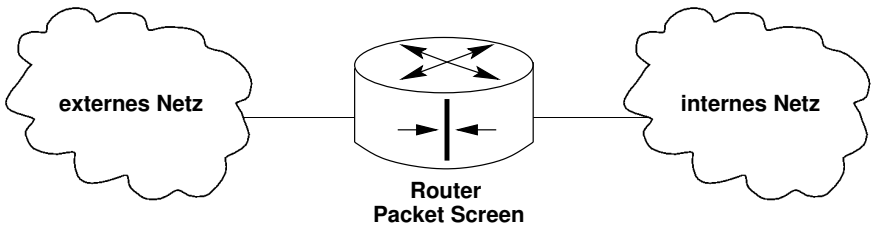


Abbildung 2.6: Packet Screen: konzeptionelle Sicht

Zur Erstellung der Filterregeln können zwei unterschiedliche Ansätze gewählt werden:

Verbotsregeln: Über die Konfiguration von Verbotsregeln für als gefährlich eingestufte Protokolle können Angriffe an der Packet Screen gestoppt werden. Da alle anderen Dienste von diesen Regeln nicht betroffen sind, bleiben bei diesem Ansatz viele Dienste erreichbar. Für Benutzer hat dies den Vorteil, daß ihre Kommunikation nur selten durch die Filterregeln unterbunden wird. Allerdings bleiben so auch viele Dienste für Angriffsversuche geöffnet, so daß dieser Ansatz nur als "Ad-hoc" Maßnahme bei konkreten Angriffen auf einzelne Dienste gewählt werden sollte.

Erlaubnisregeln: Vor der Konfiguration einer Packet Screen mit Erlaubnisregeln muß genau analysiert werden, welche Kommunikationsanforderungen (benötigte Dienste) und Sicherheitsanforderungen (akzeptables Restrisiko) bestehen. Resultat dieser Analyse muß neben einer Aufstellung weiterer Absicherungsmaßnahmen eine kurze Liste sein, welche Dienste von welchen Systemen oder Gruppen von Systemen zugreifbar sein müssen. Aus dieser Liste können anschließend die Erlaubnisregeln für die Konfiguration der

Packet Screen erstellt werden. Die Liste der Erlaubnisregeln muß immer durch eine Verbotsregel für alle nicht explizit aufgeführten Dienste abgeschlossen werden.

Eine wichtige Aufgabe von Packet Screens ist der Schutz gegen "IP-Spoofing"-Angriffe (s. Abschnitt 2.1.1.2). Durch einfache Plausibilitätsüberprüfungen kann eine Packet Screen feststellen, ob die Absender-Adresse einer eingegangenen Dateneinheit zu einem Endsystem gehört, das über die jeweilige Schnittstelle erreichbar ist. [Ferguson et al. 98]

Wird eine Packet Screen in einem Router mit mehr als zwei Schnittstellen konfiguriert, muß die Konfiguration von Filterregeln sowohl vor als auch nach der Wegfindung ("Routing") möglich sein. Auf einer Schnittstelle hereinkommende Dateneinheiten müssen vor der Wegfindung überprüft werden, um "IP-Spoofing"-Angriffe erkennen zu können. Erst nach der Wegfindung sollten die für das Zielnetz geltenden Filterregeln angewendet werden, da zu diesem Zeitpunkt aufgrund der "Routing"-Entscheidung nur noch bestimmte Zieladressen vorkommen können und somit nur die auf diese Zieladressen zutreffenden Filterregeln ausgewertet werden müssen.

Neben den grundsätzlichen Vor- und Nachteilen der Firewall-Komponente "Packet Screen" (s. Abschnitt 2.3.1.1) werden abschließend Aspekte vorgestellt, die aus dem alleinigen Einsatz einer Packet Screen als Firewall-Architektur resultieren.

Vorteile bei ausschließlichem Einsatz einer Packet Screen:

Kosten: Wesentlicher Vorteil einer Firewall-Architektur, die nur aus einer Packet Screen besteht, sind die niedrigen Kosten für Installation und Betrieb. Die niedrigen Kosten können jedoch durch zusätzlich notwendige Absicherungsmaßnahmen bei den Endsystemen in der Gesamtkalkulation zu hohen Kosten führen.

Nachteile bei ausschließlichem Einsatz einer Packet Screen:

Große Anzahl bedrohter Rechner: Der Einsatz einer Packet Screen führt üblicherweise zu einer Konfiguration mit einer großen Anzahl erreichbarer und somit bedrohter Rechner. Die Sicherheit dieser Lösung richtet sich dann nach dem schwächsten internen System, da Angreifer, die hier eindringen können, von diesem System aus auch alle anderen internen Systeme direkt erreichen können ("Island Hopping"). Um trotzdem eine sichere Installation zu erreichen, ist daher ein hoher Absicherungsaufwand bei den internen Endsystemen notwendig.

Kontrolle interner Endsysteme: Die Sicherheit dieser Lösung hängt in hohem Maße von den internen Endsystemen ab, da von der Packet Screen hereingelassene Dateneinheiten von den Endsystemen sicher verarbeitet werden müssen. Bei vielen Netzinstallationen unterliegt die Verwaltung interner Endsysteme nicht der Kontrolle des Firewall-Verwalters bzw. bei einigen Endsystemen ist eine Trennung zwischen Verwaltungsaufgaben und normaler Benutzung gar nicht möglich. Neben der versäumten Installation sicherheitsrelevanter "Patches" und der unsicheren Konfiguration stellt auch die beabsichtigte Umgehung der Packet Screen ein Problem dar, weil gesperrte Dienste an alternativen, nicht gesperrten Ports bereitgestellt werden können.

Granularität: Die Überprüfung von Protokoll-Headern reicht für die Abwehr vieler Angriffe nicht aus.

2.3.2.2 Kombinationen von Packet Screen und Bastion

Die Probleme, die durch die große Anzahl bedrohter Rechner und durch die mangelhafte Kontrolle der Endsysteme bei der Packet Screen Architektur auftreten können, lassen sich durch eine weitere Einschränkung des Zugriffs lösen. Bei Kombinationen von Packet Screen und Bastion werden eine oder mehrere Packet Screens so konfiguriert, daß nur noch ein einzelnes Endsystem mit dem internen und dem externen Netz kommunizieren kann; die direkte Kommunikation zwischen internen und externen Endsystemen wird von der Packet Screen unterbunden. Dieses Endsystem wird als "Bastion" bezeichnet. Eine **Bastion** wird allgemein definiert als ein besonders abgesichertes System, das überwunden werden muß, um Endsysteme im jeweils anderen Netz zu erreichen. Auf der Bastion werden Proxies (s. Abschnitt 2.3.1.2) und Anwendungs-Gateways (s. Abschnitt 2.3.1.3) eingesetzt, um die Kommunikation zwischen internen und externen Systemen mit einer sehr feinen Granularität bei Zugriffskontrolle und Audit zu ermöglichen. Da die Bastion neben der Packet Screen als einziges System von beiden Netzen aus erreichbar ist, sind hier besondere Sicherungsmaßnahmen erforderlich.

Bei der Kombination von Packet Screen und Bastion wird abhängig von der Position der Bastion zwischen vier Konzepten unterschieden: (s. [Ellermann 93])

Bastion innen (Abbildung 2.7)
Die Installation der Bastion im internen Netz ist die am einfachsten zu realisierende Lösung. Jedoch besteht hierbei zwischen internem Netz und der Bastion keine weitere Abschirmung. Angriffe auf die Bastion, die von

internen Systemen ausgehen, müssen in diesem Fall von der Bastion allein abgewehrt werden.

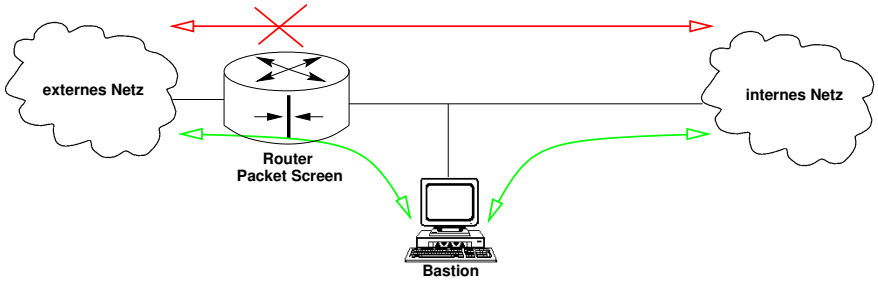


Abbildung 2.7: Kombination von Packet Screen und Bastion: konzeptionelle Sicht – Bastion innen

Bastion mittig (Abbildung 2.8)
Durch die Installation der Bastion in einem eigenen Netz an einer weiteren Schnittstelle der Packet Screen kann der Zugriff auf die Bastion vom externen und vom internen Netz besser eingeschränkt werden.

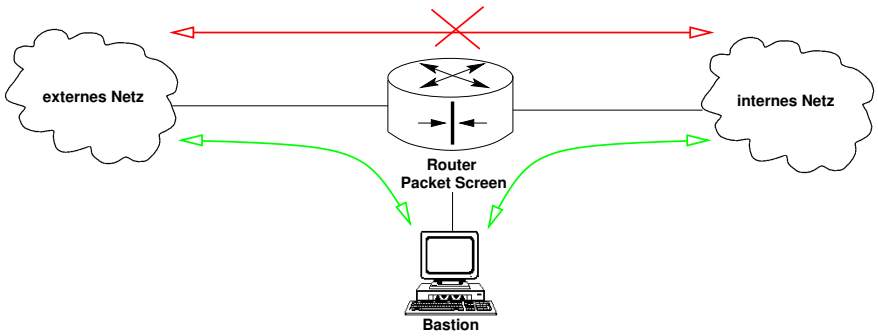


Abbildung 2.8: Kombination von Packet Screen und Bastion: konzeptionelle Sicht – Bastion mittig

Bastion außen (Abbildung 2.9)
Die Lösung, die Bastion vor der Packet Screen zu installieren, erscheint zunächst als schlechte Wahl, da Angriffe vom Internet aus auf die Bastion leicht möglich sind. Bei entsprechender Absicherung der Bastion kann mit dieser Lösung über sogenannte "Paket Sauger" (siehe [Ellermann 93, Abschnitt 4.4.1]) eine sehr detaillierte Aufzeichnung von Angriffen auf den Firewall erfolgen. Zusätzlich können von der Bastion aus alle Dienste im Internet uneingeschränkt zugegriffen werden.

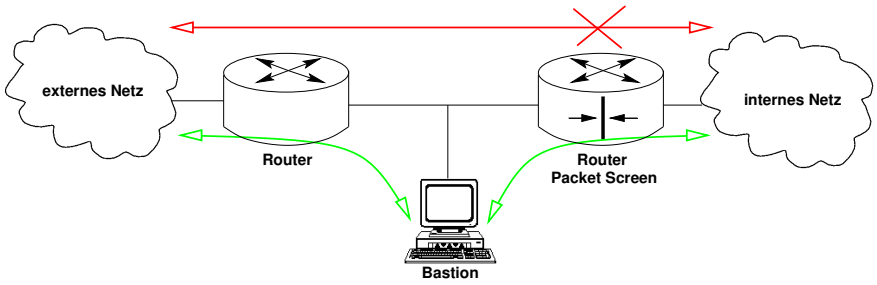


Abbildung 2.9: Kombination von Packet Screen und Bastion: konzeptionelle Sicht – Bastion außen

Bastion beidseitig abgeschirmt (Abbildung 2.10)

Bei der vorhergehend beschriebenen Installation der Bastion außen werden zwei Router benötigt, von denen aber nur einer als Packet Screen verwendet wird. Der zum externen Netz hin installierte Router ist in den meisten Fällen zusätzlich notwendig, um die Anbindung an das Internet technisch zu realisieren. Aus diesem Grund liegt es nahe, auch den zum Internet hin installierten Router als Packet Screen zu nutzen. Es ergibt sich dann die in Abbildung 2.10 vorgestellte Konfiguration mit einer beidseitig abgeschirmten Bastion. Das in beide Richtungen hin abgeschirmte Subnetz, an dem die Bastion installiert ist, wird als Grenznetz ("Screened Subnet", "Demilitarized Zone" oder "DMZ") bezeichnet.

Die Lösung der beidseitig abgeschirmten Bastion mit zwei Packet Screens kann zu der Lösung der mittig angeordneten Bastion optimiert werden, bei der eine Packet Screen logisch die Funktionen der beiden Packet Screens realisiert. Die Lösung mit zwei Packet Screens hat dieser vereinfachten Lösung gegenüber den Vorteil, daß ein Angreifer jeweils immer nur eine Packet Screen angreifen kann, während die zweite Packet Screen einen großen Anteil der Sicherheit des Firewalls weiterhin aufrechterhalten kann.

Vorteile der Kombination von Packet Screen und Bastion:

Granularität: Durch die Kombination von Packet Screen und Bastion kann eine sehr feine Granularität bei Zugriffskontrolle und Audit erreicht werden. Insgesamt wird ein sehr hohes Sicherheitsniveau erreicht.

Erweiterungsmöglichkeiten: Durch Einsatz zusätzlicher Bastionen kann diese Lösung bei Bedarf an höhere Leistungsanforderungen angepaßt werden.

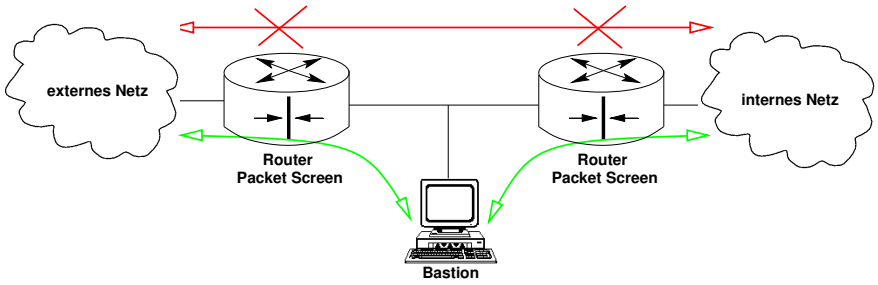


Abbildung 2.10: Kombination von Packet Screen und Bastion: konzeptionelle Sicht – beidseitig abgeschirmte Bastion

Nachteile der Kombination von Packet Screen und Bastion:

Kosten: Zur Realisierung diese Lösung ist zunächst ein hoher Installationsaufwand notwendig. Während des Betriebs können durch niedrigeren Aufwand bei Pflege und Überwachung der internen Systeme hingegen Kosten eingespart werden.

2.3.2.3 Gateway-Firewalls

Ein Gateway ist ein System, das an mindestens zwei Netze angeschlossen ist und das auf Anwendungsebene eine Kommunikation zwischen beiden Netzen ermöglicht. Durch Erweiterungen um Zugriffskontrolle und Audit zusammen mit einer besonderen Absicherung des Gateways ist der Einsatz als Firewall möglich (s. Abb. 2.11).[26] Dabei muß sichergestellt werden, daß alle Kommunikationsbeziehungen zwischen den angeschlossenen Netzen nur über den Firewall möglich sind. Auf dem Firewall kann mit Hilfe von Proxies und Anwendungs-Gateways eine sehr präzise Zugriffskontrolle definiert werden.



Abbildung 2.11: Gateway-Firewall: konzeptionelle Sicht

[26] Ein Gateway-Firewall ist per Definition eine Bastion.

Der Gateway-Firewall erreicht ähnliche Eigenschaften wie eine Kombination von Packet Screen und Bastion. Während bei letzterer Lösung die Packet Screen dafür sorgt, daß nur die Bastion mit beiden Netzen kommunizieren kann, muß bei einem Gateway-Firewall der gesamte Verkehr über dieses System geführt werden, da das Gateway als einziger Rechner an beide Netze angeschlossen ist. Eine zusätzliche Abschirmung des Gateway-Firewalls durch vorgeschaltete Packet Screens kann die Sicherheit eines Gateway-Firewalls kaum erhöhen. Das Grenznetz zwischen Packet Screen und Gateway-Firewall eignet sich jedoch hervorragend für die Installation von Informationsdiensten (Anonymous-FTP, WWW usw.), deren Installation auf einem Gateway-Firewall aus Sicherheitsgründen nicht empfohlen werden kann.

Vorteile von Gateway-Firewalls:

Granularität: Durch einen Gateway-Firewall kann eine sehr feine Granularität bei Zugriffskontrolle und Audit erreicht werden. Insgesamt wird ein sehr hohes Sicherheitsniveau erreicht.

Viele kommerzielle Lösungen: Für Gateway-Firewalls gibt es viele Anbieter.

Nachteile von Gateway-Firewalls:

Kosten: Zur Realisierung dieser Lösung ist zunächst ein hoher Installationsaufwand notwendig. Während des Betriebs können durch niedrigeren Pflege- und Überwachungsaufwand bei internen Systemen Kosten eingespart werden.

Erweiterungsmöglichkeiten: Im Gegensatz zu den Kombinationen von Packet Screen und Bastion besteht keine Erweiterungsmöglichkeit durch den Einsatz zusätzlicher Bastionen.

2.3.2.4 Firewall-Ergänzungen

Nach der Vorstellung der Firewall-Architekturen sollen in diesem Abschnitt die wichtigsten Ergänzungen und Varianten kurz angesprochen werden.

Informationsdienste: Zusätzlich zur Absicherung des internen Netzes wird von Firewall-Konzepten oft auch die sichere Integration von Informationsdiensten (Anonymous-FTP, WWW usw.) gefordert. Diese Dienste, die meist

eingesetzt werden, um öffentliche Informationen für das externe Netz bereitzustellen, haben andere Sicherheits- und Kommunikationsanforderungen als die Systeme im internen Netz. Um eine Gefährdung für das interne Netz durch die Informationsdienste ausschließen zu können, ist das Bereitstellen der Informationsdienste durch Systeme im internen Netz oder auf dem Firewall selbst nicht akzeptabel. Besser ist es, die Informationsdienste außerhalb des Firewalls anzubieten. Da der Informationsserver so nicht mehr durch den Firewall geschützt werden kann, erfordert diese Lösung jedoch eine zusätzliche Absicherung des Informationsservers.

Virtual Private Networks (VPNs): Die Verbindung entfernt voneinander liegender Teilnetze kann entweder über angemietete, teure Standleitungen oder beim Einsatz von Verschlüsselung auch über öffentliche Netze wie dem Internet erfolgen. Die Koppelung über öffentliche Netze mit Hilfe von Verschlüsselung wird als "Virtual Private Network" (VPN) bezeichnet. Eine Dateneinheit wird vom VPN-Endpunkt verschlüsselt und nach der Übertragung im öffentlichen Netz von einem weiteren VPN-Endpunkt wieder entschlüsselt. Die Ver- und Entschlüsselung geschieht für Sender und Empfänger transparent.

Die Planung eines VPNs muß im Zusammenhang mit dem Firewall-Konzept betrachtet werden, da je nach Positionierung von VPN und Firewall zueinander unterschiedliche Sicherheitseigenschaften erreicht werden [Kossakowski 98, S. 6f]. In jedem Fall ist zu beachten, daß die miteinander gekoppelten Netzwerke dasselbe Sicherheitsniveau erreichen müssen, da andernfalls Zugriffe aus einem weniger stark gesicherten Netz durch das VPN eine Bedrohung für besser gesicherte Netze darstellen können.

2.3.3 Nicht durch Firewalls abgedeckte Sicherheitsprobleme

Firewalls werden oft als "Allheilmittel" für die Netzwerksicherheit angesehen. Ein Firewall ist jedoch lediglich ein Zugriffskontrollmechanismus, der Netze in verschiedene Sicherheitsbereiche aufteilt. Um falsche Erwartungen zu verhindern, soll auch aufgeführt werden, was Firewalls nicht leisten können.

Keine Kontrolle des internen Verkehrs: Über den Firewall findet nur die Kommunikation zwischen internem und externem Netz statt. Der Datenverkehr zwischen internen Systemen untereinander kann durch den Firewall nicht kontrolliert werden. Für den Schutz gegen Insider-Angriffe ist daher weiterhin eine Absicherung der internen Systeme erforderlich. In manchen Fällen kann auch eine weitere Aufteilung des internen Netzes in mehrere über (Intranet-)Firewalls getrennte Sicherheitszonen sinnvoll sein.

Kein Schutz gegen Abhören oder Manipulation übertragener Daten: Ein Firewall kann Angriffe nur durch Überprüfung der übertragenen Daten erkennen. Da einer Packet Screen für diese Überprüfung nur die Protokoll-Header zur Verfügung stehen, sind hier nur sehr einfache Plausibilitätstests möglich.[27] Proxies können zwar auch die übertragenen Dateninhalte überprüfen, jedoch ermöglicht auch dies keine Erkennung von Manipulationen an übertragenen Daten (z. B. "TCP-Hijacking"). Statt eines Zugriffskontrollmechanismus sind kryptographische Verfahren zur Absicherung der übertragenen Daten erforderlich.

Keine Zugriffskontrolle auf verschlüsselten Daten: Der Einsatz kryptographischer Verfahren soll das Abhören oder die Manipulation von übertragenen Daten verhindern. Dadurch wird gleichzeitig auch eine Kontrolle dieser Daten durch einen Firewall verhindert.

Begrenzte Kontrollmöglichkeiten des Firewalls: Proxies und Anwendungs-Gateways erhalten zwar Zugriff auf alle übertragenen Daten, was eine Vielzahl von Angriffen theoretisch erkennbar macht. In der Praxis übersteigen aufwendige Überprüfungen jedoch die Leistungsfähigkeit des Firewalls.

Zur Absicherung der aufgeführten Probleme müssen andere Sicherheitsmechanismen eingesetzt werden. Über die aufgeführten Probleme hinaus kann der Firewall selbst zum Ziel von Angriffen werden, was schwerwiegende Konsequenzen haben kann, wenn die Sicherheit allein vom Firewall abhängt und die Sicherheitsmaßnahmen auf den internen Systemen vernachlässigt wurden.

Ein weiteres Problem sind "Denial of Service"-Angriffe, da der Firewall gewollt als Flaschenhals in der Kommunikation zwischen internem und externem Netz eingeführt wird, um hier eine Kontrolle zu ermöglichen. "Denial of Service"-Angriffe auf diesen Engpaß haben daher weitreichende Folgen, wenn der Firewall gegen diese Angriffe nicht resistent ist.

2.4 Überblick über die Techniken von ATM-Netzen

Die zunehmende Nutzung der Netzwerkkommunikation macht neue, schnellere Netze notwendig. Die beiden wichtigsten Entwicklungen im Bereich der Hochgeschwindigkeitsnetze sind dabei ATM und die Weiterentwicklungen der bekannten Ethernet-Technologie (Fast-Ethernet, Gigabit-Ethernet). Da sich die wesentliche

[27] "IP-Spoofing"-Angriffe können über einfache Regeln [Ferguson et al. 98] teilweise erkannt und abgewehrt werden.

Funktionsweise bei Fast- und Gigabit-Ethernet nicht geändert hat, unterscheiden sich diese Netze in erster Linie durch einen höheren maximalen Durchsatz (100 MBit/s bzw. 1 GBit/s) vom herkömmlichen Ethernet. ATM als verbindungsorientiertes Netz unterscheidet sich hingegen stark von den im LAN eingesetzten, rahmenorientierten ("frame oriented") "Broadcast"-Netzen wie Ethernet. Die ATM-Technologie eignet sich zudem für die Übertragung unterschiedlicher Verkehrsarten wie Telefonie, Video und Datenübertragung. Bei dem Einsatz von ATM-Netzen zur Datenübertragung werden drei unterschiedliche Ansätze verfolgt:

1. Durch zusätzliche Protokolle und Dienste wird den Endgeräten eine zu "Broadcast"-Netzen kompatible Schnittstelle angeboten. Die bisher in "Broadcast"-Netzen verwendeten Protokolle wie IP, Decnet und IPX können so direkt eingesetzt werden. Dieser Ansatz wird bei der "LAN Emulation" (LANE) [ATM Forum 95a] verfolgt. (s. Abschnitt 2.4.6)

2. Netzwerkprotokolle werden so angepaßt, daß sie direkt auf ATM aufsetzen können. Für die Übertragung von Protokoll-Dateneinheiten muß ein geeignetes Format definiert werden. Der größte Aufwand liegt jedoch darin, die Dienste anzupassen, die auf spezifische Eigenschaften von "Broadcast"-Netzen zurückgreifen, die bei ATM nicht zur Verfügung stehen. Bei "Classical IP and ARP over ATM" (CLIP) [Laubach 94] wird zum Beispiel für die Auflösung von IP-Adressen in ATM-Adressen[28] ein ATMARP-Server spezifiziert. Dieser ATMARP-Server wird benötigt, da ATM keine "Broadcasts" unterstützt, die bei der Ethernet-Variante von ARP genutzt werden, um Anfragen ungezielt in das Netz schicken zu können.

3. ATM kann auch ohne aufgesetzte herkömmliche Netzwerkprotokolle eingesetzt werden. Anwendungen können virtuelle Verbindungen direkt nutzen, um Daten auszutauschen. Dies erfordert zwar eine Neuentwicklung der Applikationen für ATM, ermöglicht aber die beste Nutzung von ATM-spezifischen Eigenschaften wie zum Beispiel die Reservierung von Betriebsmitteln. Diese Anwendungen werden als "Native"-ATM-Anwendungen bezeichnet.

Für lokale Netze, auf denen viele unterschiedliche Protokolle übertragen werden, ist der erste Ansatz besonders vorteilhaft, da nicht jedes Protokoll einzeln angepaßt werden muß.

[28] Der Sender muß die physikalische Netzadresse des Zielrechners kennen, um eine IP-Dateneinheit senden zu können. In einem Ethernet schickt der Sender zunächst eine ARP-Anfrage als "Broadcast" an alle Stationen des Ethernets. Eine Station, die die passende Ethernet-Adresse zu der angefragten IP-Adresse kennt, antwortet daraufhin. Bei ATM wird statt eines "Broadcasts" ein dedizierter ATMARP-Server verwendet, bei dem <*IP-Adresse,ATM-Adresse*>-Tupel des Subnetzes registriert sind.

Mit dem zweiten Ansatz kann hingegen eine bessere Performanz erreicht werden, da die Protokolle direkt auf ATM aufsetzen. Außerdem ermöglicht dies eine bessere Ausnutzung von ATM-spezifischen Eigenschaften, weil eine engere Verknüpfung mit ATM erreicht werden kann. Da IP in Weitverkehrsnetzen das vorherrschende Protokoll ist, wird hier in der Regel der zweite Ansatz vorgezogen werden. Der Hauptanwendungsbereich von Firewalls ist die Absicherung beim Anschluß von lokalen Netzen an Weitverkehrsnetze. Somit wird für Firewalls "Classical IP and ARP over ATM" der vorherrschende Einsatzbereich sein. Die folgenden Analysen konzentrieren sich daher auf "Classical IP and ARP over ATM"-Netze und auf das "Next Hop Resolution Protocol" (NHRP) als Erweiterung von CLIP. "LAN Emulation" (LANE) wird nur am Rande behandelt.

Während ATM bei den ersten beiden Ansätzen nur als weitere Netztechnologie zur Übertragung von IP-Datagrammen bzw. Ethernet-Rahmen genutzt wird, favorisieren Verfechter der ATM-Technologie den dritten Ansatz, der als einziger die optimale Ausnutzung von ATM-spezifischen Eigenschaften ermöglicht. ATM wird konsequent als zukünftiger Standard im LAN und im WAN bezeichnet und derzeit genutzte Protokolle als "Legacy"-Protokolle betrachtet. Da es bis heute nur wenige "Native"-ATM-Anwendungen gibt und da "Native"-ATM-Anwendungen erst dann eine ähnliche Verbreitung wie IP-Anwendungen finden können, wenn die bisher bestehenden "ATM-Inseln" miteinander verknüpft worden sind, ist nicht davon auszugehen, daß IP durch "Native"-ATM verdrängt werden wird.

Da heute die wenigsten Rechner direkt an ein ATM-Netz angeschlossen sind und die Erreichbarkeit der bereits vorhandenen IP-Hosts auch zukünftig weiterhin eine Unterstützung von IP erforderlich machen wird, ist für die nähere Zukunft davon auszugehen, daß "Native"-ATM-Anwendungen nur ergänzend zu IP in speziellen Einsatzbereichen eine Rolle spielen werden. Auch in der ferneren Zukunft werden weiterhin beide Protokolle eingesetzt werden. IP und ATM haben ihre Stärken in jeweils anderen Bereichen, so daß sich beide Technologie nicht gegenseitig ausschließen, sondern einander ergänzen. Wegen des aufwendigen Verbindungsaufbaus eignet sich ATM weniger zur Übertragung kurzer Nachrichten wie sie zum Beispiel bei DNS ausgetauscht werden. Das Datagramm-orientierte IP wird trotz Erweiterungen wie RSVP[29] demgegenüber nie eine Dienstgüte so garantieren können, wie es bei ATM möglich ist. Firewalls müssen daher zukünftig zusätzlich zu IP-Anwendungen auch "Native"-ATM-Anwendungen unterstützen, wobei der Schwerpunkt weiterhin bei den IP-Anwendungen bleibt.

[29] "Resource ReSerVation Protocol" (RSVP) [Braden et al. 97]

2.4.1 Kommunikation in ATM-Netzen

Einleitend soll die Kommunikation in ATM-Netzen skizziert werden.[30] "Asynchronous Transfer Mode" (ATM)[31] überträgt Daten in *Zellen*. Jede Zelle hat immer die gleiche Länge von 53 Byte, bestehend aus 48 Byte Nutzdaten und 5 Byte Header-Informationen. Auf jedem Abschnitt eines ATM-Netzes werden kontinuierlich Zellen gesendet. Falls keine Nutzdaten anliegen, werden gekennzeichnete, leere Zellen übertragen. Der Aufbau einer Zelle ist in Abbildung 2.12 dargestellt.[32]

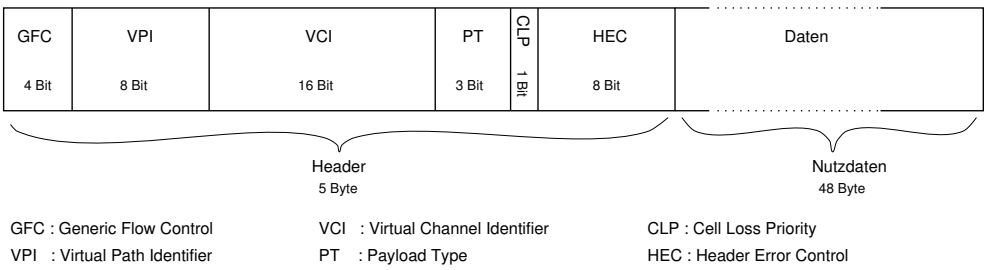


Abbildung 2.12: Aufbau einer ATM-Zelle

Virtuelle Verbindungen: Die verbindungsorientierte Zell-Weiterleitung ist das Basiskonzept von ATM. Zellen haben keine Sender- und Empfänger-Adressen, sondern nur eine lokale Kennung für einen bestimmten virtuellen Kanal ("Virtual Channel"): "Virtual Path Identifier" (VPI) und "Virtual Channel Identifier" (VCI). Anders als bei IP sind dies jedoch keine global eindeutigen Adreßinformationen sondern nur lokal gültige Angaben, die eine virtuelle Verbindung zu einem Ziel spezifizieren. Eine virtuelle Verbindung entsteht aus der Verkettung von virtuellen Kanälen und muß vor dem Beginn der Kommunikation aufgebaut

[30] Weiterführende Informationen zu ATM finden sich in [Alles 95, Badach et al. 94, Dorling et al. 96, Hein 97, Hein et al. 98, Minoli et al. 97, Peterson et al. 96, Scharf 96, Schill et al. 97].

[31] Die Übertragung von Daten über ein ATM-Netz wird als asynchron bezeichnet, weil die Aufteilung einer Leitung auf mehrere Verbindungen über ein asynchrones Zeitmultiplexing-Verfahren erfolgt. Für die Zuordnung werden bei ATM die "Virtual Path Identifier" (VPI) und "Virtual Channel Identifier" (VCI) Kontrollinformationen verwendet. Im Vergleich zu ATM werden bei ISDN drei Kanäle (zwei Daten- und ein Steuerungskanal) über ein synchrones Verfahren fest zugeordnet. Jeder Kanal erhält bei ISDN eine feste Zeitscheibe. Eine bedarfsorientierte Zuordnung der Ressourcen an eine variierende Anzahl von Verbindungen ist nur mit asynchronen Verfahren möglich.

[32] Das dargestellte Zellformat entspricht dem Format an der Schnittstelle zwischen Endgerät und ATM-Switch. Der GFC entfällt am Übergang zwischen zwei Switches; dafür verlängert sich der VPI auf 12 Byte.

werden. Auf dem Weg einer Zelle zu ihrem Ziel überprüft jeder dazwischen liegende ATM-Switch das VPI/VCI-Paar einer hereinkommenden Zelle und wählt den dazu gehörenden herausgehenden virtuellen Kanal aus. Der herausgehende virtuelle Kanal zum nächsten Switch oder zum Endgerät kann dabei ein anderes VPI/VCI-Paar nutzen (s. Abb. 2.13), so daß die VCI/VPI-Informationen vor dem Senden mit den Werten für den herausgehenden virtuellen Kanal überschrieben werden müssen. Dieser Prozeß kann sehr effizient in der Hardware eines ATM-Switches implementiert werden.

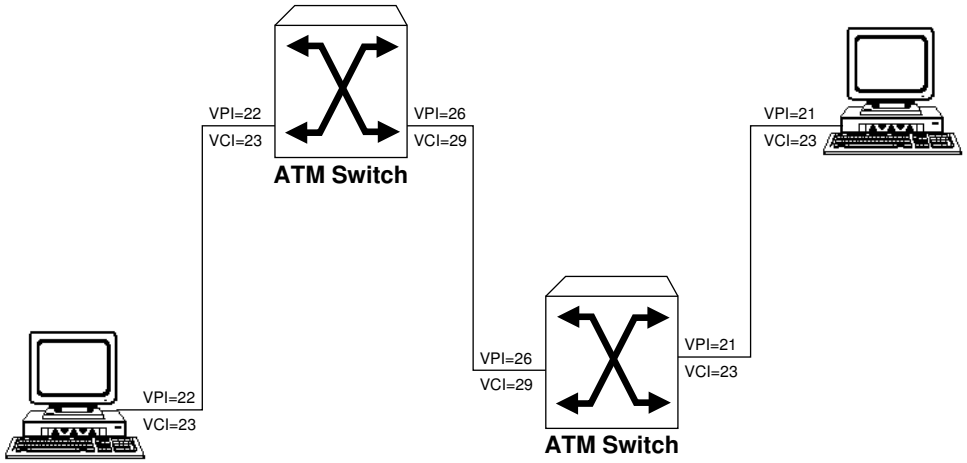


Abbildung 2.13: Virtuelle Verbindung

Eine virtuelle Verbindung kann entweder als "Permanent Virtual Circuit" (PVC) permanent konfiguriert sein, oder sie wird bei Bedarf als "Switched Virtual Circuit" (SVC) eingerichtet. Ein PVC muß manuell auf den Endsystemen und allen auf dem Weg liegenden Switches konfiguriert werden. Der hohe Verwaltungsaufwand macht PVCs nur in wenigen Anwendungsbereichen einsetzbar.

SVCs werden von den Endsystemen bei Bedarf geöffnet. Über das "User Network Interface" (UNI) kann das Endsystem dem angeschlossenen Switch den Verbindungswunsch zum Zielrechner und die geforderte Dienstgüte mitteilen. Der Switch signalisiert die Daten an den nächsten Switch auf dem Weg zum Empfänger. Dabei werden die jeweiligen VPI/VCI-Paare für den SVC registriert und überprüft, ob die geforderte Dienstgüte ("Quality of Service", QoS) unterstützt werden kann.

"User-Network Interface" (UNI): Die wichtigste Aufgabe der UNI-Signalisierung ist der Auf- und Abbau von virtuellen Punkt-zu-Punkt-Verbindungen. Für den Aufbau einer virtuellen Verbindung werden die UNI-Nachrichten "SETUP", "CONNECT", "CONNECT_ACKNOWLEDGE" und falls notwendig "CALL_

PROCEEDING" verwendet. Der Abbau einer virtuellen Verbindung erfolgt über "RELEASE"- und "RELEASE_COMPLETE"-Nachrichten. In Abbildung 2.14 ist der prinzipielle Ablauf beim Verbindungsauf- und -abbau wiedergegeben. Die UNI-Signalisierungsnachrichten werden auf einem fest definierten PVC (VPI=0, VCI=5) übermittelt.

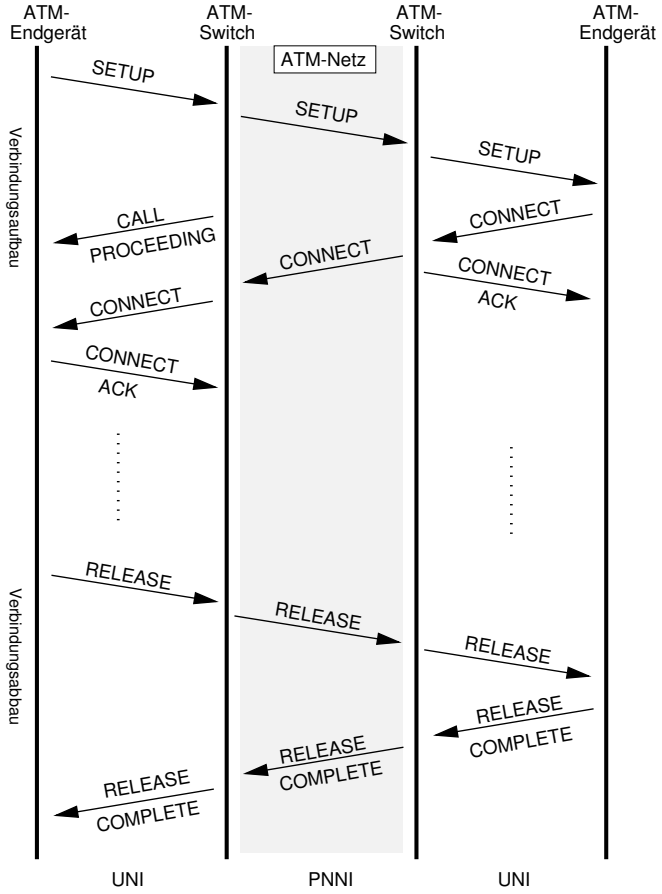


Abbildung 2.14: Signalisierung zum Auf- und Abbau einer virtuellen Verbindung

Für die Signalisierung zwischen Endgerät und Switch sind derzeit drei unterschiedliche UNI-Versionen im Einsatz. Am gebräuchlichsten ist UNI 3.1 [ATM Forum 94], ältere Systeme verwenden jedoch teilweise noch das *inkompatible* UNI 3.0. Absehbar ist schon der Übergang zu UNI 4.0 [ATM Forum 96d], das die Verwendung zusätzlicher Verkehrsparameter ermöglicht.

"Private Network-Network Interface" (PNNI): Eine Kommunikation über ein beliebig komplexes Netz hinweg, kann nur dann zustande kommen, wenn es möglich ist, einen Pfad zum Zielsystem zu finden. In herkömmlichen Datagramm-orientierten Netzen wie dem Internet wird für jedes Datagramm anhand der

Adreßinformationen der nächste Schritt zum Ziel bestimmt. Bei ATM-Netzen geschieht diese Wegwahl einmalig beim Aufbau einer virtuellen Verbindung. Das ATM-Forum hat für diese Aufgabe das "Private Network-Network Interface" (PNNI) [ATM Forum 96c] definiert. PNNI implementiert zwei Funktionen:

Verwaltung von Erreichbarkeitsinformationen: Mit sogenannten "HELLO"-Dateneinheiten sammelt jeder Switch Informationen über die Erreichbarkeit benachbarter Switches und tauscht diese Informationen mit anderen Switches aus. Da die Menge der auszutauschenden Informationen mit der Anzahl der Switches rasch steigt, wird bei PNNI eine hierarchische Struktur verwendet. Dabei werden mehrere Switches zu einer sogenannten "Peer Group" (PG) zusammengefaßt, die zwar untereinander ihre gesamten Erreichbarkeitsinformationen austauschen, nach außen über ihren "Peer Group Leader" (PGL) jedoch nur aggregierte Erreichbarkeitsinformationen weitergeben. Die PGL bilden auf einer höheren Hierarchiestufe wiederum eine PG mit einem eigenen PGL. Durch diese Hierarchie wird erreicht, daß jeder Switch Erreichbarkeitsinformationen über das gesamte Netz erhält. Gleichzeitig wird die Datenmenge reduziert, da die Erreichbarkeitsinformationen anderer PGs nur aggregiert vorliegen.

Signalisierung zum Aufbau virtueller Verbindungen: Für einen über UNI von einem Endgerät an einen ATM-Switch übermittelten Verbindungswunsch kann bereits der erste ATM-Switch mit Hilfe der gesammelten Erreichbarkeitsinformationen den Pfad bis zum Zielsystem bestimmen. Dieses Verfahren wird auch als "Source Routing" bezeichnet. Der Verbindungsaufbauwunsch des Endsystems wird zusammen mit dem generierten Pfad ("Designated Transit List", DTL) vom ATM-Switch in einer PNNI-Nachricht an den nächsten Switch in der DTL gesendet. Am Übergang zu einer benachbarten PG müssen jeweils die aggregierten Pfadinformationen dieser PG durch detaillierte Informationen ersetzt werden.

Diese PNNI-Signalisierungsnachrichten werden wie auch die UNI-Signalisierungsnachrichten auf einem fest definierten PVC zwischen benachbarten Knoten ausgetauscht. Für PNNI wird der PVC (VPI=0, VCI=18) verwendet.

"Integrated Local Management Interface" (ILMI): Für die Unterstützung von Netzwerkmanagementaufgaben wird mit dem "Integrated Local Management Interface" (ILMI) [ATM Forum 96e] ein auf dem "Simple Network Management Protocol" (SNMP) basierendes Protokoll eingesetzt, dessen wichtigste Aufgabe derzeit die Übertragung von Konfigurationsinformationen zwischen ATM-Switch und ATM-Endgerät ist. Die automatische Konfiguration über ILMI ermöglicht das sogenannte "Plug-and-Play", die Installation von Komponenten, ohne eine

manuelle Konfiguration zu erfordern. ILMI überträgt nur Informationen zwischen benachbarten Knoten auf einem festen PVC (VPI=0, VCI=16).

"Quality of Service" (QoS): Wichtiges Merkmal von ATM ist, daß beim Aufbau einer virtuellen Verbindung die geforderte Dienstgüte ("Quality of Service", QoS) spezifiziert werden kann. Hierzu werden die geforderten QoS-Parameter in den Signalisierungsnachrichten beim Verbindungsaufbau übertragen, so daß jeder Switch und das Endsystem bereits beim Verbindungsaufbau sicherstellen können, daß die geforderten Parameter später eingehalten werden können. Andernfalls wird der Verbindungsaufbau abgelehnt. Sind die Dienstgüteanforderungen erfüllbar, werden die für die Einhaltung der Dienstgüte benötigten Betriebsmittel (Ressourcen) für diese virtuelle Verbindung reserviert und erst beim Verbindungsabbau wieder freigegeben.

Die Reservierung der benötigten Ressourcen bedeutet jedoch nicht, daß der Sender sich in jedem Fall auch an diese Vereinbarungen halten muß. Es ist durchaus für den Sender möglich, die vereinbarten Grenzen zu überschreiten. Sollten die dann benötigten zusätzlichen Ressourcen verfügbar sein, werden die Zellen den Empfänger voraussichtlich erreichen. Der Sender muß jedoch damit rechnen, daß die über die Vereinbarung hinaus gesendeten Daten auf dem Pfad zum Empfänger vernichtet werden, wenn keine ausreichenden Ressourcen zur Verfügung stehen.

Für unterschiedliche Anwendungen sind auch unterschiedliche Strategien bei der Dienstgüte empfehlenswert. Deshalb werden mehrere verschiedene Verkehrsklassen definiert. Abbildung 2.15 stellt den Datenverkehr für die jeweilige Verkehrsklasse aus Netzsicht dar.

"Constant Bit Rate" (CBR): CBR dient für die Übertragung eines gleichmäßigen Datenstroms mit fester Verzögerung und kann zum Beispiel für Audio und Video verwendet werden.

"Real-Time Variable Bit Rate" (rt-VBR): rt-VBR ist ebenfalls eine Übertragung mit fester Verzögerung, jedoch wird nicht kontinuierlich die gleiche Datenmenge gesendet. Der vom Sender nur kurzzeitig benötigte maximale Durchsatz (Spitzenlast) wird angegeben als "Peak Cell Rate" (PCR). Der mittlere obere Durchsatz (Dauerlast) wird über die "Sustainable Cell Rate" (SCR) festgelegt. Die maximale Länge, für die die SCR überschritten werden darf, wird als "Maximum Burst Size" (MBS) bezeichnet. Diese Verkehrsklasse eignet sich für komprimierte Video-Signale, bei denen kontinuierlich nur die kleinen Unterschiede zwischen aufeinanderfolgenden Bildern gesendet werden und nur bei Bildschnitten große Datenmengen übertragen werden müssen.

"Non-Real-Time Variable Bit Rate" (nrt-VBR): Bei dem Abspielen von Video- oder Audio-Aufzeichnungen (z.B. bei "Video on Demand") können auch größere Verzögerungen im Sekundenbereich durch Pufferung beim Empfänger ausgeglichen werden. Hierfür eignet sich die Verkehrsklasse "Non-Real-Time Variable Bit Rate" (nrt-VBR).

"Unspecified Bit Rate" (UBR): Bei UBR kann lediglich die "Peak Cell Rate" (PCR) spezifiziert werden, weitere Dienstgüteanforderungen können nicht unterstützt werden.

"Available Bit Rate" (ABR): Der momentan nicht von anderen Anwendungen in Anspruch genommene Durchsatz kann von der Verkehrsklasse "Available Bit Rate" (ABR) verwendet werden. Abhängig von dem Verhalten anderer Sender können hier sehr stark schwankende Durchsätze erzielt werden. Für ABR kann eine "Minimum Cell Rate" (MCR) angefordert werden, die das Netz dem Sender jederzeit mindestens zur Verfügung stellen muß.

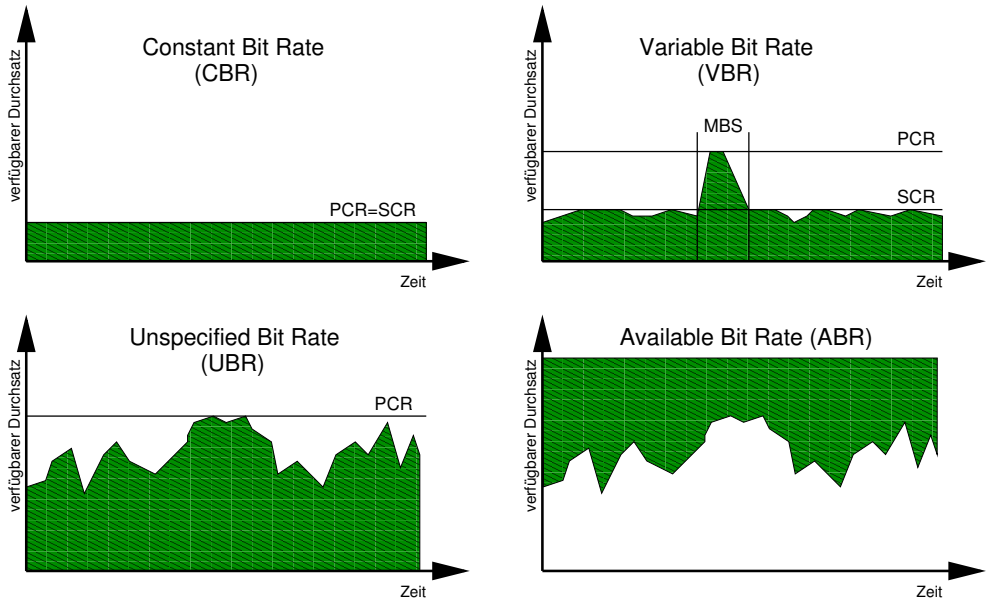


Abbildung 2.15: Überblick über die Verkehrsklassen bei ATM

ATM-Anpassungsschicht (AAL): Für die Übertragung von Nutzdaten wird auf ATM eine Anpassungsschicht — "ATM Adaption Layer" (AAL) gelegt. Sie sorgt für das Zerlegen der Nutzdaten in Zellen und nach der Übertragung wieder für das Zusammenfügen der Zellen ("Segmentation and Reassembly"). Da die ATM nutzenden Anwendungen (Video, Audio oder Datenübermittlung) unterschiedliche Eigenschaften und Anforderungen haben, werden für verschiedene

Diensttypen angepaßte AAL-Typen angeboten. Für Telefonie und Videoübertragungen bestehen beispielsweise Anforderungen hinsichtlich der Verzögerung von Zellen, der Varianz der Verzögerung sowie der Synchronisation zwischen Sender und Empfänger. Für die Kopplung von TK-Anlagen wird AAL-1 eingesetzt.

Die Datenübermittlung hat demgegenüber viel geringere Anforderungen. Hier ist in erster Linie die Effizienz der eingesetzten Verfahren wichtig, um einen möglichst geringen "Overhead" bei der Übertragung zu erzielen. Hier wird hauptsächlich AAL-5 eingesetzt.

Integration höherer Protokollschichten über ATM: Für einige ausgewählte Protokolle werden bei ATM fest vordefinierte virtuelle Verbindungen verwendet. Bereits erwähnt wurden UNI (VPI=0, VCI=5), PNNI (VPI=0, VCI=18) und ILMI (VPI=0, VCI=16). Grundsätzlich sind VPI=0 mit VCIs von 0 bis 31 für Protokolle der ITU und des ATM-Forums reserviert. Alle anderen VPI/VCI-Paare können für beliebige Protokolle genutzt werden.

Die Auswahl des auf einer virtuellen Verbindung genutzten Protokolls erfolgt beim Verbindungsaufbau innerhalb der Signalisierungsnachrichten über das "Broadband Low Layer Information"-"Information Element" (BLLI-IE) [ATM Forum 94, S. 212ff]. Im BLLI-IE können Schicht-2- und Schicht-3-Protokolle definiert werden, die auf der virtuellen Verbindung genutzt werden sollen. Die für IP-Verkehr wichtigsten beiden Fälle sind der Transport von IP-Datagrammen in AAL-5-Rahmen und die Übertragung von IP-Dateneinheiten mit zusätzlichem SNAP-Header innerhalb von AAL-5-Rahmen. Der erste Fall, bei dem auf der virtuellen Verbindung ausschließlich IP-Dateneinheiten transportiert werden können, wird als *VC-basiertes Multiplexing* bezeichnet, da für jedes weitere Protokoll eine zusätzliche virtuelle Verbindung (VC) verwendet werden muß. Bei der zweiten Möglichkeit ist in jeder AAL-5-Dateneinheit ein SNAP-Header [Heinanen 93] enthalten, in dem über den "EtherType" das in dieser AAL-5-Dateneinheit übertragene Protokoll definiert wird. Dadurch ist es möglich, auf derselben virtuellen Verbindung verschiedene Protokolle zu verwenden. Diese Form des Multiplexings wird auch als *"LLC Encapsulation"* bezeichnet und erschwert die Kontrolle der verwendeten Protokolle durch Firewalls.

Bei sogenannten "Native"-ATM-Anwendungen muß zusätzlich zum BLLI-IE die angesprochene Entität des Zielsystems über ein "Broadband High Layer Information"-"Information Element" (BHLI-IE) [ATM Forum 94, S. 211f] identifiziert werden. "Native"-ATM-Anwendung werden in [ATM Forum 96b] beschrieben. Das BHLI-IE für den Zugriff auf die "Native"-ATM-Anwendung "Video on Demand" ist in [ATM Forum 97a, S. 27] spezifiziert.

2.4.2 "Classical IP" (CLIP)

Für die Versendung von IP-Dateneinheiten muß der Sender die physikalische Adresse des Empfängers kennen. Für die Ermittlung dieser physikalischen Adresse wird in herkömmlichen Netzen das "Address Resolution Protokoll" (ARP) [Plummer 82] verwendet. Bei ARP schickt der Sender einen "Broadcast" mit einer Anfrage ("ARP_Request"), um die zu einer IP-Adresse gehörende physikalische Adresse zu ermitteln. Wenn die zuständige Station den "Broadcast" empfängt, sendet sie als Antwort ("ARP_Reply") das Tupel aus physikalischer Adresse und IP-Adresse zurück.

In einem ATM-Netz steht keine "Broadcast"-Funktionalität zur Verfügung, mit der "ARP_Requests" an alle Stationen eines Netzes gesendet werden können. Daher wird ARP bei CLIP durch eine Client-Server-Lösung ersetzt. Bei dem zentralen ATMARP-Server müssen sich alle Systeme (ATMARP-Clients), die über das CLIP-Netz kommunizieren wollen, zunächst mit ihrer ATM-Adresse und IP-Adresse registrieren. Wenn anschließend IP-Dateneinheiten versendet werden sollen, sendet der ATMARP-Client zunächst eine ATMARP-Anfrage an den ATMARP-Server, um die ATM-Adresse des Empfängers zu erfahren. Der ATMARP-Client baut anschließend zu dieser ATM-Adresse eine virtuelle Verbindung (SVC) auf, über die die beiden Stationen dann IP-Dateneinheiten austauschen.

Virtuelle Verbindungen werden nur zwischen Systemen desselben logischen IP-Subnetzes ("Logical IP Subnet", LIS) aufgebaut; Systeme außerhalb des LIS müssen über Router angesprochen werden. Ein LIS hat folgende Eigenschaften: [Luciani et al. 98a]

- Alle Systeme eines LIS gehören zum selben IP-Subnetz (gleiche IP-Netzadresse und Subnetzmaske).[33]
- Alle Systeme eines LIS sind direkt an dasselbe ATM-Netz angeschlossen.
- Systeme außerhalb des LIS sind über Router erreichbar.
- Alle Systeme innerhalb eines LIS kommunizieren direkt, d. h. ohne die Hilfe eines Routers, miteinander.

Das CLIP-Modell orientiert sich an dem Vorgehen bei herkömmlichen Netzen, wo zum Beispiel in einem Ethernet alle Stationen direkt miteinander kommunizieren können und für die Kommunikation mit Systemen in anderen Subnetzen der Weg über den Router gewählt werden muß. In den Endsystemen und Routern sind daher auch nur geringe Änderungen notwendig.

[33] Z. B. 192.168.14.0/255.255.255.0

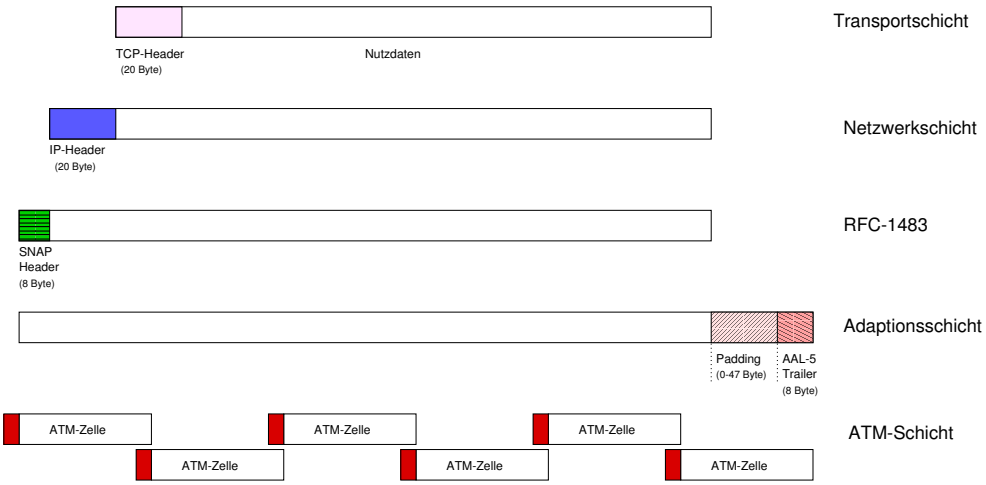


Abbildung 2.16: "Classical IP"-Protokollstapel

Auf der von einem ATMARP-Client geöffneten virtuellen Verbindung zu einem anderen System im LIS wird jede IP-Dateneinheiten mit einem SNAP-Header (RFC 1483, [Heinanen 93]) versehen und in einem AAL-5-Rahmen übertragen (s. Abbildung 2.16). Ein weiterer Parameter, der von CLIP [Laubach et al. 98] festgelegt wird, ist die maximale Länge, die eine IP-Dateneinheit erreichen darf, ohne daß sie zur Übertragung fragmentiert werden muß. Die "Maximum Transfer Unit" (MTU) ist bei CLIP mit 9.180 Byte definiert. Optional dürfen abweichende MTUs beim Aufbau der virtuellen Verbindung ausgehandelt werden, jedoch ist die MTU durch die maximale Länge der in einem AAL-5-Rahmen übertragbaren Daten begrenzt (64 KByte). Bei herkömmlichen Netzen sind sehr viel geringere MTUs gebräuchlich (z. B. Ethernet mit 1.500 Byte). Bei größeren IP-Dateneinheiten, die durch die größere MTU möglich werden, ist natürlich der anteilige "Overhead" durch die Protokoll-Header geringer, jedoch kann eine große MTU wie bei CLIP auch zu unerwünschten Effekten führen, die negative Auswirkungen auf die Performanz von TCP haben [Moldeklev et al. 95].

2.4.3 "Next Hop Resolution Protocol" (NHRP)

Bei größerer Verbreitung von ATM-Netzen ist der größte Nachteil von CLIP-Netzen, daß die Kommunikation zwischen Systemen verschiedener LIS immer über Router geführt werden muß, auch wenn das ATM-Netz eine direkte virtuelle Verbindung zwischen beiden Endsystemen ermöglichen würde. Das "NBMA Next Hop Resolution Protocol" (NHRP)[34] stellt eine Erweiterung von ATMARP

[34] Wichtigste Vertreter für "Non-Broadcast, Multiple-Access"-(NBMA)-Netze sind ATM-Netze.

dar. Der "Next Hop Server" (NHS) kann zusätzlich zu den Funktionen eines ATMARP-Servers auch Anfragen nach Adressen außerhalb des LIS bearbeiten. Durch diese Erweiterung der Adreßauflösung ("Address Resolution") auf Systeme außerhalb des LIS wird es für einen "Next Hop Client" (NHC) möglich, virtuelle Verbindungen zu Systemen in anderen LIS aufzubauen.

Für die Kommunikation innerhalb eines LIS verhält sich der NHS wie ein ATMARP-Server. NHCs müssen sich zunächst beim NHS registrieren, bevor sie anschließend Anfragen ("NHRP Resolution Requests") an den NHS senden können. Ein NHS kann daher auch als ATMARP-Server eingesetzt werden [Luciani 98]. Unterschiedlich ist das Verhalten erst bei der Kommunikation mit IP-Adressen außerhalb des LIS. Ein NHC kann eine Anfrage nach Auflösung einer IP-Adresse außerhalb des LIS an den NHS senden. Wenn der NHS die ATM-Adresse für die angefragte IP-Adresse nicht kennt, wird die Anfrage an den nächsten NHS entlang des Pfades zum Empfänger weitergeleitet. Der Pfad, den die Anfrage nimmt, wird dabei durch die "Routing"-Informationen bestimmt, die auch den Weg der IP-Dateneinheit zum Empfänger bestimmen. Ein NHS muß daher mit einem Router gekoppelt werden.[35] Die NHRP-Anfrage wird solange weitergeleitet, bis ein NHS erreicht wird, bei dem die Adresse des Empfängers registriert ist. Der NHC erhält die gefundene ATM-Adresse und kann eine virtuelle Verbindung direkt zum Empfänger aufbauen. In dem Fall, daß sich der Empfänger außerhalb des ATM-Netzes befindet und somit keine ATM-Adresse des Empfängers verfügbar ist, wird die Adresse des letzten Routers im ATM-Netz ("Egress-Router") an den NHC zurückgesendet.

Für einen NHC gibt es zwei Möglichkeiten, Systeme außerhalb des LIS zu erreichen; neben der direkten virtuellen Verbindung zu einer ATM-Adresse, die über NHRP ermittelt wird, besteht weiterhin die Möglichkeit, das "Forwarding" von IP-Dateneinheiten über Router zu nutzen. Die Entscheidung, welcher der beiden Wege gewählt werden soll, liegt im Ermessen des NHCs. Der Aufbau einer virtuellen Verbindung direkt zum Endsystem kann erforderlich sein, wenn zum Beispiel bei einer Applikation die Einhaltung gewisser "Quality of Service"-Parameter erforderlich ist. Andererseits ist der Aufbau einer virtuellen Verbindung mit einer Verzögerung[36] und eventuell auch mit Kosten verbunden. Daher lohnt sich der Aufbau einer virtuellen Verbindung nicht für kürzere Kommunikation (z. B. DNS-Anfragen oder Auslieferung einer E-Mail über SMTP). Besteht die Kommunikation nur aus dem Austausch weniger Dateneinheiten, ist das "Forwarding" von IP-Dateneinheiten über Router effizienter als der Aufbau einer virtuellen Verbindung. Dadurch wird die "Routing"-Entscheidung für Endsysteme komplizier-

[35] Demgegenüber kann ein ATMARP-Server auch auf einem ATM-Switch oder sogar auf einem Endgerät installiert sein.

[36] Die Verzögerung ergibt sich aus Dauer für die Beantwortung der NHRP-Anfrage und der Verbindungsaufbauzeit für den SVC.

ter. Endsysteme in herkömmlichen IP-Netzen müssen anhand der Empfänger-IP-Adresse unterscheiden zwischen lokalen Adressen, die direkt angesprochen werden können, und Adressen anderer IP-Subnetze, die über einen Router angesprochen werden müssen. Bei NHRP muß diese "Routing"-Entscheidung um einen dritten Aspekt, die QoS-Anforderungen der Applikation, erweitert werden. Zur Realisierung dieser erweiterten "Routing"-Entscheidung geben die RFCs für NHRP (RFC 2332 [Luciani et al. 98a] und RFC 2333 [Cansever 98]) keine weiteren Hinweise.

Zusätzlich zu Endsystemen können auch Router NHRP einsetzen, um die ATM-Adressen von Empfängern zu ermitteln, zu denen IP-Datagramme auf einer direkten virtuellen Verbindung gesendet werden sollen. Dies führt jedoch zu zwei Problemen. Wenn jeder Router auf einem Pfad zwischen zwei Kommunikationspartnern versucht, virtuelle Verbindungen direkt zum Empfänger aufzubauen, kann dies dazu führen, daß unnötig viele virtuelle Verbindungen für dieselben Kommunikationspartner aufgebaut werden. Tatsächlich genutzt wird davon anschließend aber nur eine virtuelle Verbindung, da alle anderen Router mit dieser virtuellen Verbindung "übersprungen" werden. Dieses Problem kann vermieden werden, wenn nur der erste NHRP-fähige Router auf dem Pfad zu Empfänger NHRP-Anfragen senden darf, um für diese Kommunikationsbeziehung eine direkte virtuelle Verbindung einzurichten. Ein noch schwerwiegenderes Problem ist das Auftreten von "Routing-Loops", die auftreten können, wenn NHRP benutzt wird, um virtuelle Verbindungen zwischen Routern aufzubauen [Cole et al. 96, S. 18ff]. Dieses Problem wird bei NHRP umgangen, indem gefordert wird, daß virtuelle Verbindungen nicht zwischen Routern aufgebaut werden dürfen, beziehungsweise daß die virtuelle Verbindung auf mindestens einer der beiden Seiten bei einem Endsystem enden muß. Dies stellt eine sehr starke Einschränkung für die Verwendbarkeit von NHRP dar. Der Einsatz von NHRP zwischen Routern wird derzeit noch erforscht [Rekhter et al. 98].

NHRP darf nicht mit einem "Routing"-Protokoll (OSPF, RIP usw.) verwechselt werden. NHRP ist lediglich eine Erweiterung der Adreßauflösung, "Routing"-Protokolle werden weiterhin benötigt.

NHRP liegt die implizite Annahme zugrunde, daß eine direkte virtuelle Verbindung über ein ATM-Netz immer effizienter ist als ein Pfad über mehrere Router. Diese Annahme trifft nicht immer zu, da ein SVC nur einmal eröffnet wird und dann über eine längere Zeit verwendet wird. Wenn nach Öffnen des SVCs eine kürzere Verbindung (z. B. nach einem Ausfall) wieder verfügbar wird, bleibt trotzdem der ungünstige SVC bestehen. Das Eröffnen eines SVCs auf dem kürzesten Pfad kann auch deshalb nicht klappen, weil der angeforderte Durchsatz hier nicht zur Verfügung steht. Zwischen Routern werden hingegen häufig PVCs verwendet oder es dürfen nach den für das ATM-Netz festgelegten Richtlinien für die SVCs zwischen Routern höhere Durchsätze reserviert werden als für SVCs zwischen

Endgeräten. In diesen Fällen kann ein Pfad über mehrere Router effizienter sein als eine direkte virtuelle Verbindung.

2.4.4 "Multicast Address Resolution Server" (MARS)

"Multicast" bezeichnet das gleichzeitige Senden einer Nachricht an mehrere Empfänger. In "Broadcast"-orientierten Netzen ist die "Multicast"-Fähigkeit einfach zu implementieren, da jede übertragene Nachricht ohnehin jedes Endgerät erreicht und jedes Endgerät somit allein über die Auswahl der zu empfangenden Daten entscheiden kann. Um herkömmliche "Unicast"-Nachrichten von "Multicast"-Nachrichten zu unterscheiden, wird für "Multicasts" ein spezieller IP-Adreßbereich von 224.0.0.0 bis 239.255.255.255 verwendet [Deering 89]. In einem verbindungsorientierten ATM-Netz müssen "Multicasts" über Punkt-zu-Mehrpunkt-Verbindungen realisiert werden. Der entscheidende Unterschied zu "Multicasts" in herkömmlichen Netzen ist, daß der Sender die Adressen aller Empfänger benötigt, um die Punkt-zu-Mehrpunkt-Verbindung zu allen Empfängern aufzubauen. Mit dem "Multicast Address Resolution Server" (MARS) wird in [Armitage 96] ein Server spezifiziert, bei dem sich alle Teilnehmer registrieren müssen, wenn sie den Verkehr zu einer ausgewählten "Multicast"-Adresse empfangen wollen. Der MARS ist konzeptionell eine Erweiterung des ATMARP-Servers (s. Abschnitt 2.4.2). Beim MARS wird für jede "Multicast"-Adresse eine Liste mit den ATM-Adressen aller aktuellen Teilnehmer geführt, die den Verkehr zu dieser "Multicast"-Adresse erhalten wollen. Um die Teilnehmer über hinzukommende oder abgehende Stationen zu informieren, unterhält der MARS eine Punkt-zu-Mehrpunkt-Verbindung zu allen Teilnehmern. Dies ermöglicht jedem Teilnehmer das Unterhalten einer Punkt-zu-Mehrpunkt-Verbindung zu allen anderen Teilnehmern. Da jedes Endsystem eine Verbindung zu jedem anderen Endsystem unterhält wird dies auch als "Cluster" bezeichnet. Wie bei CLIP beschränkt sich der Einsatz von MARS auf ein logische IP-Subnetz (LIS). Für "Multicasts" in andere IP-Subnetze werden "Multicast Router" benötigt. Der Aufbau von direkten virtuellen Verbindungen in andere IP-Subnetze hinein wird bisher explizit ausgeschlossen.

2.4.5 "Label-Switching"

Wie bereits dargestellt gibt es beim NHRP zwei wichtige Einschränkungen. Einerseits ist der Einsatz heute nur zwischen Endsystemen sowie zwischen Endsystemen und Routern erlaubt. Der Einsatz zwischen Routern kann zu "Routing-Loops" führen und ist daher explizit ausgenommen. Die zweite Einschränkung ergibt sich daraus, daß praktisch nur ein Endsystem mit Kenntnis der Kommunikationsanforderungen der verwendeten Applikation entscheiden kann, ob

es sinnvoll ist, eine direkte virtuelle Verbindung zum Empfänger aufzubauen. Für einen Router ist es hingegen unmöglich herauszufinden, ob zukünftig weitere Dateneinheiten zum selben Ziel folgen werden und somit der Aufbau einer direkten virtuellen Verbindung lohnen könnte. Entgegen der anfänglichen Euphorie ist heute nicht mehr damit zu rechnen, daß die Mehrzahl der Endgeräte direkt an ein ATM-Netz angeschlossen werden wird, wie dies für den Einsatz von NHRP notwendig wäre. Es zeichnet sich stattdessen ab, daß ATM primär im Backbone-Bereich eingesetzt wird, während die Mehrzahl der Endsysteme kostengünstig über Fast-Ethernet angeschlossen wird. Daher ist es notwendig Lösungen zu finden, die zwischen Routern eingesetzt werden können und die keine Endgeräte erfordern, die direkt an das ATM-Netz angeschlossen sind. Das "Label-Switching" soll hierfür eine Lösung bieten. Auf der Basis desselben Konzeptes sind eine Reihe von Ansätzen entwickelt worden, wie das "IP-Switching" [Newman et al. 97a, Newman et al. 97b], das "Multiprotocol Label Switching" (MPLS) [Callon et al. 97, Rosen et al. 98] und das "Cisco Tag Switching" [Rekhter et al. 97]. Nachfolgend wird kurz das "IP-Switching" skizziert.

Die Basis für diesen Ansatz bildet die Erkennung eines Datenflusses. Ein Datenfluß ("Flow") ist definiert als eine Folge von Datagrammen, die von einer beliebig komplexen "Routing"-Funktion identisch behandelt werden. Der IP-Switch [Newman et al. 97a] ist ein Router kombiniert mit einem ATM-Switch, wobei die Wegwahlentscheidung des Routers in der Switch-Hardware gespeichert ("cached") werden kann. Der IP-Switch "routet" zunächst die eingehenden IP-Datagramme zwischen den einzelnen Schnittstellen. Dabei untersucht der Router die übertragenen Dateneinheiten mit dem Ziel der "Flow"-Erkennung. Für einen identifizierten Datenfluß richtet der IP-Switch einen virtuellen Kanal ein, der von derselben Eingangsschnittstelle zum Router führt. Dem zum Sender des Datenflusses hin gelegenen Router ("upstream Router") wird daraufhin in einer "Redirection"-Nachricht das VPI/VCI-Paar mitgeteilt, das den eingerichteten virtuellen Kanal identifiziert. Wenn der "upstream Router" die "Redirection"-Nachricht akzeptiert, indem er beginnt, Dateneinheiten über diesen virtuellen Kanal zu übertragen, ändert er das Datagrammformat für diesen Fluß. Bei den über diesen virtuellen Kanal übertragenen IP-Dateneinheiten werden die im "Flow" definierten Header-Informationen entfernt [Newman et al. 96]. Dieses Entfernen der Header-Informationen ist möglich, da die Informationen bei jeder Dateneinheit des "Flows" per Definition gleich sein muß. Erst beim letzten IP-Switch, bei dem der "Flow" endet, werden die Header-Informationen wieder hinzugefügt. Diese Maßnahme ist aus Sicherheitsgründen eingeführt worden, da sonst die Gefahr besteht, daß ein Angreifer auf einer für einen "Flow" eingerichteten, virtuellen Verbindung auch Dateneinheiten mit anderen Header-Informationen überträgt, um dadurch Filterregeln in einem Router zu umgehen.

Effizient wird dieses Verfahren für den IP-Switch erst dann, wenn auch der zum Empfänger des "Flows" hin gelegene "downstream Router" den "Flow" erkennt

und seinerseits eine "Redirection"-Nachricht sendet. Der IP-Switch kann jetzt den eigenen virtuellen Kanal für diesen Flow mit dem in der "Redirection"-Nachricht angegebenen VPI/VCI-Paar verbinden. Die weitere Verarbeitung dieses "Flows" beschränkt sich hiernach auf das "Forwarding" der Zellen zwischen diesen beiden virtuellen Kanälen. Dieses "Forwarding" erfolgt ausschließlich im Switch, der Router ist ab sofort an diesem "Flow" nicht mehr beteiligt.

Für die Erkennung eines "Flows" ist keine Unterstützung durch Endsysteme notwendig. Stattdessen entscheiden die Router mit Hilfe einer Heuristik, ob ein "Flow" vorliegt, für den eine virtuelle Verbindung geschaltet werden sollte.

Ein wichtiger Unterschied zu den bereits vorgestellten anderen Konzepten zur Übertragung von IP-Dateneinheiten über ein ATM-Netz ist der Verzicht auf die sonst bei ATM verwendeten Signalisierungsprotokolle (PNNI, UNI). Das sehr viel einfachere Protokoll, das im wesentlichen auf den "Redirection"-Nachrichten basiert, reicht aus, um die benötigte Funktionalität zu implementieren. Dadurch wird die Duplizierung der Funktionalität in IP und ATM vermieden, was eine einfachere Implementation ermöglicht. Zudem wird hierdurch eine stärkere Integration von IP mit dem ATM-Netz erreicht, als es zum Beispiel bei NHRP oder MPOA möglich ist. Dadurch wird auch eine Vermeidung der Probleme beim "Routing" erreicht, die sich beim Einsatz von NHRP zwischen Routern durch "Routing-Loops" bemerkbar machen.

Beim "Label-Switching" reduzieren sich die Aufgaben der IP-Switches auf das Identifizieren von "Flows" und das Einrichten von virtuellen Verbindungen. Ist die virtuelle Verbindung eingerichtet, kann das Weiterleiten effizient in der Hardware des Switches erledigt werden. Nur noch die Router in den IP-Switches an den beiden Enden der virtuellen Verbindung müssen jede einzelne Dateneinheit bearbeiten. Hier fällt sogar eine etwas höhere Last an, da die den "Flow" definierenden Header-Informationen entfernt bzw. wieder hinzugefügt werden müssen.

Interessant ist dieses Konzept nicht zuletzt, da Sicherheitsgesichtspunkte bereits frühzeitig berücksichtigt wurden. Durch das Entfernen der Header-Informationen vor der Übertragung über eine für einen "Flow" eingerichtete virtuelle Verbindung wird die Nutzung dieses Ansatzes auch über eine Packet Screen hinweg möglich. Bei NHRP muß die als "Short-Cut" eingerichtete virtuelle Verbindung demgegenüber vor der Packet Screen enden, damit eine Filterung durch die Packet Screen durchgeführt werden kann. Beim "IP-Switching" kann eine Packet Screen in einem IP-Switch demgegenüber dem Aufbau einer virtuellen Verbindung zustimmen, wenn der "Flow" den Regeln der Packet Screen genügt. Hierdurch wird zudem eine Verringerung des Filterungsaufwands der Packet Screen erreicht, da Dateneinheiten, die über die virtuelle Verbindung des "Flows" übertragen werden, nicht mehr gefiltert werden müssen, so daß sich die Anzahl der Datagramme, die von der Packet Screen gefiltert werden müssen, stark verringert. Da "IP-Switching" nur zwischen Routern angewendet wird, sind Firewall-

Konzepte auf Anwendungsebene, die auf Endsystemen eingesetzt werden müssen (Proxies), vom "IP-Switching" nicht direkt betroffen. Die Anbindung von direkt an ein ATM-Netz angeschlossenen Endsystemen muß beispielsweise über CLIP erfolgen.

2.4.6 "LAN Emulation" (LANE)

Die vom ATM-Forum entwickelte "LAN Emulation" (LANE) [ATM Forum 95a, ATM Forum 97c] ist ein weiterer Ansatz, um Dateneinheiten herkömmlicher Netze auf ATM-Netzen übertragen zu können. Im Gegensatz zu CLIP werden hier nicht Dateneinheiten auf Schicht 3 des ISO-OSI Modells übertragen. Stattdessen ermöglicht LANE die Übertragung von Dateneinheiten der Schicht 2. LANE unterstützt zwei Rahmenformate: Ethernet (IEEE 802.3) und Token-Ring (IEEE 802.5) [ATM Forum 97c, S. 51f], so daß alle auf diesen beiden Netzen einsetzbaren Protokolle unterstützt werden können. Dabei können auf demselben ATM-Netz parallel mehrere "virtuelle LANs" emuliert werden. Die "LAN Emulation" basiert auf einem Client-Server-Modell, deren Komponenten im folgenden vorgestellt werden.

"LAN Emulation Client" (LEC): Der LEC muß in Endsystemen vorhanden sein, die direkt an ein ATM-Netz angeschlossen sind und über LANE kommunizieren wollen. Bevor die Kommunikation mit anderen Systemen des emulierten LANs möglich wird, muß sich ein LEC zunächst bei den folgenden drei Servern registrieren.

"LAN Emulation Configuration Server" (LECS): Ein LEC muß sich im ersten Schritt beim LECS anmelden. Der LECS entscheidet, zu welchem emulierten LAN der LEC gehört. Dem LEC wird vom LECS daraufhin die ATM-Adresse des für das emulierte LAN zuständigen "LAN Emulation Servers" mitgeteilt.

"LAN Emulation Server" (LES): Der LES hat bei der "LAN Emulation" eine ähnliche Aufgabe wie der ATMARP-Server bei CLIP. Beim LES müssen sich alle LECs eines emulierten LANs mit ihrer MAC-Adresse und ihrer ATM-Adresse registrieren. Repräsentiert ein LEC mehrere Endgeräte, so muß der LEC auch alle MAC-Adressen dieser Endgeräte registrieren. Dies ist zum Beispiel bei einem sogenannten "Edge-Device" der Fall, das ein Ethernet mit einem ATM-Netz verbindet. Nach der Registrierung kann der LEC beim LES über das "LAN Emulation Address Resolution Protocol" (LE-ARP) die ATM-Adressen anderer LECs im selben emulierten LAN erfragen. Wie auch bei CLIP werden für die Übertragung von Daten zwischen LECs virtuelle Verbindungen aufgebaut.

"Broadcast and Unknown Server" (BUS): Die in ATM-Netzen nicht vorhandene "Broadcast"-Funktionalität wird vom BUS emuliert. Zusätzlich wird der BUS für Dateneinheiten benutzt, deren MAC-Adressen nicht beim LES registriert sind. Diese Dateneinheiten an unbekannte ("unknown") MAC-Adressen werden wie "Broadcasts" an alle Stationen des emulierten LANs übermittelt.

2.4.7 "Multi-Protocol over ATM" (MPOA)

Die "LAN Emulation" hat wie CLIP den Nachteil, daß eine Kommunikation nur zwischen Systemen desselben virtuellen LANs möglich ist. Um über ein Schicht-3-Protokoll wie IP mit Systemen anderer virtueller LANs zu kommunizieren, müssen die Dateneinheiten auch dann einen Weg über einen Router nehmen, wenn beide virtuellen LANs auf demselben ATM-Netz basieren. "Multi-Protocol over ATM" (MPOA) [ATM Forum 97d] ergänzt LANE, um direkte virtuelle Verbindungen ("Short-Cuts") zwischen LECs verschiedener virtueller LANs zu ermöglichen. Zur Ermittlung der ATM-Adressen von LECs anderer virtueller LANs wird NHRP (s. Abschnitt 2.4.3) eingesetzt.

Damit direkt an das ATM-Netz angeschlossene Hosts und "Edge Devices" (vgl. LANE) über MPOA Verbindungen in andere Netze aufbauen können, werden sie zusätzlich zu einem LEC um einen "MPOA Client" (MPC) ergänzt. Der "MPOA Server" (MPS) wird zusammen mit einem "Next Hop Server" (vgl. NHRP) in einen Router integriert. Wie bei der "LAN Emulation" werden Datagramme zu Systemen anderer virtueller LANs zunächst mit Hilfe des Routers versendet. Innerhalb der über den Router gesendeten Datagramme identifiziert der MPC einzelne, länger andauernde Datenströme ("Flows"), für die dann direkte virtuelle Verbindungen zu den Empfängern bzw. den zuständigen "Edge Devices" aufgebaut werden. Die ATM-Adresse des Hosts bzw. des zuständigen "Edge Devices" erhält der MPC über eine MPOA-Anfrage ("MPOA Query") an den MPS. Der MPS ermittelt die gesuchte ATM-Adresse über NHRP wie im Abschnitt 2.4.3 beschrieben. Für die Datenübertragung über eine zwischen unterschiedlichen virtuellen Netzen aufgebaute virtuelle Verbindung kann nicht mehr das bei LANE verwendete Rahmenformat verwendet werden, da die bei LANE enthaltenen Ethernet- bzw. Token-Ring-Header nur innerhalb desselben virtuellen LANs benutzt werden können. Die Protokoll-Header der Schicht 2 werden daher vom MPC vor dem Senden entfernt. Der MPC am anderen Ende der virtuellen Verbindung fügt wieder einen Schicht-2-Header hinzu, der die Systeme in dem dortigen virtuellen LAN adressiert.

MPOA ist noch eine sehr neue Entwicklung. Die Spezifikation des ATM-Forums [ATM Forum 97d] erschien erst 1997, so daß MPOA bisher noch keine nennenswerte Verbreitung finden konnte.

2.4.8 Unterschiede zwischen ATM und herkömmlichen Netzen

Für die Erarbeitung von Firewall-Konzepten in ATM-Netzen und für die Identifikation von Problembereichen beim Einsatz klassischer Firewall-Techniken in ATM-Netzen ist zunächst die Identifikation von ATM-spezifischen Eigenschaften im Vergleich zu klassischen "Broadcast" Netzen wie Ethernet — oft als "Legacy"-Netzwerke bezeichnet — notwendig.[37] Die wichtigsten Unterscheidungsmerkmale von ATM zu herkömmlichen (IP-)Netzen sind in der folgenden Liste aufgeführt.

Zellen statt Rahmen: Eine Datenübertragung findet bei ATM über Zellen statt. Zellen haben im Gegensatz zu Rahmen anderer Protokolle eine feste Länge von 53 Byte. Daher können Zellen im Vergleich zu Rahmen nur relativ kleine Datenmengen aufnehmen, so daß sich selbst kürzere Nachrichten auf mehrere Zellen verteilen. Insbesondere gelingt es praktisch nicht,[38] eine komplette IP-Dateneinheiten in einer Zelle unterzubringen, da bereits die Header bzw. Trailer von TCP, IP, SNAP und AAL-5 zusammen eine Länge von 56 Byte erreichen (s. Tabelle 4.2, Seite 229).[39] Die Aufteilung stellt ein Problem dar für Anwendungen, die auf Rahmen zugreifen, um die darin enthaltenen IP-Datagramme auszuwerten. Betroffen sind hiervon Netzwerk-"Sniffer" und Packet Screens, die bei ATM zunächst die empfangenen Zellen zu IP-Datagrammen reassemblieren müssen und nicht direkt auf den empfangenen Zellen arbeiten können.

ATM ist verbindungsorientiert: Bei herkömmlichen IP-Netzen werden einzelne Dateneinheiten mit Adreßinformationen versehen und auf den Weg zum Empfänger geschickt. Die geringe Header-Länge einer ATM-Zelle von nur 5 Byte reicht dagegen nicht aus, um wie in einem IP-Header global einheitliche Adreßinformationen unterzubringen. Bevor mit dem Senden von Dateneinheiten begonnen werden kann, muß zuerst eine Verbindung zwischen Sender und Empfänger aufgebaut werden. In den übertragenen Zellen sind

---

[37] Erfahrungen mit herkömmlichen IP-Netzen werden vorausgesetzt. Für einen Einstieg in die Techniken von IP-Netzen wird [Comer 91] empfohlen.

[38] Beim Einsatz von UDP als Transportprotokoll werden 44 Byte Header-Informationen versendet, so daß lediglich 4 Byte in der Zelle für Nutzdaten übrig bleiben. Selbst ICMP-Nachrichten überschreiten in der Regel die 48 Byte, die sich in einer Zelle unterbringen lassen.

[39] Dies bedeutet, daß selbst TCP-Segmente, die keine Nutzdaten enthalten, auf 2 Zellen aufgeteilt werden müssen. TCP-Segmente ohne Nutzdaten werden etwa beim Verbindungsaufbau gesendet (SYN und SYN-ACK) sowie während einer Verbindung als Empfangsbestätigung, wenn die Empfangsbestätigung nicht "Huckepack" mit Nutzdaten übertragen werden kann.

anschließend nur noch Multiplex-Informationen[40] enthalten, die es ermöglichen, einzelne Zellen aus einem Zellstrom den Verbindungen zuzuordnen.

Die Folgen des verbindungsorientierten Ansatzes sind:

- Kommunizierende Rechner können nicht ohne Kenntnis der beim Verbindungsaufbau ausgetauschten Informationen nur anhand einer einzelnen Zelle identifiziert werden.
- Länger andauernde Verbindungen und Verbindungen, auf denen große Datenmengen übertragen werden, profitieren vom Verbindungsaufbau, weil der Aufwand der Wegfindung nur beim Verbindungsaufbau anfällt. Kurze Verbindungen, bei denen nur eine kurze Nachricht ausgetauscht werden soll, sind jedoch durch die Verbindungsaufbauzeiten benachteiligt. Protokolle bei denen verbindungslose, datagrammorientierte (UDP-)Nachrichten ausreichen (etwa DNS), erhalten bei ATM einen höheren "Overhead".

Trennung von Wegfindung und Datenübertragung: Bei IP wird das "Routing" (Wegfindung) für die Übertragung jeder Dateneinheit durchgeführt. Ein Router, der eine Dateneinheit auf dem Weg zum Empfänger erhält, trifft anhand der Adreßinformationen der Dateneinheit die Entscheidung, über welche Leitung der Empfänger erreicht werden kann. Diese Entscheidung kann bereits bei der nächsten Dateneinheit anders ausfallen.

Bei ATM findet das "Routing" beim Verbindungsaufbau statt. Das bedeutet, daß der Sender vor dem Senden der ersten Dateneinheit bereits weiß, daß der Empfänger erreichbar ist.[41] Die anschließend gesendeten Daten nutzen immer denselben einmal gefundenen Weg zum Empfänger. Wird dieser Weg unterbrochen, können die Zellen keinen alternativen Weg verwenden.

Skalierbarkeit: ATM ist sehr leicht skalierbar. Ohne Änderungen an den zugrundeliegenden Konzepten sind Übertragungsraten von 2,4 GBit/s und mehr (vgl. Tab. 4.1, Seite 228) erreichbar. Netzkomponenten werden schnell zum Engpaß, wenn sie nicht ähnlich skalierbar sind wie die angeschlossenen ATM-Netze. Firewalls sind bisher nur schlecht skalierbar.

Kein "Shared-Medium": "Legacy"-LANs verwenden ein "Shared Medium". Erhält ein Netzknoten Zugriff auf das Medium, kann dieser die gesamten

[40] Es handelt sich hierbei um VPI und VCI. (vgl. S. 48)

[41] Bei IP-Netzen werden die Dateneinheiten auch ohne Kenntnis der Erreichbarkeit des Empfängers gesendet. Wenn ein Router auf dem (potentiellen) Weg zum Empfänger das IP-Datagramm weder an den Empfänger noch an einen geeigneten Router weiterleiten kann, wird das Datagramm vernichtet und der Sender mit einem ICMP-"Destination Unreachable" darüber informiert.

momentan verfügbaren Ressourcen verwenden. Andere Knoten werden solange blockiert, bis der Sender (kurzzeitig) das Netz wieder freigibt.

Bei ATM werden die Zugriffe auf das Netz kontrolliert. Ein Netzknoten kann daher nicht durch Überschreiten einer vorher ausgehandelten maximalen Übertragungsrate andere Netzknoten blockieren. Dadurch kann gewährleistet werden, daß jedem Netzknoten auch dann die ausgehandelten Übertragungsraten zur Verfügung stehen, wenn das Netz von anderen Knoten intensiv genutzt wird.

ATM-Netze bestehen aus aktiven Komponenten: Klassische Ethernets werden aus Coaxial-Kabeln aufgebaut. Auf diese passive Komponente haben alle angeschlossenen Stationen unkontrollierbaren Zugriff. ATM-Netze benötigen einen Switch als aktive Komponente, um ein Netz aufzubauen. Switches ermöglichen eine bessere Kontrolle der Datenflüsse und der Zugriffe. Unter der Voraussetzung, daß der Switch unter der Kontrolle des Administrators ist, können die angeschlossenen Hosts sehr genau kontrolliert werden.[42]

Ressource-Reservierung und -Zuteilung: Bei "Legacy"-LANs werden Ressourcen mit zusätzlichen, neuen Benutzern geteilt, die bereits vorhandenen Benutzer müssen sich die Bandbreite mit den neuen Benutzern teilen. Bei ATM wird ein Zugriff eines neuen Benutzers mitunter abgelehnt, wenn keine ausreichenden Ressourcen zur Verfügung stehen. Bereits registrierte Benutzer erhalten weiterhin die anfangs garantierte Bandbreite und brauchen daher keine Degradierung ihrer Kommunikationsqualität zu befürchten.

Kontrolle des Netzzugriffes: Bei "Legacy"-LANs kann ein Knoten bzw. eine Anwendung auf das Netz zugreifen, wenn es gerade frei ist. Die Anwendung erhält also mitunter keinen Zugriff, wenn das Netz gerade von einem anderen Knoten blockiert wird. Sobald sie aber Zugriff erhält, kann sie die gesamte zur Verfügung stehende Leistung verwenden. Bei ATM müssen Anwendungen die Charakteristik ihres Datenverkehrs beim Verbindungsaufbau beschreiben und sich anschließend an diese Beschreibung halten.

Die ATM Zugriffskontrolle wird realisiert als Kombination aus Überwachungsfunktion in den Switches und "Traffic Shaping" in den Endsystemen. Der Switch hat eine bestimmte Kapazität zur Verfügung, die er per "Vertrag" einer Verbindung zur Verfügung stellen kann. Das Endsystem

[42] Switches können auch in Ethernet-Umgebungen eingesetzt werden. Switches werden in Ethernet-Umgebungen in erster Linie zur Performanzsteigerung eingesetzt. Zusätzlich spricht aber auch bei Ethernets die bessere Kontrolle der angeschlossenen Stationen für den Austausch von einer "Shared-Medium"-Verkabelung gegen eine strukturierte Verkabelung mit Switches.

garantiert in dem Vertrag im Gegenzug, daß es die ausgehandelte Zellrate nicht überschreiten wird. Erzeugt eine Anwendung eine höhere Zellrate als ausgehandelt, versucht das Endsystem[43] über "Traffic Shaping" die Zellrate einzuhalten. Der Switch kontrolliert die Einhaltung der ausgehandelten Zellrate und vernichtet Zellen des Endsystems, falls dies notwendig wird. Die Aushandlung dieses Vertrages geschieht beim Aufbau einer Verbindung. [Danthine et al. 95]

Das Problem bei derzeitigen ATM-Lösungen ist jedoch, daß bei "Classical IP" und LANE die ATM-Schicht vor den Anwendungen verborgen wird und daher den Anwendungen diese Informationen nicht zur Verfügung stehen. Erst "Native"-ATM-Anwendungen können von der Reservierung von Betriebsmitteln wirklich profitieren.

Keine "Broadcast" Unterstützung: ATM unterstützt kein "Broadcast". Dort wo die Funktionalität benötigt wird, muß sie emuliert werden. Dies wird durch einen zentralen Server realisiert, bei dem sich Stationen registrieren, die "Broadcasts" empfangen möchten. Nachrichten an die "Broadcast"-Adresse müssen dann an alle registrierten Adressen gesendet werden. Der BUS bei der "LAN Emulation" (s. Abschnitt 2.4.6) ist ein Beispiel für einen zentralen "Broadcast"-Server. Ein offensichtliches Problem ist die Abhängigkeit von einer zentralen Komponente im Fehlerfall.

ATM ist in LAN und WAN einsetzbar: Klassische LAN-Protokolle wie Ethernet eignen sich nicht für die Datenkommunikation im WAN. Umgekehrt sind WAN-Techniken meist ungeeignet für LANs. Der Einsatz unterschiedlicher Technologien im LAN und WAN erforderten am Übergang zwischen den beiden Netzen Komponenten wie einen Router und ein Protokoll wie IP, das unabhängig vom jeweiligen Netz eine einheitliche Schnittstelle zur Verfügung stellt. ATM eignet sich demgegenüber für den Einsatz in LANs und WANs. Dadurch kann der bisher an der Grenze zwischen LAN und WAN notwendige Übergang entfallen, und es entsteht ein Netz, das aus LAN- und WAN-Segmenten besteht, die sich nahtlos aneinanderfügen.

ATM ist Dienste-integrierend: ATM eignet sich gleichermaßen für die Übertragung von Daten, Video und Telefonie. Dadurch entsteht eine Grundlage für flexible Nutzung eines gemeinsamen Netzwerkes für unterschiedliche Dienste.

[43] Das "Traffic Shaping" wird üblicherweise in den ATM-Adapterkarten implementiert.

2.5 Neue Herausforderungen für Firewalls

In diesem Kapitel wurde der heutige Stand der Entwicklung vorgestellt. Durch die Vernetzung von Computern, insbesondere über das erfolgreiche Internet, ist es heute viel einfacher, fremde Systeme anzugreifen. Die verschiedenen netzbasierten Angriffe wurden im Abschnitt 2.1 unterschiedlichen Kategorien zugeordnet. Die heute beobachteten Angriffe nutzen meist Implementations- und Konfigurationsfehler aus. In der letzten Zeit werden jedoch zunehmend auch konzeptionelle Schwächen wie fehlende Authentizität, Integrität und Vertraulichkeit übertragener Daten ausgenutzt. Gegen diese konzeptionellen Schwächen vieler Protokolle werden zunehmend kryptographische Verfahren eingesetzt, über die Abschnitt 2.2 einen Überblick gab. Gegen Implementations- und Konfigurationsfehler hilft hingegen nur eine Zugriffskontrolle, die unberechtigte Zugriffe blockiert. Als Zugriffskontrollmechanismus in Netzen haben sich Firewalls durchgesetzt. Firewall-Komponenten und die wichtigsten Firewall-Architekturen wurden im Abschnitt 2.3 vorgestellt. Im Abschnitt 2.4 wurde schließlich ATM als neue Technologie für Hochgeschwindigkeitsnetze vorgestellt, die sich stark von herkömmlichen Netzen unterscheidet.

Firewalls haben sich in den letzten Jahren nur noch geringfügig verändert, etwa durch zusätzliche oder verbesserte Proxies sowie durch die Entwicklung graphischer Benutzerschnittstellen zur Vereinfachung der Konfiguration von Firewalls. Aus dem Spannungsfeld der Bereiche, die in den Abschnitten dieses Kapitels diskutiert wurden, ergeben sich eine Reihe von Herausforderungen, die die Weiterentwicklung heutiger Firewall-Techniken erforderlich machen.

Kryptographische Protokolle: Der Einsatz kryptographischer Verfahren zur Absicherung der Kommunikation steht heute noch am Anfang. Firewalls können auf der einen Seite von diesen Verfahren profitieren, da sie beispielsweise eine bessere Authentisierung der Zugriffe ermöglichen. Auf der anderen Seite ist es bei verschlüsselten Daten für einen Firewall nicht möglich, eine Zugriffskontrolle durchzuführen. Die Wechselwirkungen zwischen kryptographischen Verfahren und Firewalls wurden in einer Studie des DFN-FWLs erarbeitet [Benecke et al. 98c] und werden in dieser Arbeit nicht weitergehend behandelt.

ATM: Da sich ATM stark von herkömmlichen Netzen unterscheidet, eignen sich heutige Firewalls nur unzureichend für die Absicherung von ATM-Netzen. Für einen Einsatz in ATM-Netzen werden daher Änderungen und Ergänzungen an Firewalls erforderlich. ATM ermöglicht jedoch auch neue Möglichkeiten für eine erweiterte Zugriffskontrolle, die in herkömmlichen Netzen nicht möglich war. Die Aspekte der Integration von Firewalls in ATM-Netze werden im Kapitel 3 ausführlich diskutiert.

Hochgeschwindigkeitsnetze: Unabhängig von den technischen Herausforderungen, die der Einsatz von ATM zur Folge hat, stellen Hochgeschwindigkeitsnetze wie ATM, Fast- und Gigabit-Ethernet ein Problem für Firewalls dar, weil mit dem Firewall bewußt ein Engpaß geschaffen wird, um hier die Kommunikation überwachen zu können. Bisher waren Firewalls ausreichend leistungsfähig, um trotz dieses Engpasses eine akzeptable Performanz gewährleisten zu können. Die Durchsatzsteigerung bei Hochgeschwindigkeitsnetzen und insbesondere die leichte Skalierbarkeit von ATM drohen Firewalls zukünftig zu einem Performanzengpaß werden zu lassen. Bisher lagen noch keine systematischen Untersuchungen der Leistungsgrenzen herkömmlicher Firewalls vor. Im Kapitel 4 wird die Performanz von Firewalls in einem ATM-Netz untersucht. Im Kapitel 5 wird anschließend untersucht, wie weit Parallelisierungskonzepte für die Steigerung der Performanz von Firewalls eingesetzt werden können.

3 Firewalls in ATM-Netzen

Inhalt

3.1 Firewall-Aspekte von ATM-Netzen 73
3.2 Neue Schwachstellen in ATM-basierten Netzen 80
3.2.1 Schwachstellen von ATM-Netzen 80
3.2.1.1 Angriffe auf die Vertraulichkeit 81
3.2.1.2 Angriffe auf die Authentizität und Integrität 83
3.2.1.3 Angriffe auf die Verfügbarkeit 85
3.2.1.4 Implementations- und Konfigurationsfehler 87
3.2.2 Schwachstellen von CLIP-Netzen 88
3.2.2.1 Angriffe auf die Vertraulichkeit 89
3.2.2.2 Angriffe auf die Authentizität und Integrität 89
3.2.2.3 Angriffe auf die Verfügbarkeit 93
3.2.2.4 Implementations- und Konfigurationsfehler 94
3.2.3 Schwachstellen bei NHRP 94
3.2.3.1 Angriffe auf die Vertraulichkeit 94
3.2.3.2 Angriffe auf die Authentizität und Integrität 95
3.2.3.3 Angriffe auf die Verfügbarkeit 96
3.2.3.4 Implementations- und Konfigurationsfehler 97
3.2.4 Schwachstellen bei MARS 97
3.2.5 Schwachstellen von “LAN Emulation” 98
3.2.5.1 Angriffe auf die Authentizität und Integrität 98
3.2.5.2 Angriffe auf die Verfügbarkeit 98
3.2.6 Schwachstellen bei MPOA 99
3.3 ATM-spezifische Firewall-Komponenten 99
3.3.1 Kontrolle der Signalisierung 100
3.3.1.1 Zugriffskontrolle über einfache Signalisierungsfilter . 103

3.3.1.2 Erweiterte Möglichkeiten eines Signalisierungskontrollers . . . 124
3.3.1.3 Integration von Sicherheitskomponenten über NHRP 138
3.3.1.4 Fazit: Kontrolle der Signalisierung . . . 141
3.3.2 Packet Screens für ATM-Netze . . . 142
3.3.2.1 Klassische Packet Screens für CLIP-Netze . . . 142
3.3.2.2 Packet Screens auf ATM-Schicht . . . 145
3.3.2.3 Granularität der Zugriffskontrolle von Packet Screens in ATM-Netzen . . . 155
3.3.2.4 Packet Screen und Paket-Klassifikation . . . 155
3.3.2.5 Fazit: Packet Screens in ATM-Netzen . . . 156
3.3.3 Kontrollkomponenten auf Anwendungsebene . . . 156
3.3.3.1 Proxies und Anwendungs-Gateways in ATM-basierten IP-Netzen . . . 157
3.3.3.2 "Native"-ATM-Proxies . . . 158
3.3.3.3 Transparente Proxies in NHRP-Netzen . . . 166
3.3.3.4 Fazit: Kontrollkomponenten auf Anwendungsebene . 177
3.3.4 Kontrolle des Netzwerkmanagements . . . 178
3.4 Firewall-Architekturen für ATM-Netze . . . 179
3.4.1 Firewall-Architekturen für CLIP-Netze . . . 180
3.4.2 Einsatz virtueller Netze für Firewall-Architekturen . . . 182
3.4.3 Kombination von Bastion und Signalisierungskontroller . . . 187
3.4.4 Asymmetrische Kontrolle von Verbindungen . . . 190
3.4.5 Zentrale Kontrolle und virtuelle Netze . . . 192
3.4.6 Einsatz virtueller Schnittstellen . . . 200
3.4.7 "Native"-ATM-Firewalls . . . 202
3.5 Zukünftige Firewall-Lösungen für ATM-Netze . . . 204

Seit der Entwicklung der ersten Firewalls Anfang der 90er hat es eine stetige Weiterentwicklung in kleinen Schritten gegeben. Da es hierbei keine bahnbrechenden Innovationen gab, haben sich die grundlegenden Firewall-Konzepte über die Jahre kaum verändert. Heute sind Firewalls ein gut verstandener Zugriffskontrollmechanismus zur Absicherung von IP-Netzen.

ATM ist eine neue Netztechnologie, die sich wesentlich von den herkömmlichen Netztechnologien unterscheidet. Hauptaufgabe bleibt aber vorerst auch bei ATM die Übertragung von IP-Dateneinheiten, wie sie bisher schon in herkömmlichen

Netzen genutzt wurden. ATM wird dabei hauptsächlich im WAN/MAN und Backbone eingesetzt. Für einen flächendeckenden Einsatz im LAN ist die ATM-Technologie zu teuer und wird sich daher voraussichtlich nicht gegen Ethernet durchsetzen können. Da im Haupteinsatzbereich von Firewalls am Übergang zum Unternehmensnetz bereits heute auf beiden Seiten des Firewalls – dem WAN und dem Backbone – ATM-Technologie eingesetzt wird, müssen Lösungen für den Einsatz von Firewalls in ATM-Netzen gefunden werden.

Die Folgen, die ATM für Firewalls hat, werden im Abschnitt 3.1 einleitend vorgestellt. Im Abschnitt 3.2 folgt eine Untersuchung der neuen Schwachstellen, die durch den Einsatz von ATM entstehen. In den Abschnitten 3.3 und 3.4 werden anschließend neue Firewall-Komponenten und Firewall-Architekturen für ATM-Netze untersucht. Dabei werden nur die ATM-spezifischen Aspekte der Absicherung eines Netzes durch Firewalls behandelt. Nicht diskutiert werden allgemeine Fragen des Aufbaus von Firewalls, wie etwa die Entwicklung von Sicherheitsrichtlinien, die Absicherung des Firewalls und sichere Realisierung von Proxies für ausgewählte Protokolle. Diese auch für Firewalls in herkömmlichen Netzen zu beachtenden Punkte sind ausführlich in einschlägiger Literatur diskutiert [Chapman et al. 95, Cheswick et al. 94, Ellermann 93] und sollen daher an dieser Stelle nicht wiederholt werden.

3.1 Firewall-Aspekte von ATM-Netzen

Die im Abschnitt 2.4.8 identifizierten ATM-spezifischen Eigenschaften bilden die Grundlage für Untersuchungen zum Einsatz von Firewalls in ATM-Netzen. In diesem Abschnitt werden die für Firewalls relevanten Aspekte von ATM-Netzen einleitend vorgestellt und das weitere Vorgehen umrissen. Ziel ist es dabei, den gesamten Problembereich aufzuzeigen. Eine detaillierte Analyse der aufgezeigten Probleme erfolgt in späteren Abschnitten bzw. in folgenden Kapiteln.

Der Einsatz von Firewalls in ATM-Netzen birgt noch viele zu untersuchende Aspekte:

Performanzprobleme: Um eine Kontrolle der Kommunikation zwischen Netzen zu ermöglichen, werden Firewalls als beabsichtigte Engpässe zwischen Netzen eingefügt. In ATM-Netzen und anderen Hochgeschwindigkeitsnetzen kann ein Firewall daher leicht auch bei der Performanz zum Engpaß werden.

Die ersten Firewalls wurden zu Zeiten entwickelt, als 9,6 KBit/s üblich und 2 MBit/s Internet-Anschlüsse noch wenigen, privilegierten Universitäten vorbehalten waren. Auch bei der in LANs damals üblichen Geschwindigkeit von 10 MBit/s konnten Firewalls noch eingesetzt werden, ohne zum

"Flaschenhals" zu werden. ATM-Netze haben üblicherweise Durchsätze von mindestens 155 MBit/s und lassen sich problemlos auf höhere Übertragungsraten skalieren, so daß sich bereits bestehende Performanzprobleme von Firewalls zukünftig noch verschärfen werden.

Für Firewall-Komponenten auf Anwendungsebene (Proxies und Anwendungs-Gateways) ist die Performanz der wesentliche Aspekt, da von den unteren Schichten abstrahiert wird. Durch dieses Verdecken der unteren Schichten wird es für einen Proxy auf Anwendungsebene irrelevant, ob der Verkehr über Ethernet, Fast-Ethernet, Gigabit-Ethernet oder über ATM übertragen wird. Natürlich können bei diesem Herangehen die spezifischen Vor- und Nachteile der jeweiligen Technologie nicht adäquat berücksichtigt werden.

Einfluß des Verbindungsmanagements: Ein weiterer wichtiger Aspekt ist die Dauer des Verbindungsaufbaus. Während bei den bisher eingesetzten Netzen der Durchsatz das wichtigste Performanzkriterium ist, gewinnt bei Hochgeschwindigkeitsnetzen die Verbindungsaufbauzeit eine größere Bedeutung. Dauert die Übertragung einer 100 KByte-Datei auf einer 64 KBit/s Leitung etwa 15 Sekunden, so werden bei einem Hochgeschwindigkeitsnetz mit einem Durchsatz von 622 MBit/s für die Übertragung derselben Datei nicht einmal zwei Millisekunden benötigt. Da die Verbindungsaufbauzeit von der Nachrichtenlaufzeit[1] abhängt, bleibt sie somit in Hochgeschwindigkeitsnetzen und klassischen Netzen etwa gleich. Da in Hochgeschwindigkeitsnetzen die Datenübertragungszeiten bei gleicher Nachrichtenlänge geringer werden, nehmen die durch die Nachrichtenlaufzeit bedingten Wartezeiten, etwa die Verbindungsaufbauzeit, bei Hochgeschwindigkeitsnetzen einen signifikanten Anteil an der Gesamtdauer einer Verbindung ein.

Neben dem Verbindungsaufbau gibt es weitere Fälle, bei denen in Protokollen eine Antwort des Kommunikationspartners abgewartet werden muß, bevor weitere Daten gesendet werden können. Protokolle für Hochgeschwindigkeitsnetze müssen daher so optimiert werden, daß die Anzahl der Fälle minimiert wird, bei denen auf eine Antwort des Kommunikationspartners gewartet werden muß [Peterson et al. 96, S. 438-441].

[1] Die Nachrichtenlaufzeit bezeichnet die Zeit, die zur Übertragung einer Nachricht vom Sender zum Empfänger benötigt wird. Sie setzt sich zusammen aus der Signalübertragungszeit im Medium, den Verzögerungszeiten in Netzwerkkomponenten (Switches und Router) und den Verzögerungszeiten in den Endsystemen. Selbst wenn die Verzögerungszeiten durch Netzwerkkomponenten auf einer direkten Verbindung entfallen und die Verzögerungszeiten in den Endsystemen minimiert werden, bleibt eine minimale Nachrichtenlaufzeit, die sich aus dem Quotienten aus Entfernung und Medium-abhängiger Signalgeschwindigkeit ergibt. Die Ausbreitung eines Signals im Medium kann nie die Lichtgeschwindigkeit überschreiten. Als Ausbreitungsgeschwindigkeit wird üblicherweise etwa 2/3 der Lichtgeschwindigkeit angenommen.

Der Umkehrschluß dieser Aussage heißt, daß Protokolle, die viele Anfragen und Antworten zwischen den Kommunikationspartnern erfordern, etwa wiederholt Verbindungen auf- und abbauen (HTTP) oder zur Authentisierung viele Nachrichten austauschen (SSH, SSL), in Hochgeschwindigkeitsnetzen ein schlechtes Performanzverhalten zeigen.

Firewalls erhöhen die Nachrichtenlaufzeit in mehrfacher Hinsicht:

- Bei Proxies müssen zwei Verbindungen (vom Client zum Proxy und vom Proxy zum Server) aufgebaut werden. Erst nachdem vom Firewall die Autorisierung des Clients überprüft wurde, beginnt der Verbindungsaufbau zum Server. Der beiden "Handshakes" zum Aufbau der Verbindungen können also nicht parallelisiert für beide Verbindungen gleichzeitig ausgeführt werden.
- Zur Überprüfung der Autorisierung und für die Audit-Aufzeichnungen werden die IP-Adressen der beteiligten Rechner vom Firewall in logische DNS-Namen aufgelöst.[2] Die Wartezeit für die Beantwortung der DNS-Anfragen erhöht die Nachrichtenlaufzeit zusätzlich. Insbesondere, wenn die Anfrage-Datagramme beim UDP-basierten DNS verloren gehen oder wenn zur Beantwortung von DNS-Anfragen mehrere DNS-Server befragt werden müssen, können mehrere Sekunden vergehen.
- Oft wird bei Firewalls der "Ident"-Dienst [St. Johns 93] benutzt, um Benutzer zu identifizieren. Bei einer hereinkommenden Verbindung wird zuerst eine TCP-Verbindung zurück zum Client eröffnet, um dort den Benutzernamen zu erfragen, der die ursprüngliche TCP-Verbindung aufgebaut hatte.
- Weitere (manuelle) Authentisierungsschritte und Verfahren zum Austausch kryptographischer Schlüssel erhöhen die Firewall-bedingte Nachrichtenlaufzeit weiter.

Im Firewall-Design für ATM-Netze ist es daher besonders wichtig, die Faktoren, die die Nachrichtenlaufzeit erhöhen, zu minimieren.

Intensivierung der Angriffe durch Hochgeschwindigkeitsnetze: Firewalls müssen in ATM-Netzen nicht nur hohe Durchsätze ermöglichen, um nicht zum Flaschenhals zu werden. Auch die Widerstandskraft gegen Angriffe muß die in ATM möglichen hohen Durchsätze berücksichtigen. Die hohen Datendurchsätze von ATM können nicht nur von berechtigten Nutzern verwendet werden, sondern können auch von Angreifern zur Intensivierung der Angriffe ausgenutzt werden. Insbesondere bei "Denial of Service"-Angriffen, wie zum Beispiel den "Broadcast-Ping"-Angriffen (s. Ab-

[2] Da die Authentizität von DNS-Informationen nicht überprüft werden kann, birgt dieses Vorgehen Sicherheitsrisiken.

schnitt 2.1.1.4), können "Paket-Stürme" beim Firewall eintreffen. Besonders gefährdet sind auch Dienste, bei denen bereits wenige Pakete Ressourcen-intensive Berechnungen verursachen (z. B. Schlüsselberechnungen bei Key-Management-Protokollen). Angreifer, die in einem Hochgeschwindigkeitsnetz mit einer Vielzahl gefälschter Pakete sehr viele Berechnungen von Schlüsseln verursachen können, binden so die Ressourcen des Firewalls. Da die Unterstützung von kryptographischen Verfahren zukünftig ein wichtiger Bestandteil von Firewalls sein wird [Benecke et al. 98c], werden derartige "Denial of Service"-Angriffe und deren Vermeidung eine große Rolle spielen.

Selbst wenn Angriffe vom Firewall erkannt werden, besteht noch das Problem des Audits. Das Vorgehen heutiger Firewalls, für jedes sicherheitsrelevante Ereignis einen Audit-Eintrag zu schreiben, führt in Hochgeschwindigkeitsnetzen schnell zu einer Überlastung des Firewalls durch das Audit-System. Insbesondere ist es in Hochgeschwindigkeitsnetzen nicht mehr möglich, für jedes blockierte Datagramm einen Audit-Eintrag zu erzeugen. Da ein UDP-Datagramm mit bis zu 4 Byte Nutzdaten in eine Zelle paßt (siehe Berechnung im Abschnitt 4.2.2.5), können in einem 155 MBit/s ATM-Netz über 100.000 Datagramme pro Sekunde vom Angreifer erzeugt werden.

Firewalls für ATM-Netze müssen daher auf Angriffe möglichst effizient reagieren können. "Denial of Service"-Angriffe müssen frühzeitig erkannt werden, um deren Auswirkungen begrenzen zu können. Bei der Aufzeichnung von Audit-Einträgen ist eine frühzeitige Reduktion der Audit-Daten vor der Aufzeichnung erforderlich, um die zu erwartenden großen Datenmengen verkraften zu können.

Neue Risiken durch neue Protokolle und neue Netzkomponenten: Das ATM-Konzept unterscheidet sich stark von der Technik herkömmlicher Netzwerke. Neue Komponenten wie der ATM-Switch, neue Protokolle zum Aufbau von virtuellen Verbindungen (PNNI, IISP, UNI) und zum Austausch von Managementinformationen (ILMI) sowie neue Dienste zur Unterstützung der über ein ATM-Netz genutzten Protokolle (bei CLIP, NHRP, LANE und MPOA) werden benötigt. Alle diese Bausteine bergen Sicherheitsrisiken in sich, die bisher noch nicht ausreichend untersucht sind.

Identifikation der kommunizierenden Rechner: Prinzipiell ist es bei ATM sehr viel besser möglich, kommunizierende Rechner zu identifizieren, da die Kommunikation verbindungsorientiert und nicht datagrammorientiert ist wie bei IP oder Ethernet. Ein (IP-)Datagramm kann leicht mit gefälschten Adreßinformationen versehen werden und an fast jedem Punkt ins Netz eingespielt werden. Der zur Kommunikation in ATM-Netzen notwendige Verbindungsaufbau erlaubt eine bessere Identifizierung des Kommunikationspartners. Jedoch sind die Signalisierungsnachrichten des Signalisie-

rungsprotokolls, das beim Verbindungsaufbau genutzt wird, nicht authentisiert, so daß auch hier die Absender-Adresse gefälscht werden kann. Wie verläßlich die Identifikation eingesetzt werden kann, hängt von der jeweiligen Topologie des Netzes ab. In ATM-Netzen, bei denen alle ATM-Switches des LANs mit in das Sicherheitskonzept integriert werden können, ist es möglich, Fälschungen bereits am Switch, an dem die Workstation des Angreifers angeschlossen ist, zu erkennen und zu unterbinden.[3] In diesem Falle wäre die Identifikation relativ vertrauenswürdig und könnte im Firewall genutzt werden.

Risiken bei virtuellen Verbindungen: Eine Kommunikation zwischen zwei Rechnern wird in einem ATM-Netz über eine virtuelle Verbindung geführt. Im Gegensatz zu den TCP-Verbindungen bei IP-Netzen ist eine virtuelle Verbindung nicht fest an einen Dienst gekoppelt. Bei einer TCP-Verbindung an den Port 25 kann fast sicher davon ausgegangen werden, daß an dem Port ein Mail-Server installiert ist und die Verbindung somit für die Übertragung von E-Mail dient. Eine virtuelle Verbindung kann hingegen für die Übertragung von Zugriffen auf beliebige Dienste des Kommunikationspartners genutzt werden. Beim Aufbau einer virtuellen Verbindung, über die zwei Endgeräte mit CLIP kommunizieren wollen, wird zum Beispiel nur angegeben, daß CLIP verwendet werden soll. Einem Firewall ist es daher nicht möglich, beim Aufbau der virtuellen Verbindung genauer zu unterscheiden, auf welche IP-basierten Dienste zugegriffen werden soll. Erst während der auf den Verbindungsaufbau folgenden Datenübertragung zeigt sich, auf welche IP-basierten Dienste zugegriffen werden soll. Einmal aufgebaute virtuelle Verbindungen können anschließend für Zugriffe auf beliebige IP-basierte Dienste genutzt werden. Da die virtuellen Verbindungen zudem bidirektional sind, ist nicht einmal die Richtung der Zugriffe im Vorweg zu erkennen.

Reservierung von Betriebsmitteln: Ein entscheidender Vorteil von ATM-Netzen gegenüber herkömmlichen Netzen ist die weitreichende Unterstützung von Dienstgüteanforderungen ("Quality-of-Service"). ATM-Netze ermöglichen neben der Reservierung eines garantierten Datendurchsatzes auch enge Toleranzen bei der Verzögerung von Zellen während der Übertragung.

Bisher unterstützen Firewalls noch keine Dienstgüteanforderungen. Mit zunehmender Verbreitung von ATM-Netzen wird dies jedoch eine wichtige Anforderung an zukünftige ATM-Firewalls sein.

[3] Der Angriff kann erkannt werden, da an jedem Anschluß des Switches nur ein Endgerät (Workstation) oder ein weiterer Switch angeschlossen sein kann. Die Adressen aller direkt angeschlossenen Endgeräte können kontrolliert werden. Kontrollieren auch alle anderen Switches die Adressen ihrer angeschlossenen Endgeräte, kann auch (transitiv) den von anderen Switches erhaltenen Adreßinformationen vertraut werden.

Verschiebung des LAN-WAN Übergangs: Klassische Netzwerktopologien setzen einen Router am Übergang zwischen LAN und WAN ein. Dies ist schon aufgrund des Wechsels der unterliegenden Technologie (z. B. Ethernet im LAN und X.25 im WAN) notwendig. ATM kann im LAN und im WAN gleichermaßen eingesetzt werden und macht somit keinen Wechsel der unterliegenden Technologie notwendig. Dies ermöglicht neue Topologien, bei denen die Übertragung von IP-Datagrammen ohne den Einsatz eines Routers an der LAN-WAN Schnittstelle auskommt. Ein Beispiel hierfür ist NHRP, bei dem Endgeräte direkte virtuelle Verbindungen unter Umgehung des Routers aufbauen. Bei solchen Topologien steht der Router nicht mehr für den Einsatz von Packet Screens an der LAN-WAN Schnittstelle zur Verfügung.

Neue Anwendungen: Durch ATM werden viele neue Anwendungen erst möglich. Dies hat zur Folge, daß Firewalls in Zukunft mehr anwendungsspezifische Protokolle sicher unterstützen müssen. Neue Anwendungen werden durch ATM-Netze auf zwei verschiedenen Wegen ermöglicht. Durch die höheren zur Verfügung stehenden Datendurchsätze können in ATM-Netzen neue IP-basierte Anwendungen eingesetzt werden. Proxies zur Absicherung dieser Anwendungen unterscheiden sich von herkömmlichen Proxies primär darin, daß eine höhere Leistungsfähigkeit erforderlich ist. Diese Proxies setzen jedoch weiterhin auf IP auf, so daß im wesentlichen die herkömmlichen Konzepte erhalten bleiben. Die ATM-spezifischen Vorzüge, insbesondere die Garantie einer festgelegten Dienstgüte, können erst mit "Native"-ATM-Anwendungen genutzt werden. Mit "Native"-ATM werden Anwendungen bezeichnet, die in einem ATM-Netz direkt genutzt werden können, ohne einen zusätzlichen Einsatz von IP, Ethernet oder anderen herkömmlichen Protokollen zu erfordern. Bei diesen "Native"-ATM-Anwendungen greifen die bekannte IP-basierten Konzepte nicht mehr. Daher muß untersucht werden, auf welchen Wegen eine Absicherung von "Native"-ATM-Anwendungen erreicht werden kann.

Zell-Konzept: Die Datenübermittlung findet bei ATM in Form von 53 Byte langen Zellen statt. Für Firewalls auf Anwendungsebene hat die Art der Datenübermittlung nur geringe Relevanz, da der auf Anwendungsebene ankommende Datenstrom weitgehend unabhängig von den darunterliegenden Schichten ist. Das Zell-Konzept von ATM hat jedoch Auswirkungen auf Packet Screens, da diese in tieferen Schichten eingesetzt werden. In Ethernet-basierten Netzen überprüft eine Packet Screen einen Ethernet-Rahmen, indem die im Datenbereich des Rahmens enthaltenen IP- und TCP- bzw. UDP-Header ausgewertet werden. Diese Adreßinformationen sind bei ATM in der Regel nur im Datenbereich der ersten Zelle, die zu einem IP-Datagramm gehört, enthalten. Das Einfügen von IP-Optionen oder zusätzlichen Headern kann zudem dazu führen, daß die relevanten

Informationen nicht mehr in die erste Zelle passen und sich stattdessen über mehrere Zellen erstrecken können. Neben den nur lokal bedeutsamen VPI/VCI-Angaben enthalten die ATM-Zellen selbst keine Adreßinformationen.

Virtuelle Netze: Wie bereits bei der Vorstellung von "Classical IP over ATM" (Abschnitt 2.4.2) dargestellt, können mehrere logische IP-Subnetze (LIS) auf demselben ATM-Netz aufgebaut werden.

Für die Kommunikation zwischen verschiedenen IP-Subnetzen werden, wie auch bei der Verknüpfung von nicht ATM-basierten Subnetzen, Router benötigt.

In herkömmlichen Netzen wie zum Beispiel Ethernet ist es praktisch unmöglich, verschiedene Subnetze auf derselben Infrastruktur aufzubauen und trotzdem eine strikte Trennung der Subnetze durchzusetzen. Die Zuordnung von Knoten zu Subnetzen erfolgt bei herkömmlichen Netzen in erster Linie anhand der räumlichen Nähe; typischerweise werden Subnetze etagenweise oder pro Gebäude zugeordnet. Aus Performanzgründen können etagenweise Subnetze in weitere Subnetze aufgeteilt werden. Die neue Möglichkeit von ATM zum Aufbau getrennter, logischer (IP-)Netze auf demselben ATM-Netz reduziert die äußeren Anforderungen durch bauliche Gegebenheiten und erlaubt es, Subnetze anhand von Sicherheitsanforderungen zu konfigurieren.

ATM-Switch als Kontrollkomponente: Da ATM-Netze auf Switches als aktive Komponenten basieren, ist es sinnvoll, diese mit in die Kontrolle einzubeziehen. Bisher kontrollieren Switches lediglich, ob die für die virtuelle Verbindung ausgehandelten "Quality of Service"-Parameter nicht überschritten werden (Betriebsmittel-basierte Zugriffskontrolle). Ein Switch eignet sich durch seine zentrale Position in einem ATM-Netz jedoch auch für die Durchsetzung und Überwachung von Sicherheitsrichtlinien (Sicherheitsrichtlinienbasierte Zugriffskontrolle).

In herkömmlichen Netzen war die Abwehr von "IP-Spoofing", "Denial of Service" und anderen netzbasierten Angriffen nur durch die Spezifikation von Filterregeln in Routern möglich. Durch die Definition von einfachen Filterregeln im Router wurde der erste Schritt für die Entwicklung von Firewalls gemacht. Durch ähnliche Kontrollmechanismen können ATM-Switches zukünftig auch eine starke Sicherheitsfunktion übernehmen.

Die einleitend vorgestellten Aspekte der Integration von Firewalls in ATM-Netze lassen sich in zwei Bereiche aufteilen.

- Wichtiger Aspekt beim Einsatz von Firewalls in ATM-Netzen sind die sich ergebenden Performanzprobleme. Diese Performanzprobleme werden

durch die hohen Datendurchsätze in ATM-Netzen verursacht und treten somit auch in anderen Hochgeschwindigkeitsnetzen wie Fast- und Gigabit-Ethernet auf. Die Performanz von Firewalls in Hochgeschwindigkeitsnetzen wird im Kapitel 4 untersucht

- ATM-Netze weisen Eigenschaften auf, die sich stark von herkömmlichen Netzen unterscheiden. Die Beachtung der funktionalen Unterschiede ist einerseits notwendig, um den zusätzlichen Sicherheitslücken in ATM-Netzen mit dem Firewall-Konzept geeignete Sicherheitsmechanismen entgegenzustellen. Andererseits können ATM-spezifische Eigenschaften auch ausgenutzt werden, um eine höhere Sicherheit zu erreichen, als es bei Firewalls in herkömmlichen Netzen möglich wäre. Die funktionalen Aspekte der Integration von Firewalls in ATM-Netze werden in diesem Kapitel untersucht.

Im folgenden Abschnitt werden die Schwachstellen ATM-basierter Netze aufgezeigt. Im Abschnitt 3.3 werden ATM-spezifische Firewall-Komponenten diskutiert, aus denen daran anschließend im Abschnitt 3.4 Firewall-Architekturen für ATM-Netze entwickelt werden.

3.2 Neue Schwachstellen in ATM-basierten Netzen

Die Integration eines Firewalls erfordert zunächst eine Analyse der Sicherheitsrisiken, gegen die der Firewall einen Schutz bieten soll bzw. die für den Firewall selbst eine Bedrohung darstellen. Für Firewalls in herkömmlichen Netzen sind die Sicherheitsrisiken bereits hinlänglich bekannt (s. Abschnitt 2.1). Obwohl einige Verfechter der ATM-Technologie die Auffassung vertreten, daß ATM-Netze sehr viel sicherer als IP-basierte Netze wie das Internet sind, fehlen auch in ATM-Netzen wichtige Schutzmechanismen. Um IP-Kommunikation über ein ATM-Netz zu ermöglichen, sind zudem weitere Dienste wie zum Beispiel CLIP, NHRP, LANE oder MPOA notwendig, deren Schwachstellen noch nicht ausreichend untersucht sind.[4]

3.2.1 Schwachstellen von ATM-Netzen

In diesem Abschnitt werden die Schwachstellen von ATM-Netzen aufgezeigt. Eine kurze Einführung über die Techniken von ATM-Netzen findet sich im Abschnitt

[4] Einen knappen Überblick über diesen Problembereich gibt [Vilela 98].

2.4.1 (Seite 48). Ein ATM-Netz kann selbst wiederum in mehrere Schichten aufgeteilt werden. Eine getrennte Untersuchung der Schwachstellen in der Steuerungsschicht (Signalisierung), Benutzerschicht und Managementschicht wäre zwar wünschenswert, würde an dieser Stelle aber den Rahmen sprengen. Bei dieser Untersuchung der Schwachstellen von ATM-Netzen wird das ATM-Netz stattdessen als Ganzes untersucht.

3.2.1.1 Angriffe auf die Vertraulichkeit

Ein Abhören kann bei herkömmlichen Netzen in den Endgeräten, den aktiven Komponenten, in an der Kommunikation nicht beteiligten Endgeräten und im Medium selber erfolgen. In herkömmlichen Netzen werden zum Abhören in den meisten Fällen Endgeräte benutzt, die zwar an der Kommunikation nicht direkt beteiligt sind, jedoch trotzdem den gesamten auf dem Netzstrang geführten Verkehr aufzeichnen können. Anders als in "Broadcast"-Netzen erhält ein Endgerät in einem ATM-Netz keinen Zugriff auf Dateneinheiten, die zwischen anderen Endgeräten ausgetauscht werden. Die in herkömmlichen Netzen bedeutsamen "Sniffer"-Angriffe (s. Abschnitt 2.1.1.1) stellen daher in dieser Form bei ATM-Netzen keine Bedrohung dar.

Durch den Einsatz optischer Netze wird auch das Abhören des Mediums selbst schwieriger. Mit höherem technischen Aufwand ist auch das Abhören von Glasfaserleitungen gleichwohl möglich. Zudem sind nicht alle ATM-Netze zwangsläufig auch als optische Netze realisiert. So wird in lokalen ATM-Netzen häufig eine kostengünstigere UTP-Verkabelung gewählt, die zum Beispiel auch für Ethernet Verwendung findet.

Beim Abhören der Kommunikation anderer Benutzer im selben Endgerät bestehen keine Unterschiede zwischen Endgeräten, die an ein ATM-Netz oder an ein herkömmliches Netz angeschlossen sind. In beiden Fällen ist das Abhören vergleichsweise einfach.

Die wichtigsten Komponenten eines ATM-Netzes sind die Switches. Hier fließen die Datenströme zusammen und werden an weitere Switches oder an Endgeräte weitergeleitet. Wer Kontrolle über einen Switch hat, kann die vom Switch übertragenen Daten abhören. Am einfachsten wird das Abhören eines Datenstroms durch Umkonfigurieren des Switches erreicht (s. Abb. 3.1). Der Switch kopiert dafür die auf einer virtuellen Verbindung ($VC_{client-server}$) übertragene Zellen zusätzlichen auf eine weitere virtuelle Verbindung ($VC_{sniffer}$), an dem ein Endgerät die Daten aufzeichnet. "Sniffer"-Angriffe gewinnen dadurch eine neue Qualität, da das zum Abhören verwendete Endgerät nicht direkt an den Switch angeschlossen zu sein braucht. Dadurch kann jedes beliebige, an das ATM-Netz angeschlossene Endgerät zum Aufzeichnen der Daten verwendet werden. Die

in Abbildung 3.1 eingezeichneten ATM-Teilnetze können jeweils aus mehreren ATM-Switches bestehen. Dabei wäre es auch möglich, daß die beiden virtuellen Verbindungen ($VC_{client-server}$ und $VC_{sniffer}$) über mehrere Teilnetze hinweg parallel verlaufen. Dadurch kann jeder beliebige ATM-Switch auf dem Pfad zwischen Client und Server für diesen Angriff verwendet werden. Es ist nicht notwendig, daß an dem kompromittierten Switch, wie in Abbildung 3.1 dargestellt, der Pfad zum Angreifer abzweigt.

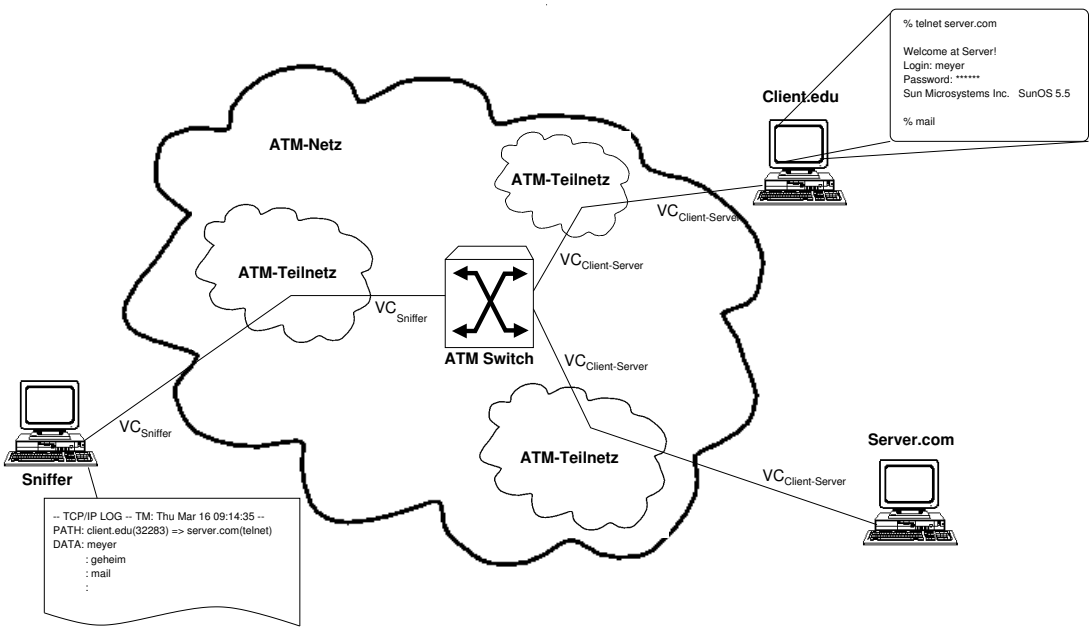


Abbildung 3.1: "Sniffer"-Angriff in einem ATM-Netz

Eine wichtige Einschränkung dieses Angriffs ist, daß er sich nur für unidirektionale Verbindungen eignet, da Datenströme von zwei Sendern (Client und Server) nicht auf dieselbe virtuelle Verbindung zum "Sniffer" kopiert werden können. Diese Einschränkung auf unidirektionale Verbindungen beseitigt jedoch nicht die grundsätzliche Gefahr, da durch eine kleine Variation dieses Angriffs noch weitergehende Angriffsmöglichkeiten erzielt werden. Wenn statt einer virtuellen Verbindung zwei virtuelle Verbindungen zum "Sniffer" hin verwendet werden, kann statt des Kopierens ein Umleiten der übertragenen Daten erfolgen. Dadurch ergibt sich für den Angreifer neben dem Abhören auch eine Möglichkeit zur Manipulation der übertragenen Daten.

Fazit: Bei einem verbindungsorientierten ATM-Netz ist das Abhören aufwendiger als bei "Broadcast"-Netzen wie z. B. Ethernet. Trotzdem stellt das Abhören auch in ATM-Netzen eine Bedrohung dar. Voraussetzung für einen Angriff ist der

physikalische Zugang zum Netz oder die Kontrolle eines ATM-Switches. Während das Abhören der auf dem Medium übertragenen Daten die Möglichkeiten der meisten Angreifer heute noch übersteigt, stellen Angriffe auf einen ATM-Switch eine neue Bedrohung dar, die bei herkömmlichen Netzen in dieser Form noch nicht ausgenutzt werden konnte. Auch wenn das Abhören in einem ATM-Netz etwas aufwendiger ist als in herkömmlichen Netzen, muß man davon ausgehen, daß mit steigender Verbreitung von ATM-Netzen auch "Sniffer"-Angriffe auftreten werden.

3.2.1.2 Angriffe auf die Authentizität und Integrität

Durch das verbindungsorientierte Konzept von ATM wird das Senden von Dateneinheiten mit gefälschten Adreßinformationen, wie es beim "IP-Spoofing" (s. Abschnitt 2.1.1.2) der Fall ist, verhindert. Die in den ATM-Zellen vorhandenen Adreßinformationen VCI und VPI (s. Abschnitt 2.4.1) sind nur von lokaler Bedeutung zwischen Endgerät und ATM-Switch und werden beim Verbindungsaufbau festgelegt. Die lokale Bedeutung ergibt sich aus der Verknüpfung des VPI/VCI-Paares mit dem Port des Switches; auf anderen Ports des Switches können dieselben VPI/VCI-Paare erneut verwendet werden. Trägt ein Angreifer ein VPI/VCI-Paar als Adreßinformationen in eine Zelle ein, das nicht zuvor bei einem Verbindungsaufbau vergeben wurde, muß die Zelle vom Switch verworfen werden, da sie keiner Verbindung zugeordnet werden kann. Das "Spoofen" von ATM-Zellen durch Endgeräte ist somit unmöglich.

"Spoofing"-Angriffe in ATM-Netzen können jedoch bereits beim Verbindungsaufbau ansetzen. Bei der Signalisierung kann neben der Zieladresse ("Called Party ID") optional auch die Absender-Adresse ("Calling Party ID") angegeben werden. Da die Signalisierungsnachrichten nicht authentisiert sind, können die Informationen vom Empfänger nicht überprüft werden [Stevenson et al. 95b]. Es wurden zwar bereits mehrere Vorschläge zur Ergänzung von Signalisierungsnachrichten um Authentisierungsinformationen gemacht [Chuang 95, Deng et al. 95, Laurent et al. 97b, Stevenson et al. 95a, Stevenson et al. 95b, Tarman 96], jedoch wird bis zur Standardisierung *einer* Lösung und zum flächendeckenden Einsatz noch längere Zeit vergehen. Die sich hieraus ergebenden Angriffsmöglichkeiten gehen über die Möglichkeiten beim "IP-Spoofing" weit hinaus [Benecke et al. 98a]. Im Internet werden die zurückgesendeten Antworten an die IP-Adresse gesendet, die beim "IP-Spoofing"-Angriff als Absender-Adresse eingetragen wurde. Dadurch erhält der Angreifer im Internet üblicherweise keine Antworten auf "gespoofte" Dateneinheiten. Das Endsystem, dessen IP-Adresse vom Angreifer verwendet wurde, reagiert zudem auf die erhaltenen Antworten und verhindert so meist das Gelingen eines "IP-Spoofing"-Angriffs. Die bei einem "Spoofing"-Angriff in einem ATM-Netz eröffneten Verbindungen sind jedoch

üblicherweise bidirektional, so daß der Empfänger seine Antworten auf derselben Verbindung wieder zurücksenden wird. Dadurch erhält der Angreifer nicht nur die Antworten vom angegriffenen Empfänger, sondern es wird auch verhindert, daß das System, dessen Adresse verwendet wurde, den Angriff bemerkt. "Spoofing"-Angriffe sind daher in ATM-Netzen als weitaus gefährlicher einzustufen als "IP-Spoofing" im herkömmlichen Internet.

Die beschriebenen "Spoofing"-Angriffe in ATM-Netzen können durch die Switches über eine Kontrolle bei der Signalisierung verhindert werden. Bei direkt angeschlossenen Endgeräten kann der Switch überprüfen, ob die bei der Signalisierung angegebene "Calling Party ID" mit der ATM-Adresse des angeschlossenen Endgerätes übereinstimmt. Die Filterung der Signalisierung durch den Switch ist vom Konzept her vergleichbar mit der Filterung von IP-Dateneinheiten, die in Routern gegen "IP-Spoofing"-Angriffe eingesetzt wird [Ferguson et al. 98]. Dort wird die Kenntnis der Netztopologie dazu eingesetzt, daß der Router eingehende Dateneinheiten nur dann weiterleitet, wenn das Netz, dessen Absender-Adresse in den Dateneinheiten verwendet wird, auch über die jeweilige Schnittstelle des Routers angeschlossen ist. Die Kontrolle durch einen ATM-Switch kann demgegenüber sehr viel genauer sein, da statt ganzer Subnetze einzelne Endgeräte kontrolliert werden. Da der Konfigurationsaufwand für entsprechende Filter in den Switches noch sehr hoch ist und zudem keine automatisierten Werkzeuge zur Unterstützung bei der Konfiguration der Filter existieren, wird bisher kaum von der Möglichkeit Gebrauch gemacht, den Switch gegen "Spoofing"-Angriffe einzusetzen.

Eine weitere Möglichkeit für "Spoofing"-Angriffe besteht über das "Integrated Local Management Interface" (ILMI). Dieses Protokoll kann von einem Endgerät genutzt werden, um nach dem "Booten" die ATM-Adresse beim Switch zu registrieren. Da auch diese Nachrichten nicht authentisiert sind, kann ein Angreifer über ILMI zusätzliche oder fremde ATM-Adressen registrieren [Ellermann et al. 98a]. Mit zusätzlichen ATM-Adressen können etwaige Filter umgangen werden, die die Signalisierung von bestimmten Adressen einschränken sollen. Fremde ATM-Adressen können nur registriert werden, wenn das jeweilige System nicht bereits registriert ist. Gelingt die Registrierung einer fremden ATM-Adresse jedoch, so können nicht nur virtuelle Verbindungen *von* dieser Adresse aus aufgebaut werden, sondern auch Verbindungen, die *zu* dieser Adresse aufgebaut werden sollen, können vom Angreifer angenommen werden. Neben der Registrierung von ATM-Adressen kann über ILMI auch der Schnittstellentyp registriert werden, was es einem Angreifer ermöglicht, sein ATM-Endgerät als ATM-Switch zu registrieren. Dadurch erhält der Angreifer eine Ausgangsbasis für nachfolgende PNNI-Angriffe.

PNNI (s. Abschnitt 2.4.1) hat zwei Aufgaben: die Signalisierung zum Aufbau von virtuellen Verbindungen und die Ermittlung von Pfaden zu den von Endgeräten

angeforderten ATM-Adressen. Die Ermittlung von Pfaden erfordert den Austausch von Erreichbarkeitsinformationen mit anderen Switches, durch die sich jeder Switch ein Bild von der Netzwerktopologie machen kann. Die ausgetauschten Topologie-Informationen können vom Empfänger bisher nicht authentisiert werden. Dies kann von einem Angreifer ausgenutzt werden, um anderen Systemen ein falsches Bild von der Netztopologie zu vermitteln [Ellermann et al. 98a]. Die Angabe besonders günstiger Pfade zu bestimmten Netzen veranlaßt andere Switches dazu, virtuelle Verbindungen immer über diese vermeintlich günstigen Pfade aufzubauen. Die über einen kompromittierten Switch führenden virtuellen Verbindungen können von einem Angreifer dann zum Beispiel leicht abgehört werden.

Fazit: Die häufig geäußerte Aussage, daß ATM wegen seines verbindungsorientierten Konzeptes weniger anfällig gegen "Spoofing"-Angriffe ist als ein IP-basiertes Netz, wurde in diesem Abschnitt widerlegt. Zwar ist es bei ATM nicht möglich, die Adreßinformationen einzelner Dateneinheiten (Zellen) vor der Übertragung zu manipulieren, jedoch können stattdessen Adreßinformationen beim Aufbau einer virtuellen Verbindung gefälscht werden. Da die gefälschte Absender-Adresse anders als bei IP-Netzen nicht für das Senden von Antworten zurück an den Angreifer benötigt wird, ist das Gefährdungspotential durch "Spoofing"-Angriffe bei ATM-Netzen sogar erheblich höher als bei IP-Netzen.

3.2.1.3 Angriffe auf die Verfügbarkeit

Die Möglichkeiten für "Denial of Service"-Angriffe auf ein ATM-Netz sind vielfältig. Ziel dieser Angriffe sind in erster Linie die ATM-Switches, da erst sie die Datenkommunikation in einem ATM-Netz ermöglichen und ein Ausfall daher die größten Folgen hat.

Exzessives Signalisieren: Während das Weiterleiten von ATM-Zellen effizient in der Hardware eines Switches implementiert ist, erfordert der Aufbau virtueller Verbindungen einen hohen Verarbeitungsaufwand im Switch. Der Durchsatz von virtuellen Verbindungen, die der Switch innerhalb eine Sekunde aufbauen kann, ist sehr begrenzt.[5] Für den normalen Betrieb stellt

[5] Messungen im Testnetz (s. Abschnitt 4.2) ergaben bei einer 155 MBit/s ATM-Schnittstelle einen Durchsatz von etwa 70 virtuellen Verbindungen, die zwischen zwei "Ultra-1" Endgeräten über einen ATM-Switch pro Sekunde auf- und wieder abgebaut werden können. Als Engpaß erweist sich hier der Signalisierungskanal, für den lediglich ein Durchsatz von 64 KBit/s zur Verfügung steht. Weitere Experimente, bei denen ein höherer Durchsatz für den Signalisierungskanal reserviert wurde und der Signalisierungskanal daher nicht mehr der Engpaß sein konnte, zeigen einen nur geringfügig höheren Durchsatz von etwa 100 Verbindungen pro Sekunde.

dies keine Einschränkung dar, weil virtuelle Verbindungen nicht so häufig aufgebaut werden müssen. Ein Angreifer kann jedoch durch exzessives Aufbauen von Verbindungen die Ressourcen eines ATM-Switches verbrauchen. Neben dem durch die CPU-Leistung des Switches beschränkten Durchsatz beim Verbindungsaufbau wird auch der Speicher des Switches aufgebraucht.

Aufbrauchen der maximalen Anzahl virtueller Verbindungen: Bei UNI [ATM Forum 94] stehen einem Endgerät maximal 8 Bit für VPI und 16 Bit für VCI zur Verfügung; es können also insgesamt 2^{24} virtuelle Verbindungen eröffnet werden. Ein Angriff durch Aufbrauchen dieser fast 17 Millionen virtuellen Verbindungen erscheint aussichtslos, jedoch wird die maximale Anzahl virtueller Verbindungen im "Device"-Treiber meist auf einen relativ geringen Wert (z.B. 1024 oder 2048) beschränkt [Benecke et al. 98a], der von einem Angreifer durchaus erreicht werden kann. Nachdem der Angreifer die entsprechende Anzahl virtueller Verbindungen zu diesem Endgerät geöffnet hat, kann dieses selbst keine weiteren Verbindungen mehr öffnen oder annehmen.

Freigeben der von Dritten reservierten Betriebsmittel: Auch bei "Denial of Service"-Angriffen kann ein Angreifer ausnutzen, daß die bei der Signalisierung verwendeten Nachrichten nicht authentisiert werden. Durch eine RESTART-Nachricht werden alle durch einen Signalisierungs-VC (<VPI=0, VCI=5> bei UNI und <VPI=0, VCI=18> bei PNNI) reservierten Betriebsmittel freigegeben [ATM Forum 94, S. 177]. Auch hierfür ist keine Authentisierung notwendig, wodurch sogar das Freigeben von Betriebsmitteln möglich wird, die für VCs fremder Benutzer reserviert wurden [Stevenson et al. 95b].

Reservieren von Betriebsmitteln: Damit ATM-Netze eine Dienstgüte garantieren können, wird ein Verbindungsaufbau von einem ATM-Switch nur dann erlaubt, wenn die für die virtuelle Verbindung angeforderten Betriebsmittel noch zur Verfügung stehen. Ein Angreifer, dem es gelingt, alle Betriebsmittel eines Engpasses für sich zu reservieren, kann so die Kommunikation anderer Teilnehmer beeinträchtigen oder gar verhindern. Ein Angreifer kann zum Beispiel eine virtuelle Verbindung eröffnen und dabei den gesamten noch nicht reservierten Datendurchsatz[6] für diese Verbindung reservieren. Für virtuelle Verbindungen anderer Systeme können anschließend keine Betriebsmittel mehr an diesem Engpaß reserviert werden, so daß der Verbindungsaufbauwunsch abgelehnt wird. Bei diesem "Denial of Service"-Angriff ist es im Gegensatz zu "Denial of Service"-Angriffen in herkömmlichen Netzen nicht notwendig, daß der Angreifer tatsächlich auch große Mengen von

[6] Der Datendurchsatz wird durch einen Engpaß auf einem beliebigen Abschnitt des Netzes zwischen Sender und Empfänger begrenzt.

Daten sendet. Neben dem Verhindern des Aufbaus neuer Verbindungen können auch bereits bestehende virtuelle Verbindungen betroffen sein, wenn für diese Verbindungen die entsprechenden Ressourcen nicht fest reserviert wurden, wie es zum Beispiel bei ABR ("Available Bit Rate") und UBR ("Unspecified Bit Rate") der Fall ist.

Da die für das Reservieren von Betriebsmitteln verwendeten Signalisierungsnachrichten nicht authentisiert sind, kann diese Form von Angriffen nicht verhindert werden. Eine teilweise Kontrolle ist lediglich über Zugriffskontrollmechanismen möglich, die es nicht zulassen, daß durch Endsysteme oder nicht kontrollierbare Teilnetze mehr als ein festgelegter Anteil der zur Verfügung stehenden Betriebsmittel reserviert wird.

Manipulationen an Zellen: ATM-Switches können mit geringem Aufwand durch direkte Manipulationen an den übertragenen Zellen die Kommunikation in einem ATM-Netz stören. Durch das Entfernen, Modifizieren oder Hinzufügen einer einzelnen Zelle kann die Übertragung einer großen Anzahl von Zellen sinnlos werden. Diese Manipulation kann erst vom Empfänger z.B. beim Reassemblieren des AAL-5-Rahmens erkannt werden. Wenn bei einem Datentransfer über CLIP IP-Dateneinheiten mit MTU-Größe (9180 Byte Länge) übertragen werden, reicht es beispielsweise aus, etwa nach jeweils 200 Zellen eine einzige Zelle nicht zu übertragen, um jede IP-Dateneinheit zu zerstören. Da die anderen Switches diesen Angriff nicht erkennen können, müssen alle anderen Zellen trotzdem bis zum Empfänger weitergeleitet werden, der das Fehlen der Zelle durch die fehlerhafte Prüfsumme des AAL-5-Rahmens erkennt und diesen AAL-5-Rahmen schließlich vernichtet.

PNNI-Angriffe: PNNI ist ein "Routing"-Protokoll, das von dem im Internet verwendeten OSPF ("Open Shortest Path First") abgeleitet ist. Ein Angreifer kann über die Angabe falscher Topologie-Informationen mit Hilfe von PNNI die Erreichbarkeit anderer Endsysteme oder Netze beeinträchtigen. (s. Abschnitt 3.2.1.2)

3.2.1.4 Implementations- und Konfigurationsfehler

Neben den beschriebenen konzeptionellen Schwächen von ATM-Netzen können praktische Realisierungen weitere Schwachstellen aufweisen. ATM-Netze haben bis heute noch keine vergleichbare Verbreitung erreicht wie zum Beispiel Ethernet. Daher sind auch ATM-Switches bisher noch nicht vergleichbar intensiven Angriffen ausgesetzt gewesen wie klassische Netzwerkkomponenten. Zwar wurde offensichtlich bei der Entwicklung von ATM-Switches auf herkömmliche Netz-

werkkomponenten zurückgegriffen,[7] trotzdem sind für ATM-spezifische Komponenten umfangreiche Neuentwicklungen notwendig. Einen Anhaltspunkt für den Entwicklungsaufwand kann zum Beispiel der Umfang der PNNI-Spezifikation [ATM Forum 96c] mit 366 Seiten geben. Bei einer Implementierung dieser umfangreichen Spezifikation muß mit schwerwiegenden Fehlern gerechnet werden, die von Angreifern zusätzlich zu den bereits diskutierten konzeptionellen Schwächen ausgebeutet werden können.

Ein weiteres Risiko entsteht durch eine unsichere Konfiguration des ATM-Netzes. Der Aufbau eines Netzes gestaltet sich bei ATM weitaus komplexer als bei herkömmlichen Technologien. Da zudem die Kompetenz für den Aufbau und Betrieb von ATM-Netzen nicht ausreichend zur Verfügung steht, wird versucht, den Verwalter durch sogenannte "Plug-and-Play"-Konfigurationen zu unterstützen. Diese "Plug-and-Play"-Konfigurationen sind sehr offen, da das Hinzufügen und Entfernen von Endgeräten ohne aufwendige Konfiguration möglich sein soll. Dies schließt eine Zugriffskontrolle aus, so daß solche Konfigurationen leicht von Angreifern ausgenutzt werden können.

Neben der initialen Konfiguration eines ATM-Netzes ist auch die Überwachung im Betrieb sehr aufwendig, so daß auf Netzwerkmanagementprotokolle wie SNMP selbst bei Netzen mittlerer Größe nicht verzichtet werden kann. Die Funktionen von SNMP reichen vom Abfragen einfacher Statistiken bis zur Konfiguration von Netzkomponenten. Da SNMP bisher hauptsächlich eine schwache, Paßwortbasierte Authentisierung über "Community Names" [Case et al. 90] verwendet, bestehen vielfältige Möglichkeiten, einen ATM-Switch über SNMP anzugreifen. Zusätzlich zu SNMP existiert bei ATM-Switches meist ein Telnet-Zugang, über den ebenfalls eine Konfiguration des Switches möglich ist. Bei Telnet werden ebenfalls abhörbare Paßworte zur Authentisierung verwendet.

Eine schwache Zugriffskontrolle sowie Implementations- oder Konfigurationsfehler können es einem Nutzer ermöglichen, in einem Endsystem direkten Zugriff auf die Signalisierung zu erhalten. Dadurch wird zum Beispiel das Konstruieren beliebiger Signalisierungsnachrichten ermöglicht.

3.2.2 Schwachstellen von CLIP-Netzen

Die Kommunikation in CLIP-Netzen[8] hängt von dem ATMARP-Server ab, der IP-Adressen in die dazugehörigen ATM-Adressen auflösen kann. In herkömmlichen Netzen werden Anfragen nach einer physikalischen Netzwerkadresse über "Broadcasts" an alle Stationen des Subnetzes gesendet. Eine Station, die für

[7] ATM-Switches des Herstellers "Cisco" verwenden zum Beispiel das Betriebssystem IOS, das bereits seit Jahren bei Routern dieses Herstellers eingesetzt wird.

[8] Eine kurze Einführung zu CLIP-Netzen findet sich im Abschnitt 2.4.2 auf Seite 55.

die gewünschte Adresse verantwortlich ist, antwortet auf die Anfrage mit dem Tupel <physikalische Netzwerkadresse, IP-Adresse>. Da die "Broadcast"-Funktionalität in einem ATM-Netz nicht zur Verfügung steht, wurde das Auflösen von Adressen bei CLIP als Client-Server-Architektur implementiert. Statt eines "Broadcasts" werden Anfragen an den ATMARP-Server gesendet, bei dem die Tupel <ATM-Adresse, IP-Adresse> für alle Stationen des logischen IP-Subnetzes ("Logical IP Subnet", LIS) registriert sind. Als zentrale, nicht redundant ausgelegte Komponente hängt die Kommunikation in einem CLIP-Netz von der korrekten Funktion des ATMARP-Servers ab. Dies macht den ATMARP-Server zum lohnenden Ziel einer Reihe von Angriffen, die in den folgenden Unterabschnitten diskutiert werden sollen. Neben dem ATMARP-Server werden weitere Schwachstellen in CLIP aufgezeigt. Die Schwachstellen in CLIP-Netzen sind in erster Linie auf die fehlende Authentisierungsmöglichkeit von ATMARP-Informationen zurückzuführen. Da CLIP-Netze auf ATM-Netzen aufsetzen, bestehen zusätzlich die bereits diskutierten Schwachstellen von ATM-Netzen. Hinweise auf die im folgenden diskutierten Schwachstellen von CLIP-Netzen finden sich in [Armitage 97, Armitage et al. 97, Benecke et al. 98a].

3.2.2.1 Angriffe auf die Vertraulichkeit

Durch den Zugriff auf den ATMARP-Server erhält ein Angreifer eine einfache Möglichkeit, ein LIS zu "scannen". Nach dem Registrieren eines <ATM-Adresse, IP-Adresse>-Tupels können beliebige "ATMARP_Requests" gesendet werden. Ist die angefragte Adresse nicht vorhanden, antwortet der ATMARP-Server mit einem expliziten "ATMARP_NAK". Es brauchen dafür keine IP-Dateneinheiten zu der jeweiligen IP-Adresse gesendet werden. Dadurch erhält ein Angreifer mit relativ geringem Aufwand eine Liste aller beim ATMARP-Server registrierten Systeme. Da diese Liste wiederum für weitere Angriffe verwendet werden kann, ist es in den meisten Fällen unerwünscht, daß diese Informationen auch von Systemen außerhalb des lokalen Netzes bzw. des LIS abgefragt werden können. Dieses "Scannen" eines LIS ist für jeden Angreifer möglich, der eine virtuelle Verbindung zum ATMARP-Server aufbauen kann.

3.2.2.2 Angriffe auf die Authentizität und Integrität

"IP-Spoofing"-Angriffe sind in CLIP-Netzen einfacher durchzuführen als in herkömmlichen Netzen.

3.2.2.2.1 "IP-Spoofing"

Im einfachsten Fall öffnet der Angreifer eine virtuelle Verbindung zum Opfer und schickt über diese Verbindung eine konstruierte ATMARP-Antwort ("InATM-

ARP_Reply") mit der für den Angriff gewählten IP-Adresse. Abhängig von der CLIP-Implementation des Opfers wird diese Angabe direkt in den "ARP-Cache" übernommen. Beim Senden von Dateneinheiten wird der "ARP-Cache" konsultiert, was zur Folge hat, daß das Opfer anschließend IP-Dateneinheiten an den Angreifer zurückschickt. Dies hat für den Angreifer eine Reihe von Vorteilen:

- Der Angreifer erhält Antworten direkt vom Opfer und ist nicht wie bei "IP-Spoofing"-Angriffen in herkömmlichen Netzen gezwungen, den Inhalt der Antworten (z. B. TCP-Sequenznummern) zu erraten.
- Ein weiterer Vorteil für den Angreifer ist, daß der rechtmäßige Inhaber der für den Angriff verwendeten IP-Adresse keine Antwort-Dateneinheiten vom Opfer erhält und daher nicht durch Reaktionen auf empfangenen Dateneinheiten das Gelingen des Angriffs verhindern kann. In herkömmlichen Netzen muß der Angreifer, um dies auszuschließen, zuerst den rechtmäßigen Inhaber der Adresse blockieren, damit von dort keine Dateneinheiten den Erfolg des Angriffs behindern. Da bei einem "IP-Spoofing"-Angriff in einem CLIP-Netz die Dateneinheiten direkt an den Angreifer zurückgehen, braucht der rechtmäßige Inhaber der Adresse nicht blockiert zu werden.
- Nicht nur zum Versenden der Antworten auf gefälschte Dateneinheiten wird die im "ARP-Cache" eingetragene ATM-Adresse verwendet. Auch beim Aufbau neuer TCP-Verbindungen oder beim Versenden von UDP-Datagrammen werden alle IP-Dateneinheiten zu der beim Angriff verwendeten IP-Adresse über die virtuelle Verbindung zum Angreifer gesendet. Dadurch kann der Angreifer zum Beispiel auch auf "Ident"-Anfragen [St. Johns 93] reagieren, die in herkömmlichen Netzen einen "IP-Spoofing"-Angriff stark erschweren können.
- Für den Angreifer ergeben sich durch einen "IP-Spoofing"-Angriff in einem CLIP-Netz umfangreiche Möglichkeiten für "Man in the Middle"-Angriffe, da die IP-Dateneinheiten der angegriffenen Systeme direkt zum Angreifer zurückkommen und das System, dessen Adresse verwendet wurde, nicht blockiert werden muß und weiterhin normal kommunizieren kann. Dadurch können "IP-Spoofing"-Angriffe gegen zwei Systeme ausgeführt werden, wobei der Angreifer beiden Systemen gegenüber vorspiegelt, der jeweilige Kommunikationspartner zu sein. Dies eröffnet sehr weitgehende Angriffsmöglichkeiten zum Beispiel gegen "Challenge-Response"-Authentisierungsverfahren.

Dieser Angriff kann von jedem System aus durchgeführt werden, das an das ATM-Netz angeschlossen ist und dadurch eine direkte virtuelle Verbindung zum angegriffenen System eröffnen kann. Der Angreifer braucht dafür selbst nicht

Teilnehmer desselben LIS zu sein; es reicht aus, beim "Spoofing" eine IP-Adresse zu verwenden, die zum LIS gehört.

Wichtig bei dieser Form des "IP-Spoofings" in CLIP-Netzen ist, daß die Daten im ATMARP-Server nicht kompromittiert werden. Ein System in einem CLIP-Netz hat prinzipiell zwei unterschiedliche Wege, über ATMARP die Abbildung zwischen ATM- und IP-Adressen herzustellen: der hier beschriebene Austausch von ARP-Informationen mit dem Kommunikationspartner oder ATMARP-Anfragen an den ATMARP-Server. Daher könnte ein CLIP-Client durch eine Kontrollanfrage an den ATMARP-Server die konstruierte ATMARP-Antwort ("InATMARP_Reply") erkennen und die Verbindung des Angreifers ablehnen.

3.2.2.2.2 "IP-Spoofing" durch Manipulation des ATMARP-Servers

Die Registrierung eines <IP-Adresse, ATM-Adresse>-Tupels beim ATMARP-Server erfordert keine Authentisierung. Dadurch ist es für einen Angreifer sehr einfach, hier manipulierte Informationen abzulegen. Hierzu muß der Angreifer lediglich eine virtuelle Verbindung zum ATMARP-Server eröffnen und einen "ATMARP_Request" an den ATMARP-Server senden, bei der die gewünschten IP-Adresse als Absender- und Empfänger-Adresse eingetragen ist. Der ATMARP-Server ergänzt das <IP-Adresse, ATM-Adresse>-Tupel in seinen Tabellen, falls dort die IP-Adresse nicht bereits mit einer anderen ATM-Adresse registriert ist. Eine doppelte Registrierung ist nicht möglich.

Für einen Angreifer gibt es nur zwei Schwierigkeiten:

- Um eine virtuelle Verbindung zum ATMARP-Server aufbauen zu können, muß dem Angreifer die ATM-Adresse des ATMARP-Servers bekannt sein. Ein Angreifer, der bereits (unprivilegierten) Zugriff auf andere Systeme des LIS hat, kann die Adresse aus der Konfiguration eines CLIP-Endsystems entnehmen.[9] Ein Angreifer ohne Zugriff auf Endsysteme im LIS muß versuchen, die ATM-Adresse des ATMARP-Servers zu erraten. Prinzipiell können Endsysteme und Switches als ATMARP-Server eingesetzt werden. Häufig werden jedoch ATM-Switches als ATMARP-Server eingesetzt,[10] wodurch der ATMARP-Server oft mit wenigen Versuchen gefunden werden kann.
- Da die doppelte Registrierung einer IP-Adresse mit unterschiedlichen ATM-Adressen vom ATMARP-Server verweigert wird, können keine IP-Adressen

[9] Die ATM-Adresse des ATMARP-Servers muß in jedem System des LIS konfiguriert sein.

[10] Die Wahl, einen ATM-Switch als ATMARP-Server einzusetzen macht Sinn, da die Funktionsfähigkeit eines CLIP-Netzes von dem ATMARP-Server *und* den ATM-Switches abhängt, während Endsysteme beliebig ein- und ausgeschaltet werden können, ohne die Funktionsfähigkeit des CLIP-Netzes zu beeinträchtigen.

von bereits registrierten Systemen verwendet werden. Weil die Registrierung beim ATMARP-Server eine Voraussetzung für die Kommunikation im LIS ist, registrieren sich CLIP-Endsysteme typischerweise sofort nach dem Einschalten. In einer Umgebung, wo Systeme nicht heruntergefahren werden, wenn kein Benutzer an der Konsole arbeitet, wird es für einen Angreifer somit schwer eine IP-Adresse zu finden, die für einen Angriff genutzt werden kann.[11] Durch zusätzliche Angriffe auf das Endsystem muß daher erreicht werden, daß das Endsystem seine Registrierung nicht mehr aufrechterhalten kann (z. B. durch "Reboot" oder durch den oben beschriebenen RESTART-Angriff). Nachdem dann die Registrierung durch den Angreifer erfolgt ist, wird jetzt die erneute Registrierung durch das Endsystem vom ATMARP-Server verweigert. Das Endsystem, dessen Adresse übernommen wurde, kann daraufhin ohne Registrierung beim ATMARP-Server nicht mehr regulär mit anderen Systemen des LIS kommunizieren. Je nach Implementation des Endsystems werden eventuell trotzdem ankommende Verbindungen angenommen, so daß der Angreifer, der die ATM-Adresse des Endsystems kennt, trotzdem Zugriffe auf das nicht registrierte Endsystem durchführen kann. Dies ermöglicht weitgehende "Man in the Middle"-Angriffe.

Nachdem die Registrierung des <IP-Adresse, ATM-Adresse>-Tupels beim ATMARP-Server erfolgt ist, kann der Angreifer die volle Funktion des Systems übernehmen, dessen IP-Adresse der Angreifer registriert hat. Alle Systeme des LIS erhalten auf ihre "ATMARP_Requests" an den ATMARP-Server die manipulierten Informationen und haben keine Möglichkeit, diese Manipulation zu erkennen. Für Dateneinheiten, die an die entsprechende IP-Adresse gehen sollen, verwenden sie daher die ATM-Adresse des Angreifers.

Dieser "IP-Spoofing"-Angriff kann von jedem System aus durchgeführt werden, das direkte virtuelle Verbindungen zum ATMARP-Server aufbauen kann sowie virtuelle Verbindungen von anderen Systemen im LIS annehmen oder dorthin aufbauen kann. Die ATM-Adressen anderer Systeme des LIS können hierfür beim ATMARP-Server erfragt werden.

Die beim "Spoofing" verwendete IP-Adresse muß im Adreßbereich des angegriffenen LIS liegen, da CLIP-Systeme keine ATMARP-Anfragen nach Systemen anderer Subnetze stellen.[12]

Eine Absicherung gegen Manipulationen der Tabellen des ATMARP-Servers ist bei dem in RFC-2225 [Laubach et al. 98] standardisierten Vorgehen praktisch

[11] "IP-Spoofing" ist nur dann sinnvoll, wenn durch das Verwendern dieser IP-Adresse zusätzliche Privilegien erworben werden können. Mit IP-Adressen, die mit keinem System verbunden sind, können in der Regel keine Privilegien erreicht werden.

[12] Systeme anderer Subnetze werden bei CLIP über Router angesprochen, auch wenn das ATM-Netz den Aufbau einer direkten virtuellen Verbindung erlauben würde.

nicht möglich. Durch eine Zugriffskontrolle beim ATMARP-Server oder über Signalisierungsfilter (s. Abschnitt 3.3.1) wäre es mit je nach Netztopologie zum Teil hohem Konfigurationsaufwand möglich die Zahl der Systeme zu reduzieren, von denen aus Angriffe durchgeführt werden können. Für eine Authentisierung der Informationen des ATMARP-Servers wären umfangreiche Ergänzungen beim RFC-2225 [Laubach et al. 98] notwendig.

3.2.2.3 Angriffe auf die Verfügbarkeit

Durch Manipulation der Informationen des ATMARP-Servers, wie bei dem im Abschnitt 3.2.2.2.2 diskutierte Angriff, kann auch die Verfügbarkeit gestört werden. Da die Registrierung beim ATMARP-Server eine Voraussetzung für die Kommunikation mit anderen Systemen eines LIS ist, sind durch Angriffe auf den ATMARP-Server auch Systeme des LIS betroffen, ohne daß diese direkt angegriffen werden müssen. Jede IP-Adresse kann beim ATMARP-Server nur einmal registriert werden. Nachfolgende Versuche, dieselbe IP-Adresse mit einer anderen ATM-Adresse zu registrieren, werden durch den ATMARP-Server abgelehnt. Durch die Registrierung von IP-Adressen anderer Systeme eines LIS kann ein Angreifer verhindern, daß sich diese Systeme beim ATMARP-Server registrieren können.

Auch durch die Registrierung beliebiger IP-Adressen kann die Verfügbarkeit des ATMARP-Servers reduziert werden. Typische Implementationen haben keine Beschränkungen der IP-Adressen, die vom ATMARP-Server registriert werden. Dadurch ist es möglich, auch beliebige Adressen außerhalb des LIS zu verwenden.[13] Der ATMARP-Server wird dadurch mit sinnlosen Informationen gefüllt, was die Funktionsfähigkeit des ATMARP-Servers reduzieren kann.

ATMARP-Server sind nicht redundant ausgelegt; für jedes LIS gibt es genau einen ATMARP-Server. Fällt dieser ATMARP-Server aus, sind alle Systeme des LIS hiervon betroffen. Nach Ausfall werden keine "ATMARP_Requests" mehr vom ATMARP-Server beantwortet, so daß Systeme im LIS keine neuen virtuellen Verbindungen zu anderen Systemen im LIS mehr aufbauen können, sofern die Informationen nicht im lokalen "Cache" des Clients vorhanden sind. Die noch im "Cache" vorhandenen Informationen können weiterhin verwendet werden. Nach 15 Minuten müssen die Informationen im lokalen "Cache" von Endsystemen erneuert werden. Spätestens zu diesem Zeitpunkt ist keine Kommunikation innerhalb des LIS mehr möglich.

[13] Ist ein ATMARP-Server für ein LIS mit IP-Adressen im Bereich 192.168.14.* zuständig, so könnte die Registrierung von IP-Adressen eines anderen Subnetzes prinzipiell abgelehnt werden. Diese Ablehnung ist meist nicht möglich, weil zur Reduzierung des Verwaltungsaufwands der IP-Adreßbereich nicht konfiguriert wird.

Von der IETF wurde mit dem "Server Cache Synchronization Protocol" (SCSP) [Luciani et al. 98b] eine Möglichkeit geschaffen, mehrere ATMARP-Server in einem LIS zu betreiben. Durch die Redundanz wird zwar eine höhere Robustheit gegen Ausfälle von ATMARP-Servern erreicht, gegen gezielte Attacken durch das Reservieren von fremden Adressen bietet diese Lösung jedoch keinen Schutz, da manipulierte Informationen über SCSP auch zu den anderen ATMARP-Servern weitergeleitet werden. Die zusätzlichen Schwachstellen, die durch den Einsatz von SCSP entstehen, sollen an dieser Stelle nicht weiter diskutiert werden.

3.2.2.4 Implementations- und Konfigurationsfehler

Implementations- und Konfigurationsfehler haben bei ATMARP-Servern besonders schwerwiegende Folgen, da die Funktionsfähigkeit eines CLIP-Netzes von der korrekten Funktion des ATMARP-Servers abhängt. Dieses Problem verschärft sich weiter, wenn derselbe ATMARP-Server für verschiedene LIS eingesetzt wird. Im normalen Betrieb stellt die Nutzung desselben ATMARP-Servers für verschiedene LIS keine Schwierigkeit dar, weil es keine Überschneidungen der Adreßbereiche geben kann. Diese Konfiguration hat zudem den Vorteil, daß die Konfiguration der Endsysteme einfacher wird, da alle Systeme denselben ATMARP-Server verwenden. Bei gebräuchlichen "Plug-and-Play"-Konfigurationen werden keine Zugriffskontrolle und kein Audit durchgeführt, so daß Angriffe weder abgewehrt noch erkannt werden können.

3.2.3 Schwachstellen bei NHRP

Das "Next Hop Resolution Protocol" (NHRP) [Luciani et al. 98a] ist eine Erweiterung von CLIP-Netzen. Daher gelten prinzipiell alle bisher für CLIP-Netze und ATM-Netze genannten Schwachpunkte. Der Unterschied zu CLIP besteht darin, daß Systeme ("Next Hop Client", NHC) zusätzlich zu den ATM-Adressen anderer NHCs desselben LIS beim "Next Hop Server" (NHS) auch ATM-Adressen von NHCs anderer Netze erfragen können.[14]

Hinweise auf einige der im folgenden diskutierten Schwachstellen von NHRP-Netzen finden sich in [Armitage 97].

3.2.3.1 Angriffe auf die Vertraulichkeit

Wie auch beim ATMARP-Server kann ein externer Angreifer beim NHS vergleichsweise einfach alle registrierten Adressen erfragen ("scannen"). Gegenüber

[14] Eine kurze Einführung zu NHRP findet sich im Abschnitt 2.4.3 auf Seite 56.

dem bereits beschriebenen Angriff auf einen ATMARP-Server vereinfachen zwei Umstände diesen Angriff bei "Next Hop Servern":

- Die ATM-Adresse des NHS ist einfacher in Erfahrung zu bringen als die ATM-Adresse eines ATMARP-Servers. Da "Next Hop Server" mit Routern gekoppelt werden, erhält ein Angreifer die Adressen, an denen ein NHS erreichbar sein könnte, über "traceroute".[15]
- Für das "Scannen" eines LIS ist es nicht mehr notwendig, eine virtuelle Verbindung direkt zum für das LIS zuständigen NHS aufzubauen. Stattdessen reicht eine virtuelle Verbindung zu einem beliebigen NHS aus, der die Anfragen an den zuständigen NHS weiterleitet.

3.2.3.2 Angriffe auf die Authentizität und Integrität

Auch beim NHS ist wie bisher beim ATMARP-Server weiterhin die Registrierung mit manipulierten Informationen möglich. Dieses "Spoofing" hat jedoch noch größere Auswirkungen als bei einem ATMARP-Server, da die Informationen eines NHS nicht nur innerhalb desselben LIS verwendet werden. Auch Anfragen nach Adressen außerhalb des LIS können gefälschte Informationen liefen, wenn an dem entfernten NHS manipulierte Informationen registriert sind.

Selbst wenn die Registrierung mit gefälschten Informationen bei einem NHS zum Beispiel über eine manuelle Konfiguration der Adressen verhindert wird, besteht weiterhin die Gefahr, daß ein Angreifer gefälschte Adreßinformationen aus dem Zuständigkeitsbereich des NHS bei einem anderen (ungeschützten) NHS registriert. Trotz der korrekten Informationen im lokalem, zuständigem NHS werden dann von einem entfernten NHS manipulierte Informationen herausgegeben. Dieser Angriff kann lange unentdeckt bleiben, da lokal keine Manipulationen erfolgt sind und die gefälschten Adreßinformationen auf entfernten NHS lange unentdeckt bleiben können.

Als Lösung für die Probleme mit der fehlenden Authentizität und Integrität der übertragenen NHRP-Nachrichten wird in RFC 2332 [Luciani et al. 98a, Abschnitt 5.3.4] eine "Authentisierungsergänzung" ("NHRP Authentication Extension") definiert. Ähnlich wie beim "Authentication Header" (AH) von IPSEC [Atkinson 95b] wird den NHRP-Nachrichten eine kryptographische Prüfsumme hinzugefügt, die mit Hilfe eines vorher vereinbarten symmetrischen Schlüssels überprüft werden kann. Als Standardverfahren wird HMAC-MD5-128 [Krawczyk et al. 97] vorgeschrieben. Außer einer manuellen Konfigurati-

[15] Das Werkzeug "traceroute" liefert die IP-Adressen aller Router auf dem Pfad zu einem Endsystem.

on der Schlüssel auf allen NHS und NHCs sind keine Schlüsselverwaltungsverfahren definiert worden. Als Schlüsselverwaltungsprotokoll bietet sich jedoch ISAKMP/Oakley [Maughan et al. 98, Orman 98, Harkins et al. 98] an, das derzeit in der IPSEC-Arbeitsgruppe der IETF entwickelt wird. Die Schwachstelle der Authentisierungsergänzung für NHRP ist, daß die Absicherung nur "Hop by Hop" und nicht "Ende zu Ende" erfolgt. Die Authentisierungsinformationen werden von jedem NHS überprüft, und vor dem Weitersenden einer Nachricht werden die Authentisierungsinformationen durch neue ersetzt. Dadurch kann jeder NHS die von ihm weitergeleiteten Daten manipulieren.

Das Sicherheitsmodell, das dieser Lösung zugrunde liegt, geht davon aus, daß jeder NHS und jeder NHC auf demselben Sicherheitsniveau liegt. Insbesondere wird angenommen, daß keines der beteiligten Systeme manipulierte Informationen mutwillig oder fahrlässig mit Authentisierungsinformationen versieht. Zudem müssen alle beteiligten Systeme dieselbe Sicherheit gegenüber Angriffen externer Systeme erreichen, da ein erfolgreicher Angriff auf eines der beteiligten Systeme die gesamte erreichte Sicherheit aushöhlt. Selbst bei überschaubaren Netzen sind diese Anforderungen kaum erfüllbar. Bei Weitverkehrsnetzen wie dem Internet ist die für NHRP definierte Authentisierungsergänzung nicht praktikabel.

3.2.3.3 Angriffe auf die Verfügbarkeit

Zusätzlich zu den bereits im Abschnitt 3.2.2.3 beschriebenen "Denial of Service"-Angriffen bei CLIP bietet NHRP noch zusätzliche Angriffsmöglichkeiten auf die Verfügbarkeit.

Eine Möglichkeit für einen "Denial of Service"-Angriff ist das Löschen von Adreßinformationen über "NHRP Purge Requests". Bestehende virtuelle Verbindungen zu diesen Adressen dürfen nach einem "Purge" nicht mehr benutzt werden. Wenn weiterhin Daten an die gleich IP-Adresse gesendet werden sollen, muß erneut über NHRP die ATM-Adresse des Zielsystems erfragt und eine neue virtuelle Verbindung zum Zielsystem aufgebaut werden. Durch das Senden eines gefälschten "NHRP Purge Requests" kann somit die Kommunikation zwischen zwei Kommunikationspartnern unterbrochen werden, bis neue Adreßinformation über NHRP ausgetauscht und eine neue virtuelle Verbindung aufgebaut wird. In größeren Netzen wird dies vermutlich zu Ausfällen von mehreren Sekunden führen. Da der Aufwand für das Senden eines "NHRP Purge Requests" relativ gering ist, kann ein einzelner Angreifer in kurzer Zeit eine große Anzahl von NHCs dazu veranlassen, neue virtuelle Verbindungen aufzubauen. Die folgenden NHRP-Anfragen und die Signalisierung der NHCs, um neue virtuelle Verbindungen aufzubauen, können zu einer Überlastung des Netzes führen.

Die in [Armitage 97] als NHRP-basierter "Denial of Service"-Angriff aufgeführte Belästigung von Endsystemen durch wiederholtes Auf- und Abbauen von vir-

tuellen Verbindungen ist nur insofern NHRP-spezifisch, als die ATM-Adressen potentieller Opfer leicht über NHRP in Erfahrung gebracht werden können. Der zugrundeliegende "Denial of Service"-Angriff – das wiederholte Auf- und Abbauen virtueller Verbindungen – ist jedoch ein ATM-spezifisches Problem.

Mögliche Angriffspunkte auf die Verfügbarkeit existieren auch bei den NHRP-Fehlermeldungen ("NHRP Error Indication", s. [Luciani et al. 98a, Abschnitt 5.2.7]). Hier eignen sich insbesondere die Fehlermeldungen "Protocol Address Unreachable" und "NHRP Loop Detected", um die Erreichbarkeit bestimmter Systeme oder Netze zu beeinträchtigen. Da die bereits beschriebene NHRP-Authentisierungsergänzung nur für NHRP-Anfragen und -Antworten ("NHRP Request", "NHRP Reply") eingesetzt werden kann, gibt es derzeit keine Schutzmaßnahme gegen das Senden gefälschter Fehlermeldungen.

3.2.3.4 Implementations- und Konfigurationsfehler

NHRP ist eine neue Entwicklung und wird bisher noch nicht in nennenswertem Umfang eingesetzt. Heutige NHRP-Implementationen dienen als Experimentierplattformen und können daher noch nicht mit ausgereifter Software verglichen werden. Bereits heute zeichnet sich ab, daß die Rolle von ATMARP-Servern zukünftig durch NHS übernommen werden wird. Der NHS wird dann für das LIS eine ebenso zentrale Rolle haben wie bisher der ATMARP-Server (s. o.). Die Aufgaben des NHS sind jedoch umfangreicher, da nur im Verbund mit anderen NHS die Adressen entfernter NHCs ermittelt werden können. Fehlfunktionen aufgrund von Implementations- oder Konfigurationsfehlern können sich daher schnell auf benachbarte NHS auswirken und in einem "Domino-Effekt" den gesamten Verbund von NHS betreffen.

3.2.4 Schwachstellen bei MARS

Der zum Ermöglichen von "Multicasts" entwickelte "Multicast Address Resolution Server" (MARS) (s. Abschnitt 2.4.4) ist eine Erweiterung des ATMARP-Servers. Auch hier wird bei der Registration keine Authentisierung durchgeführt. Da jeder Teilnehmer eines "Clusters" eine Punkt-zu-Mehrpunkt-Verbindung zu allen anderen Teilnehmern unterhält, kann ein Angreifer durch ein fortwährendes Hinzufügen und Entfernen von Teilnehmern eine sehr hohe Signalisierungslast erzeugen.[16] Das Abhören von "Multicast"-Verkehr ist zudem sehr einfach, da sich ein Angreifer bei jedem beliebigen "Cluster" registrieren kann. Es existiert bisher keine Zugriffskontrolle, die die Teilnahme an einem "Cluster" einschränkt.

[16] Alle bestehenden Teilnehmer eines "Clusters" fügen jeweils über eine ADD_PARTY-Signalisierungsnachricht den neuen Teilnehmer zu ihrer Punkt-zu-Mehrpunkt-Verbindung hinzu.

3.2.5 Schwachstellen von "LAN Emulation"

"LAN Emulation" (LANE) [ATM Forum 95a, ATM Forum 97c] stellt eine weitere Möglichkeit dar, Dateneinheiten herkömmlicher Netze über ein ATM-Netz zu transportieren. Während mit CLIP und NHRP nur IP-Dateneinheiten übertragen werden können, dient LANE zur Übertragung von Rahmen, die auch andere Protokolle (z. B. IPX) enthalten können.[17]

Sicherheitsaspekte wurden bei der Entwicklung von LANE nicht berücksichtigt. In der gesamten LANE-v2-Spezifikation [ATM Forum 97c] findet sich lediglich ein Nebensatz, in dem angemerkt wird, daß ein LECS die "Calling Party"-ATM-Adresse für eine Zugriffskontrolle verwenden kann. Weitere Hinweise auf Sicherheitsmechanismen finden sich in der LANE-Spezifikation (insgesamt 153 Seiten) nicht.

Bei LANE existieren vergleichbare Sicherheitsprobleme wie bei CLIP. Aus diesem Grund soll auf die Sicherheitsprobleme bei LANE nur kurz eingegangen werden.

3.2.5.1 Angriffe auf die Authentizität und Integrität

Zunächst muß sich ein LEC beim LECS anmelden. Die dabei verwendete ATM-Adresse kann der LEC fälschen ("spoofen"), so daß die vom LECS durchgeführte Zuordnung des LEC zu einem emulierten LAN leicht von LEC manipuliert werden kann. Die folgende Registrierung von MAC-Adresse und ATM-Adresse beim LES kann ebenfalls manipuliert werden. Erschwerend kommt hinzu, daß es für einen LEC durchaus üblich ist, mehrere Adressen zu registrieren. Dies ist zum Beispiel der Fall, wenn ein "Edge-Device" (s. Abschnitt 2.4.6, S. 62) die an einem Ethernet angeschlossenen Systeme stellvertretend registriert.

3.2.5.2 Angriffe auf die Verfügbarkeit

Auf zwei Probleme bei der Gewährleistung der Verfügbarkeit eines LANE-Netzes soll hier kurz hingewiesen werden. Das eine Problem bezieht sich auf den LECS, das andere Problem auf den BUS. Der LECS ist in der Regel für mehrere emulierte LANs gleichzeitig zuständig. Fällt der LECS aus, können sich keine neuen Stationen mehr bei einem LANE anmelden. Die Folgen eines "Denial of Service"-Angriffs sind bei einem LECS somit besonders schwerwiegend.

Ein anderes Risiko besteht beim BUS. Der BUS emuliert die in ATM-Netzen nicht vorhandene "Broadcast"-Funktionalität, indem Dateneinheiten an alle Stationen

[17] Eine kurze Einführung zur "LAN Emulation" findet sich im Abschnitt 2.4.6 auf Seite 62.

des emulierten LANs weitergeleitet werden. Der BUS multipliziert dadurch die im Netz übertragene Datenmenge und ist dadurch gut für "Denial of Service"-Angriffe einsetzbar.

Bei dieser kurzen Diskussion der Sicherheitslücken der "LAN Emulation" konnte nicht auf alle Aspekte eingegangen werden. Eine Analyse der bei LANE möglichen Angriffe findet sich in [Laurent 96].

3.2.6 Schwachstellen bei MPOA

Das im Abschnitt 2.4.7 vorgestellte MPOA wird als Ergänzung zur "LAN Emulation" eingesetzt, um virtuelle Verbindungen zwischen Systemen unterschiedlicher virtueller LANs zu ermöglichen. MPOA setzt NHRP für die Ermittlung der ATM-Adressen von Systemen anderer virtueller LANs ein. Grundsätzlich bestehen somit die bei LANE und NHRP identifizierten Schwachstellen auch bei MPOA. Welche zusätzlichen Schwachstellen bei MPOA enthalten sind, müssen detaillierte Analysen noch zeigen.

3.3 ATM-spezifische Firewall-Komponenten

Wie bei der Beschreibung herkömmlicher Firewalls in Abschnitt 2.3 sollen auch bei der Diskussion von Firewalls in ATM-Netzen zuerst die Firewall-Komponenten untersucht werden, bevor aus diesen Komponenten im Abschnitt 3.4 Firewall-Architekturen für ATM-Netze entwickelt werden.

Firewall-Komponenten erreichen abhängig von den bei der Zugriffskontrolle zur Verfügung stehenden Daten einen unterschiedlichen Detaillierungsgrad (Granularität) ihrer Zugriffskontrolle. Die erforderliche Granularität der Zugriffskontrolle richtet sich nach den Anforderungen der Sicherheitsrichtlinie.

Beim ATM-Modell wird zwischen drei Schichten unterschieden. Die Möglichkeiten und Grenzen des Einsatzes von Sicherheitsmechanismen werden für jede Schicht untersucht.

Steuerungsschicht ("Control Plane"): Der Auf- und Abbau von virtuellen Verbindungen (SVCs) über Signalisierungsnachrichten wird von der Steuerungsschicht durchgeführt. Die Signalisierung zum Aufbau von SVCs ist bereits einer Zugriffskontrolle unterworfen, die den Aufbau neuer Verbindungen ablehnt, wenn die angeforderte Dienstgüte nicht zur Verfügung gestellt werden kann. Zusätzlich zu dieser Betriebsmittel-basierten Zugriffskontrolle ist bei einigen Switches bereits eine einfache Sicherheitsrichtlinien-basierte Zugriffskontrolle implementiert, die Verbindungen zwischen

bestimmten ATM-Adressen erlauben oder verbieten kann. Die Signalisierungsnachrichten beim Verbindungsaufbau enthalten zusätzlich zu den ATM-Adressen weitere Informationen, die eine sehr leistungsfähige Sicherheitsrichtlinien-basierte Zugriffskontrolle ermöglichen, die über die bisher genutzten Möglichkeiten weit hinausgeht. Die Sicherheitsrichtlinien-basierte Zugriffskontrolle auf Signalisierungsnachrichten wird im Abschnitt 3.3.1 diskutiert.

Benutzerschicht ("User Plane"): Nach einem erfolgreichen Verbindungsaufbau werden von der Benutzerschicht Daten übertragen. Zum Einsatz kommen hier Kontrollmechanismen, die auf den Benutzerdaten arbeiten. Eine Übertragung der in herkömmlichen Netzen eingesetzten Konzepte der Packet Screen und des Proxy-Servers auf die geänderten Rahmenbedingungen in ATM-Netzen wird in den Abschnitten 3.3.2 und 3.3.3 diskutiert.

Managementschicht ("Management Plane"): Die Managementschicht dient zum Austausch von Verwaltungsinformationen in einem ATM-Netz. Hierunter fällt der Austausch von Konfigurationsinformationen über ILMI, die Übertragung von "Routing"-Informationen über PNNI und die Verwaltung von Switches über SNMP. Die Kontrolle von Managementinformationen wird im Abschnitt 3.3.4 beschrieben.

3.3.1 Kontrolle der Signalisierung

Bei ATM wird zwischen dem Aufbau einer virtuellen Verbindung (Signalisierungsphase) und der darauffolgenden Datenübertragung unterschieden. Bei der Signalisierung wird festgelegt, welche Endsysteme anschließend miteinander kommunizieren. Auf der Basis der bei der Signalisierung übertragenen Informationen (z. B. ATM-Adressen von Sender und Empfänger) kann ein Signalisierungsfilter entscheiden, ob der Aufbau der virtuellen Verbindung zugelassen werden darf.

Eine Kommunikation zwischen zwei Systemen in einem ATM-Netz kann erst nach dem Aufbau einer virtuellen Verbindung erfolgen. Bei dem Verbindungsaufbau in einem ATM-Netz wird eine virtuelle Verbindung zwischen den beiden Kommunikationspartnern geschaltet, von der die nachfolgend ausgetauschten Nachrichten nicht abweichen können. Der Aufbau virtueller Verbindungen kann entweder durch manuelle Konfiguration eines PVCs erfolgen oder die Verbindungen werden erst etabliert, wenn sie benötigt werden, wie es bei SVCs der Fall ist. Bei PVCs liegt es in der Verantwortung des Verwalters, nur solche PVCs zu konfigurieren, die den bestehenden Sicherheitsrichtlinien entsprechen. Der Aufbau eines SVCs wird durch den die Verbindung initiierenden Kommunikationspartner angefordert. Durch eine Zugriffskontrolle auf den Signalisierungsnachrichten, die für

den Aufbau eines SVCs verwendet werden, kann eine unerwünschte Kommunikation verhindert werden. Diese Zugriffskontrolle auf Signalisierungsnachrichten ist Aufgabe des Signalisierungsfilters.

Bereits heute führen ATM-Komponenten beim Aufbau eines SVCs eine einfache Zugriffskontrolle durch. Diese Zugriffskontrolle hat die Aufgabe, vor einer Reservierung von Betriebsmitteln sicherzustellen, daß noch ausreichende Betriebsmittel zur Verfügung stehen. Diese Betriebsmittel-basierte Zugriffskontrolle wird bereits bei einigen ATM-Switches um eine einfache Sicherheitsrichtlinien-basierte Zugriffskontrolle ergänzt, mit der Verbindungen zu bestimmte ATM-Adressen erlaubt oder verboten werden können.

Bei der Sicherheitsrichtlinien-basierten Zugriffskontrolle kann zwischen drei Stufen unterschieden werden.

Signalisierungsfilter: Anhand von konfigurierten Sicherheitsrichtlinien trifft der Signalisierungsfilter für eine Signalisierungsnachricht eine einfache ja/nein Entscheidung ("permit/deny"). Wenn die Sicherheitsrichtlinien die Signalisierung zulassen, wird die Nachricht unverändert weitergeleitet. Andernfalls werden die Informationselemente[18] entfernt, die gegen die Sicherheitsrichtlinie verstoßen. Falls alle Informationselemente gegen die Sicherheitsrichtlinie verstoßen beziehungsweise nach dem Entfernen der gegen die Richtlinie verstoßenden Informationselemente keine vollständige Signalisierungsnachricht mehr weitergeleitet werden kann, muß die Signalisierung durch den Signalisierungsfilter abgelehnt werden, um protokollkonform zu PNNI bzw. UNI zu bleiben.

Diese Kontrolle reicht in den meisten Einsatzbereichen aus, da bei einem Verbindungsaufbau (s. Abb. 2.14, S. 50) bereits mit der ersten Signalisierungsnachricht (SETUP) ausreichende Informationen vorliegen, um die wichtigsten Zugriffskontrollentscheidungen treffen zu können.

Die Funktionalität des Signalisierungsfilters ist vergleichbar mit der einer Packet Screen, die für jedes einzelne Paket entscheiden muß, ob ein Weiterleiten nach den Sicherheitsrichtlinien zulässig ist. Verstößt ein Paket gegen die Sicherheitsrichtlinien, erfolgt in der Regel eine Ablehnung mit einer ICMP-Nachricht. Wegen der Analogie zur Packet Screen kann für einen Signalisierungsfilter auch der englische Begriff "Signalling Screen" verwendet werden. Um die Verwendung englischer Begriffe zu vermeiden, soll im folgenden der Begriff "Signalisierungsfilter" verwendet werden.[19]

---

[18] Eine Signalisierungsnachricht besteht aus einer Reihe von Informationselementen (IEs). Eine detailliertere Beschreibung erfolgt im Abschnitt 3.3.1.1.1.

[19] Bei der Packet Screen wird hingegen weiterhin der englische Begriff verwendet, da "Packet Screen" auch im deutschen Sprachraum die gebräuchliche Bezeichnung ist. Zudem führt der deutsche Begriff "Paketfilter" zu Verwechselungen zwischen den beiden englischen Begriffen "Packet Screen" und "Packet Filter", die unterschiedliche Konzepte beschreiben.

Signalisierungsproxy: Die Funktionalität eines Signalisierungsfilters kann dadurch erweitert werden, daß die Zugriffskontrolle anhand eines Kontextes von Signalisierungsnachrichten durchgeführt wird. Durch das kontextsensitive Entfernen von Informationselementen bzw. Ablehnen von Signalisierungsnachrichten wird die Abwehr komplexer Angriffe ermöglicht.

So kann es zum Beispiel bei der Filterung einer CONNECT-Nachricht sinnvoll sein, die Informationen aus der bereits vorher kontrollierten SETUP-Nachricht zu berücksichtigen. Wichtig ist die Einbeziehung des Kontextes insbesondere bei Punkt-zu-Mehrpunkt-Verbindungen, wenn entschieden werden muß, ob das Hinzufügen eines weiteren Empfängers zu einer Punkt-zu-Mehrpunkt-Verbindung zulässig ist. Einige Zugriffskontrollentscheidungen können nur im Kontext mehrerer Verbindungen getroffen werden. Dies ist zum Beispiel für das Erkennen bestimmter Angriffsmuster ("Scannen" eines Netzes) oder für das Beenden von bereits bestehenden Verbindungen nach Eintreffen einer höher priorisierten Verbindung notwendig.

Die Bezeichnung "Signalisierungsproxy" wurde in Anlehnung an die bei Firewalls verwendeten "Proxies" gewählt, die ebenfalls einen Kontext mehrerer Nachrichten (dort besteht der Kontext aus einer Folge von Datagrammen) verwenden, um eine sehr detaillierte Zugriffskontrolle durchzuführen. Zudem findet die Signalisierung auf Anwendungsebene statt, so daß auch hier eine Übereinstimmung mit Proxies besteht.

Signalisierungskontroller: Der Signalisierungskontroller führt eine erweiterte Kontrolle der Signalisierung durch, da er Informationselemente nicht nur entfernen, sondern auch hinzufügen oder verändern kann. Statt des Ablehnens von Signalisierungsnachrichten ergibt sich hieraus die Möglichkeit, Signalisierungsnachrichten gezielt zu modifizieren, damit sie anschließend den Sicherheitsrichtlinien entsprechen.

Das Hinzufügen oder Verändern von Informationselementen kann in erster Linie dafür genutzt werden, um abhängig von den Sicherheitsrichtlinien für einzelne Verbindungen bestimmte Pfade zu wählen. Anstelle der Ablehnung einer Verbindung wird es dadurch zum Beispiel möglich, auf einem alternativen Pfad Sicherheitskomponenten einzubinden, um so die geforderte Sicherheitsdienstgüte (s. Abschnitt 3.3.1.2.1) zu erreichen. Durch die Integration weiterer Sicherheitskomponenten in einen Pfad kann eine weit über die Möglichkeiten von Signalisierungsfiltern hinausgehende Zugriffskontrolle veranlaßt werden. Die Integration der Sicherheitskomponenten Packet Screen oder Proxy-Server ist für die Zugriffskontrolle auf übertragenen Nutzerdaten zweckmäßig. Die Integration von kryptographischen Komponenten kann für die Sicherstellung von Vertraulichkeit, Integrität und Authentizität übertragener Daten auf bestimmten Teilstücken einer Verbindung (VPN) genutzt werden.

Die Bezeichnung "Signalisierungskontroller" soll auch auf die zusätzliche *Kontrolle* aufmerksam machen, die durch die Modifikation von Informationselementen der Signalisierungsnachrichten ausgeübt werden kann.

Im folgenden wird zunächst die Funktionalität des Signalisierungsfilters untersucht. An einigen Stellen ist es allerdings sinnvoll bereits auf die erweiterten Funktionalitäten des Signalisierungsproxies und des Signalisierungskontrollers einzugehen. Da diese beiden Komponenten auf einem Signalisierungsfilter basieren, trifft die Diskussion des Signalisierungsfilters auch für den Signalisierungsproxy und den Signalisierungskontroller zu. Auf die Punkte, in denen Signalisierungsproxy oder Signalisierungskontroller vom Signalisierungsfilter abweichen, wird im folgenden explizit hingewiesen.

3.3.1.1 Zugriffskontrolle über einfache Signalisierungsfilter

Handelsübliche ATM-Switches enthalten bereits einfache Filterregeln, mit denen auf einzelnen Schnittstellen eingehende und herausgehende Signalisierungswünsche gefiltert werden können. Der Aufbau der Regeln lehnt sich dabei stark an die von Routern bekannten Filterlisten für IP-Verkehr an. Statt auf IP-Adressen beziehen sich hier die Regeln jedoch auf ATM-Adressen. Neben den ATM-Adressen vom Initiator[20] und vom Zielsystem, die beim Verbindungsaufbau gesendet werden, gibt es in Signalisierungsnachrichten noch weitere Informationen, die zukünftig von einer Zugriffskontrolle überprüft werden können. Hierzu sollen zunächst die verwendeten Signalisierungsnachrichten vorgestellt werden.

Aufgrund der unterschiedlichen Signalisierungsprotokolle muß bei Signalisierungsfiltern zwischen zwei verschiedenen Einsatzbereichen unterschieden werden.

- Ein einzelnes Endgerät kann an der Schnittstelle zwischen ATM-Endgerät und ATM-Netz durch den Signalisierungsfilter überwacht werden. An dieser Schnittstelle wird das Protokoll UNI[21] eingesetzt.
- Zwischen zwei ATM-Switches kann ein Signalisierungsfilter den Verbindungsaufbau zwischen beiden Netzen kontrollieren. An dieser Schnittstelle wird zur Signalisierung PNNI[22] verwendet, dessen Format dem UNI sehr

[20] Das System von dem der Aufbau einer virtuellen Verbindung ausgeht, wird als "Initiator" bezeichnet.

[21] Diese Beschreibung bezieht sich auf UNI 3.1 [ATM Forum 94]. Ergänzungen und zusätzliche Funktionalitäten von UNI 4.0 [ATM Forum 96d] sind explizit gekennzeichnet.

[22] Neben der Signalisierung für die Steuerung von Verbindungen ist das Austauschen von "Routing"-Informationen eine zweite wichtige Aufgabe von PNNI. Die Kontrolle von PNNI-Nachrichten mit "Routing"-Informationen ist ebenfalls über Signalisierungsfilter möglich, soll an dieser Stelle aber nicht untersucht werden. (Siehe Abschnitt 3.3.4)

ähnlich ist, daß aber über zusätzliche Informationselemente (z.B. DTL und "Crankback") verfügt. Informationselemente der UNI-Nachrichten eines Endgerätes werden zum größten Teil unverändert in PNNI-Nachrichten übernommen.

Da sich UNI- und PNNI-Nachrichten vom Aufbau her nicht unterscheiden und nur einige wenige IEs nicht bei beiden Protokollen enthalten sind, werden bei der folgenden Diskussion beide Protokolle gemeinsam untersucht. Auf die Fälle, in denen Unterschiede zwischen UNI und PNNI bestehen, wird explizit hingewiesen.

3.3.1.1.1 Filterung von Signalisierungsnachrichten

Allgemeiner Aufbau von Signalisierungsnachrichten: PNNI- und UNI-Signalisierungsnachrichten bestehen aus einer Reihe von Informationselementen ("Information Element", IE) in denen die für das Management einer Verbindung benötigten Informationen enthalten sind. Für jede Signalisierungsnachricht sind die Informationselemente definiert, die in der jeweiligen Nachricht enthalten sein müssen, sowie die Informationselemente, die optional enthalten sein können. Vier Informationselemente bestimmen Typ und Länge einer Signalisierungsnachricht und müssen in jeder Nachricht enthalten sein. Die vier IEs sind "Protocol Discriminator", "Call Reference", "Message Type" und "Message Length".

"Protocol Discriminator": Dieses IE identifiziert die Signalisierungsnachricht als UNI-Nachricht (bzw. Q.2931 nach der entsprechenden ITU[23] Bezeichnung) oder als PNNI-Nachricht.

"Call Reference": Dieses IE identifiziert alle zu einer Signalisierung gehörenden Nachrichten.

"Message Type": Der Nachrichtentyp bezeichnet die jeweilige Signalisierungsnachricht (SETUP, CONNECT, etc.).

"Message Length": Die Nachrichtenlänge gibt die Länge der Signalisierungsnachricht an. Dabei werden die hier genannten vier IEs nicht eingerechnet, da sie bei jeder Signalisierungsnachricht vorhanden sein müssen und eine feste Länge von 9 Byte haben.

"Protocol Discriminator"- und "Message Type"-IE müssen von einem Signalisierungsfilter untersucht werden, um den Typ der Signalisierungsnachricht zu

[23] "International Telecommunication Union" (ITU)

identifizieren. Die anderen beiden IEs können nur für Plausibilitätsüberprüfungen durch den Signalisierungsfilter genutzt werden. Möglich ist etwa die Überprüfung, ob die "Call Reference" zu einer gerade aktiven Signalisierung gehört[24] und ob die angegebene Nachrichtenlänge korrekt ist.

Im folgenden werden für jede Signalisierungsnachricht die für eine Zugriffskontrolle geeigneten IEs untersucht.

Signalisierungsnachrichten für den Verbindungsaufbau: Der Aufbau einer virtuellen Verbindung erfolgt über die Nachrichten SETUP, CONNECT, CONNECT_ACKNOWLEDGE und CALL_PROCEEDING, wie in Abbildung 2.14 (Seite 50) dargestellt.

SETUP:

Diese Nachricht wird vom Initiator ("Calling User") an das Netzwerk und vom Netzwerk weiter bis zum Empfänger ("Called User") gesendet. SETUP ist die erste Nachricht, die bei einem Verbindungsaufbau gesendet wird.[25] In dieser Signalisierungsnachricht sind die wichtigsten für eine Zugriffskontrolle verwendeten Informationen enthalten.

Sender- und Empfänger-Adressen: Die wichtigsten Informationen für eine Zugriffskontrolle sind die Adressen der Kommunikationspartner. Das "Called Party Number"-IE enthält die ATM-Adresse des Empfängers und im "Calling Party Number"-IE ist die Adresse des Initiators (Absender) enthalten. Wie bei IP-Netzen besteht die Möglichkeit des Fälschens der Absender-Adresse. Zudem ist es nicht erforderlich, bei der Signalisierung eine Absender-Adresse anzugeben.

Eine weitere Schwierigkeit ist das Fehlen einer einheitlichen Adressierung in ATM-Netzen. Verwenden die Endsysteme nicht die in öffentlichen Netzen gebräuchlichen E.164-Adressen, muß beim Verbindungsaufbau über ein öffentliches Netz die E.164-Adresse eines Knotens des Zielnetzes angegeben werden. Die eigentliche Empfänger-Adresse wird

[24] Diese Überprüfung kann nur im Kontext mehrerer Nachrichten durchgeführt werden und erfordert daher die erweiterte Funktionalität des Signalisierungsproxies.

[25] Eine Ausnahme ist der "Leaf Initiated Join" (LIJ) mit dem sich ein Endsystem über die Nachricht LEAF_SETUP_REQUEST an eine bestehende Punkt-zu-Mehrpunkt-Verbindung ankoppeln kann. Bisher ist der LIJ nur in UNI 4.0 [ATM Forum 96d] spezifiziert. Da in PNNI noch keine Unterstützung des LIJs enthalten ist [ATM Forum 96c, S. 39], beschränkt sich die Nutzung auf die UNI-Schnittstelle zwischen Endgerät und Switch. Daher wird der LIJ an dieser Stelle nicht diskutiert. Sobald eine entsprechende Unterstützung in PNNI verfügbar wird, müssen die mit dem LIJ verbundenen Signalisierungsnachrichten von einem Signalisierungsfilter kontrolliert werden, da durch einen LIJ eine Reihe zusätzlicher Sicherheitsrisiken entstehen.

dann im "Called Party Subaddress"-IE angegeben. Mit der "Calling Party Subaddress" steht auch ein vergleichbares IE für die Absender-Adresse zur Verfügung. Ein Signalisierungsfilter muß berücksichtigen, daß die Adressen auch in diesen IEs enthalten sein können.

Eine spezielle Adreßklasse sind "Anycast"-Adressen, über die ein beliebiges System aus einer Gruppe von Systemen adressiert werden kann. Sinnvoll sind solche Adressen immer dann, wenn ein bestimmter Dienst angesprochen werden soll, es aber nicht notwendig ist, daß der Dienst von einem bestimmten System erbracht wird. Dies kann zum Beispiel zur automatischen Konfiguration von Endsystemen genutzt werden. "Anycasts" werden von PNNI und UNI 4.0 unterstützt; bei UNI 3.x sind "Anycasts" noch nicht möglich. Da bei einer "Anycast"-Adresse der Empfänger noch nicht endgültig bestimmt ist, wird die Filterung durch einen Signalisierungsfilter erschwert, da ein beliebiges System der Gruppe als Empfänger in Frage kommt. Erschwerend kommt weiterhin hinzu, daß nicht unbedingt alle Systeme der Gruppe à priori bekannt sind. Andererseits haben die Systeme, die zu einer "Anycast"-Adresse gehören, oft dieselben Eigenschaften und Sicherheitsanforderungen, können daher auch vom Signalisierungsfilter gleich behandelt werden. Verbindungen von internen Systemen zu "Anycast"-Adressen außerhalb des geschützten Netzes müssen durch den Signalisierungsfilter in der Regel verhindert werden, wenn vom Empfänger zurückgesendete Daten bei dem internen System Schäden anrichten können. Das Beispiel einer automatischen Konfiguration zeigt dieses Problem deutlich.

Fazit: Durch eine Auswertung von Sender- und Empfänger-Adressen können die Zugriffe zwischen bestimmten Endsystemen eingeschränkt werden. Derartige Regeln sind bereits in verschiedenen Switches verfügbar. Allerdings sind die Absender-Adressen leicht fälschbar und sind zudem auch nicht zwingend in jeder SETUP-Nachricht enthalten.

Pfadinformationen: Der Pfad durch das Netz ist in dem "Designated Transit List"-(DTL)-IE enthalten. Das DTL-IE ist nur bei PNNI definiert. Als "Source Route" wird der Pfad durch eine "Peer Group" bereits vom ersten Switch der "Peer Group" ("Border Node") in einem DTL-IE festgelegt. "Peer Groups" sind hierarchisch angeordnet (s. Abb. 3.2), so daß für jede Hierarchieebene eine eigene DTL angegeben werden muß. Nur die DTL der "Peer Group" auf der untersten Hierarchieebene enthält die ATM-Adressen der Switches auf dem Pfad. DTLs höherer Hierarchieebenen enthalten stattdessen nur logische Bezeichner für die "Peer Groups" auf dem Pfad zum Empfänger.

Die Auswertung von DTLs durch einen Signalisierungsfilter ist ebenso wichtig wie das Filtern der Empfänger-Adressen. Anders als in

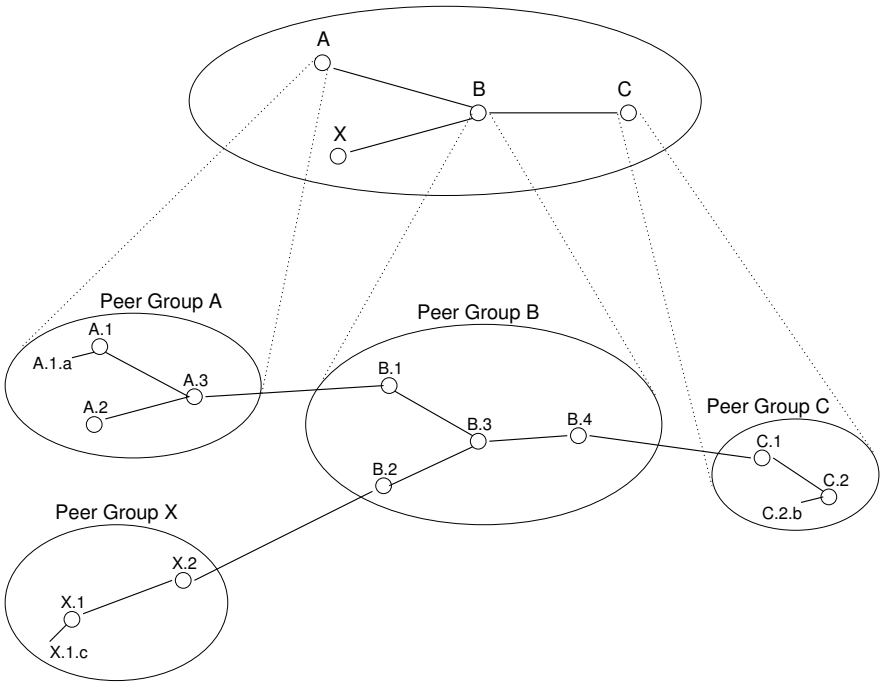


Abbildung 3.2: ATM-Netz mit zwei PNNI-Hierarchieebenen

Datagramm-orientierten Netzen, wo die Empfänger-Adresse auch den Pfad eines Datagramms bestimmt, wird der Pfad in einem ATM-Netz durch die DTL bestimmt. Die DTL kann dazu führen, daß eine Signalisierungsnachricht ein anderes als das in der Empfänger-Adresse angegebene System erreicht. Solche Angriffe müssen durch die Überprüfung der DTL durch einen Signalisierungsfilter erkannt werden, indem der Signalisierungsfilter selbst zulässige DTLs zu der angegebenen Empfänger-Adresse ermittelt und diese mit der vorliegenden DTL vergleicht.

In dem DTL-IE wird über den "Current Transit Pointer" angegeben, wie weit der Verbindungsaufbau der jeweiligen DTL bereits erfolgt ist. Über die Kontrolle der bereits bearbeiteten Einträge einer DTL können gefälschte Absender-Adressen ("ATM Address Spoofing") zumindest teilweise erkannt werden. Außerdem kann bei SETUP-Nachrichten, bei denen kein "Calling Party Number"-IE vorhanden ist, der mögliche Absender auf bestimmte Systeme eingeschränkt werden.

Die Möglichkeiten eines Signalisierungsfilters, gefälschte ATM-Adressen zu erkennen, sollen an einem kleinen Beispiel demonstriert werden. Ein Angreifer X.1.c (s. Abb. 3.2), der an den Switch X.1 in der "Peer Group" X angeschlossen ist, versucht, eine Verbindung zum System C.2.b zu eröffnen, und fälscht dabei die Absender-Adresse vom Sy-

stem A.1.a der "Peer Group" A. Die SETUP-Nachricht, die beim Signalisierungsfilter im Switch B.4 ankommt, enthält in diesem Fall zwei DTL-IEs: die DTL der lokalen Peer-Group <B.2, B.3, B.4> und die DTL auf der höheren Hierarchiestufe mit den logischen Knoten <X, B, C>. An der letzteren DTL kann der Signalisierungsfilter erkennen, daß die SETUP-Nachricht nicht aus der "Peer Group" A sondern der "Peer Group" X stammt. Selbst unter des Annahme, daß die Switches der "Peer Group" X ebenfalls kompromittiert sind und auch gefälschte DTLs mit dem Inhalt <A, B, C> generieren, kann der Signalisierungsfilter anhand der lokalen DTL <B.2, B.3, B.4> erkennen, daß der Verbindungsaufbau über den Switch B.2 erfolgt ist, was unter Kenntnis der Topologie bei einem Initiator in der "Peer Group" A nicht der Fall sein kann. Bei komplexeren Topologien kann die Erkennung gefälschter ATM-Adressen schwieriger werden bzw. nur teilweise möglich sein.

Ein weiterer Anwendungsbereich für eine Zugriffskontrolle auf den bei der Signalisierung enthaltenen Pfadinformationen ist das Ausschließen unerwünschter Pfade. Da in den DTLs bereits die Pfadinformationen bis zum Empfänger vorgegeben sind, kann der Signalisierungsfilter verhindern, daß eine virtuelle Verbindung über ein bestimmtes als unsicher bekanntes Netz hinweg aufgebaut wird.

Über das von PNNI und UNI unterstützte "Transit Network Selection" (TNS) Informationselement kann ein ATM-Endgerät die Erzeugung von DTLs beeinflussen. Ein Endgerät kann über dieses IE ein bestimmtes Netz auswählen, das im Pfad enthalten sein soll. Dadurch soll dem Endgerät die Auswahl eines bestimmten "Providers" ermöglicht werden. ATM-Switches berücksichtigen dieses Feld bei der Generierung von DTLs. Ein Signalisierungsfilter kann dieses IE filtern, um den Aufbau einer virtuellen Verbindung über ein unsicheres Netz zu verhindern.

Fazit: Durch die Auswertung der Pfadinformationen kann von einem Signalisierungsfilter der Verbindungsaufbau über unsichere Pfade unterbunden werden. Eine interessante neue Anwendung stellt die Auswertung von DTLs zur Erkennung von gefälschten ATM-Adressen dar. Beide Anwendungen ergeben eine deutliche Erweiterung der bisher verfügbaren Funktionalität von Signalisierungsfiltern. Eine noch weitreichendere Kontrolle ist über die Manipulation von DTLs durch einen Signalisierungskontroller erreichbar. Die Manipulation von DTLs zum Einbinden von Sicherheitskomponenten wird im Abschnitt 3.3.1.2 diskutiert.

Informationen zur Verarbeitung in den Endsystemen: Über das Informationselement "Broadband Low Layer Information" (BLLI) (s. Ab-

schnitt 2.4.1) wird bestimmt, welche Schicht-2- oder Schicht-3-Protokolle zwischen den Kommunikationspartnern auf der virtuellen Verbindung verwendet werden sollen. Hieran kann z. B. erkannt werden, ob die virtuelle Verbindung für den Austausch von IP-Dateneinheiten genutzt werden soll. Zu erkennen ist ferner, ob dabei ein SNAP-Header zum Einsatz kommt oder ob *"VC-basiertes Multiplexing"* (s. Abschnitt 2.4.1) eingesetzt wird. Bei der Verwendung von SNAP hat ein Signalisierungsfilter geringere Kontrollmöglichkeiten, da über den SNAP-Header innerhalb des Datenstroms — somit außerhalb der Kontrolle des Signalisierungsfilters — unterschiedliche Protokolle ausgewählt werden können.

Das Informationselement "Broadband High Layer Information" (BHLI) gibt die Entität des Empfängers auf Anwendungsebene an, für die diese Verbindung gedacht ist. Das BHLI-IE wird bei "Native"-ATM-Anwendungen zur Adressierung des Empfängers verwendet. Ein Signalisierungsfilter kann durch Kontrolle des BHLI-IEs eine ähnliche Kontrolle erreichen wie eine Packet Screen durch Filterung von TCP- und UDP-Ports.

Ein Signalisierungsfilter sollte in diesem Rahmen auch überprüfen, in wie weit zusätzliche Parameter der angeforderten Verbindung für das Endsystem geeignet sind. Die Kontrolle muß beinhalten, ob eine geeignete Adaptionsschicht mit akzeptablen maximalen SDU-Größen gewählt wurde. Diese Informationen sind im Informationselement "AAL Parameters" enthalten.

Fazit: Über das BLLI-IE kann erkannt werden, ob eine virtuelle Verbindung zum Transport von IP-Dateneinheiten dienen soll. Eine Zugriffskontrolle auf dem BHLI-IE ermöglicht bei "Native"-ATM-Anwendungen eine ähnliche Kontrolle wie eine Packet Screen mit Filterregeln auf TCP- und UDP-Ports bei IP-basierter Kommunikation.

Informationen zur angeforderten Dienstgüte: Der Anforderung von Dienstgüteparametern dienen mehrere IEs: "AAL Parameter", "ATM Traffic Descriptor", "Broadband Bearer Capability" und "QoS Parameter". Die Kontrolle des Zugriffs auf die Ressourcen des Netzes stellen eine wichtige Aufgabe eines Firewalls in einem ATM-Netz dar. Ein Signalisierungsfilter kann die Kontrolle über die Anforderung von Betriebsmitteln des Netzes übernehmen.

Weitere Informationselemente: Das Informationselement "Endpoint Reference" wird bei Punkt-zu-Mehrpunkt-Verbindungen genutzt, um für jedes Endgerät eine Identifikation zu vergeben. SETUP-Nachrichten, die ein "Endpoint Reference"-IE mit einem "Endpoint Reference Identifier Value" ungleich null (0) enthalten, ergänzen eine Punkt-zu-Mehrpunkt-Verbindung um ein weiteres Endgerät. Für den Signalisierungs-

filter kann es wichtig sein, daß es bereits weitere Endgeräte an anderen Ästen dieser Punkt-zu-Mehrpunkt-Verbindung gibt. Punkt-zu-Mehrpunkt-Verbindungen sind unidirektional, deshalb muß der Signalisierungsfilter bei der Entscheidung nur berücksichtigen, daß es bereits andere *Empfänger* auf dieser Punkt-zu-Mehrpunkt-Verbindung gibt.

Über das "Generic Identifier Transport"-IE kann ein Endgerät bei UNI 4.0 [ATM Forum 96d, Abschnitt 2.1.1] zusätzliche Informationen für den Empfänger mit in einer Signalisierungsnachricht übertragen. Die Bedeutung der in diesem IE enthaltenen Daten wird von UNI 4.0 nicht spezifiziert. Denkbar wäre es zum Beispiel, hiermit kryptographisches Schlüsselmaterial auszutauschen. Von Angreifern könnte dieses IE jedoch auch zum verdeckten Senden von Informationen ausgenutzt werden. Wenn die Bedeutung dieser Informationen durch den Signalisierungsfilter nicht erkannt werden kann, ist es empfehlenswert, bei einem Signalisierungsfilter dieses IE nicht zuzulassen.

CONNECT:

Diese Nachricht wird vom Empfänger ("Called User") als Antwort auf ein SETUP an das Netzwerk und vom Netzwerk weiter an den Initiator ("Calling User") zur Bestätigung des Verbindungsaufbaus gesendet. Die Werte, die in den IEs einer CONNECT-Nachricht gesendet werden dürfen, sind bereits durch die in der vorangegangenen SETUP-Nachricht enthaltenen Daten beschränkt. Falls beim SETUP z. B. mehrere "Broadband Low Layer Information"-IEs (s. o.) angegeben wurden, kann beim CONNECT eines dieser BLLI-IEs ausgewählt werden. Auch bei den "AAL Parametern" kann der Empfänger über die CONNECT-Nachricht andere Parameter (z. B. eine kleinere SDU-Größe) an den Initiator zurücksenden. Auch das Aushandeln von Dienstgüteparametern ist mit PNNI möglich; an der Schnittstelle zwischen ATM-Endgerät und ATM-Netz ist das Aushandeln aber erst mit UNI 4.0 möglich. Grundsätzlich gilt, daß alle IEs, die vom Signalisierungsfilter bereits zur Kontrolle der SETUP-Nachricht genutzt wurden, bei einer CONNECT-Nachricht erneut überprüft werden müssen.

Die Möglichkeit über das "Generic Identifier Transport"-IE verdeckt Informationen zu senden, besteht auch bei der CONNECT-Nachricht. (vgl. SETUP)

CONNECT_ACKNOWLEDGE:

Diese Nachricht ist nur für UNI definiert. Nachdem der Benutzer ein CONNECT gesendet hat, bestätigt das Netz mit dem CONNECT_ACKNOWLEDGE den Aufbau der Verbindung. Diese Nachricht hat nur eine lokale Bedeutung zwischen Endgerät und ATM-Switch.

Für die Kontrolle dieser Nachricht durch einen Signalisierungsfilter konnte keine sinnvolle Anwendung gefunden werden.

CALL_PROCEEDING:

Diese Nachricht kann als Antwort auf ein SETUP gesendet werden, um den Empfang zu bestätigen. Diese Nachricht hat nur lokale Bedeutung. Dies bedeutet, daß sie nur zwischen benachbarten ATM-Komponenten ausgetauscht wird.

Für die Kontrolle dieser Nachricht durch einen Signalisierungsfilter konnte keine sinnvolle Anwendung gefunden werden.

Signalisierungsnachrichten für den Verbindungsabbau: Für den Abbau virtueller Verbindungen werden die Nachrichten RELEASE und RELEASE_COMPLETE verwendet (s. Abbildung 2.14, Seite 50).

RELEASE und RELEASE_COMPLETE:

Mit dieser Nachricht teilt ein ATM-Knoten den Verbindungsabbau mit. Die "Call Reference" bleibt jedoch noch erhalten bis ein RELEASE_COMPLETE eintrifft. RELEASE und RELEASE_COMPLETE werden bis zum Endgerät durchgereicht.

Eine Zugriffskontrolle des Signalisierungsfilters auf RELEASE-Nachrichten hat nur einen beschränkten Nutzen, da die Signalisierungsnachrichten nicht authentisiert werden können. Die Auswertung von RELEASE- und RELEASE_COMPLETE-Nachrichten durch einen Signalisierungs*proxy* kann jedoch dazu dienen, den Zustand von Verbindungen zu verfolgen. Die Verfolgung des Zustands von Verbindungen wird etwa benötigt, um Betriebsmittelanforderungen zu überwachen oder um typische Angriffsmuster (z. B. das "Scannen" eines Netzes) zu erkennen.

Bei PNNI kann bei einem RELEASE und einem RELEASE_COMPLETE jeweils ein "Crankback"-IE enthalten sein, mit dem mitgeteilt wird, an welcher Stelle ein Verbindungsaufbau fehlgeschlagen ist. Der Switch, der die DTL generiert hat, kann dann einen alternativen Pfad suchen. Der Signalisierungsfilter kann aus diesem IE Informationen über den Zustand des Netzes gewinnen. Eine Zugriffskontrolle auf diesem IE erscheint nur dann sinnvoll, wenn durch den Signalisierungsfilter die Struktur des Netzes verdeckt werden soll und daher die in dem IE enthaltenen ATM-Adressen nicht an die Öffentlichkeit dringen dürfen.

Die Möglichkeit über das "Generic Identifier Transport"-IE verdeckt Informationen zu senden, besteht auch bei der RELEASE-Nachricht. (vgl. SETUP)

Steuerung von Punkt-zu-Mehrpunkt-Verbindungen: Zur Steuerung von Punkt-zu-Mehrpunkt-Verbindungen werden ADD_PARTY, ADD_PARTY_AC-

KNOWLEDGE, ADD_PARTY_REJECT, DROP_PARTY und DROP_PARTY_ACKNOWLEDGE Signalisierungsnachrichten verwendet.

ADD_PARTY:

Punkt-zu-Mehrpunkt-Verbindungen gehen immer von einem "Wurzel"-System ("Root") aus, das für den Verbindungsaufbau und das Hinzufügen von weiteren Teilnehmern ("Leaves") verantwortlich ist.[26]

Mit der ADD_PARTY-Nachricht werden weitere Teilnehmer zu einer bestehenden Punkt-zu-Mehrpunkt-Verbindung hinzugefügt. ADD_PARTY-Nachrichten werden nur für bestehende Verbindungen genutzt. Zum Aufbau einer neuen Punkt-zu-Mehrpunkt-Verbindung und für die neu hinzukommenden Teilstücke der Punkt-zu-Mehrpunkt-Verbindung werden SETUP-Nachrichten verwendet.

Die ADD_PARTY-Nachricht enthält im wesentlichen dieselben Informationselemente wie eine SETUP-Nachricht (s. o.). Dadurch bestehen auch dieselben Filtermöglichkeiten auf Sender- und Empfänger-Adressen, auf Pfadinformationen (DTLs) und auf den Informationen zur Verarbeitung in den Endsystemen (BLLI, BHLI, AAL-Parameter). Die für die Verarbeitung in den Endsystemen verwendeten Informationen müssen dabei den in der ersten SETUP-Nachricht gesendeten Informationen entsprechen. Die Übereinstimmung sollte durch einen Signalisierungsproxy überwacht werden.

Zusätzlich sind Filterregeln sinnvoll, bei denen das Hinzufügen in Abhängigkeit von den bereits bestehenden Teilnehmern erlaubt wird. Ein Signalisierungskontroller[27] könnte eventuell auch bestehende Teilnehmer von einer Punkt-zu-Mehrpunkt-Verbindung ausschließen (DROP_PARTY, s. u.), sobald bestimmte neue Teilnehmer hinzugefügt werden.

ADD_PARTY_ACKNOWLEDGE:

Die ADD_PARTY_ACKNOWLEDGE-Nachricht wird als Bestätigung einer erfolgreichen ADD_PARTY-Nachricht an das Wurzel-System zurückgesendet. Die erlaubten IEs und die Filtermöglichkeiten für einen Signalisierungsfilter entsprechen bei PNNI und UNI 4.0 weitgehend denen einer CONNECT-Nachricht. Erforderlich ("mandatory") ist jedoch nur das "Endpoint Reference"-IE. Bei UNI 3.x wird bei dieser Signalisierungsnachricht außer dem "Endpoint Reference"-IE kein weiteres IE unterstützt.

---

[26] In UNI 4.0 [ATM Forum 96d] ist zusätzlich auch ein "Leaf Initiated Join" (LIJ) vorgesehen, durch den sich Teilnehmer auch ohne Beteiligung des "Wurzel"-Systems bei einer Punkt-zu-Mehrpunkt-Verbindung anmelden können. Für den LIJ gibt es jedoch in PNNI [ATM Forum 96c] noch keine Unterstützung. Auf eine Untersuchung wird daher an dieser Stelle verzichtet.

[27] Da zusätzliche Signalisierungsnachrichten generiert werden müssen, wird für diese Aufgabe ein Signalisierungskontroller benötigt. Ein Signalisierungsproxy reicht für diese Aufgabe nicht aus.

ADD_PARTY_REJECT:
Diese Nachricht wird an das Wurzel-System zurückgesendet, wenn das entsprechende ADD_PARTY nicht erfolgreich war. Diese Nachricht entspricht der RELEASE_COMPLETE-Nachricht (s. o.). An der PNNI-Schnittstelle kann auch die ADD_PARTY_REJECT-Nachricht ein "Crankback"-IE enthalten, um die Stelle zu identifizieren, an der der Verbindungsaufbau gescheitert ist.

DROP_PARTY und DROP_PARTY_ACKNOWLEDGE:
Die DROP_PARTY-Nachricht dient zum Entfernen eines Teilnehmers einer Punkt-zu-Mehrpunkt-Verbindung. Zur Bestätigung des Entfernens wird das DROP_PARTY_ACKNOWLEDGE verwendet. Die beiden IEs entsprechen RELEASE- und RELEASE_COMPLETE-Nachrichten.

Die Auswertung der Nachrichten ADD_PARTY_ACKNOWLEDGE, ADD_PARTY_REJECT, DROP_PARTY und DROP_PARTY_ACKNOWLEDGE ist für einen Signalisierungsproxy notwendig, wenn der Zustand der Punkt-zu-Mehrpunkt-Verbindung überwacht werden soll. Dies ist zum Beispiel erforderlich, falls Zugriffskontrollentscheidungen anhand der Liste der momentanen Teilnehmer gefällt werden sollen.

Weitere Nachrichten sind RESTART, RESTART_ACKNOWLEDGE, STATUS_ENQUIRY und STATUS.

RESTART:
Mit einem RESTART kann ein Endgerät oder eine Netzkomponente ihre benachbarte Komponente auffordern, die für eine virtuelle Verbindung reservierten Ressourcen freizugeben. Die Verbindung muß danach neu etabliert werden. Es ist auch möglich, mit einer RESTART-Nachricht *alle* Verbindungen zu einem benachbarten System freizugeben. Die RESTART-Nachricht hat jedoch nur lokale Bedeutung[28] und kann daher nicht genutzt werden, um Betriebsmittel bei entfernten Knoten freizugeben.

Da mit RESTART nur Verbindungen abgebaut werden können, von denen das die RESTART-Nachricht sendende System ebenfalls betroffen ist, sind durch RESTART-Nachrichten keine "Denial of Service"-Angriffe möglich, die nicht auch ohne RESTART über RELEASE-Signalisierungsnachrichten möglich wären. Das Filtern von RESTART-Nachrichten kann die Sicherheit daher nur geringfügig erhöhen.

STATUS_ENQUIRY und STATUS:
Mit der STATUS_ENQUIRY-Nachricht kann jederzeit der Zustand einer

[28] Nachrichten mit lokaler Bedeutung werden nur zwischen benachbarten ATM-Komponenten ausgetauscht.

Signalisierung bei der benachbarten ATM-Komponente abgefragt werden. Die Anfrage muß mit einer STATUS-Nachricht beantwortet werden. STATUS-Nachrichten können darüber hinaus auch jederzeit für Fehlermeldungen gesendet werden.

Diese Nachrichten haben nur lokale Bedeutung, sie können daher nicht ausgenutzt werden, um den Zustand von Verbindungen zwischen entfernten Systemen auszuforschen. Der Zustand von Verbindungen zu *benachbarten* Systemen kann prinzipiell jedoch auch aus dem Zustand der Verbindung im eigenen System abgeleitet werden. Daher ist eine Zugriffskontrolle für STATUS_ENQUIRY- und STATUS-Nachrichten durch einen Signalisierungsfilter nicht sinnvoll.

Denkbar wäre hingegen die Verwendung von Statusanfragen durch den Signalisierungsfilter. Der Signalisierungsfilter bräuchte dann nicht selbst den Zustand aller Verbindungen zu verfolgen. Um in seltenen Fällen eine Zugriffskontrollentscheidung vom Zustand anderer Verbindungen abhängig zu machen, kann es vielfach ausreichen, den Zustand anderer Verbindungen über Statusanfragen bei einem benachbarten Switch zu erfragen.[29] Die Anwendbarkeit dieser Lösung ist auf spezielle Szenarien beschränkt, da die Zugriffskontrollentscheidungen des Signalisierungsfilters durch die Angaben des Switches beeinflußt werden und daher die erreichte Sicherheit auch vom Switch abhängt.

Zugriffskontrollregeln können sich auf einzelne IEs beziehen. In den meisten Fällen müssen jedoch Kombinationen mehrerer IEs ausgewertet werden. Bereits für die Einschränkung der Kommunikation zwischen bestimmten Endsystemen werden zwei IEs benötigt ("Calling Party Number" und "Called Party Number"). Auch zur Abwehr von "Spoofing"-Angriffen wird eine kombinierte Filterung von "Calling Party Number"-IE und DTL-IE benötigt. Weitere sinnvolle Filterungsmöglichkeiten, für die mehrere IEs gemeinsam ausgewertet werden müssen, sind z. B. Beschränkungen der Dienstgüte, die von bestimmten Endsystemen oder für bestimmte "Native"-ATM-Anwendungen angefordert werden dürfen. Schließlich können sich auch durch das Fehlen bestimmter IEs wichtige Informationen für einen Filter ergeben. Zum Beispiel ist bei CLIP immer die Angabe der Absender-Adresse ("Calling Party Number") erforderlich [Perez et al. 95, S. 7]. SETUP-Nachrichten die im BLLI-IE eine CLIP Verbindung identifizieren, aber kein "Calling Party Number"-IE enthalten, müssen vom Signalisierungsfilter blockiert werden. Aufgrund der vielfältigen Möglichkeiten, auf der Basis von Signalisierungsnachrichten und ihren unterschiedlichen Informationselementen eine Zugriffskontrolle durchzuführen, wird für die Spezifikation von Filterregeln eine aufwendige Filtersprache benötigt.

[29] Der Signalisierungsfilter kann bei dieser Lösung nicht vollständig zustandslos sein, da bei einer STATUS_ENQUIRY-Nachricht eine "Call Reference" angegeben werden muß.

3.3.1.1.2 Mögliche Aktionen eines Signalisierungsfilters

Die Reaktionsmöglichkeiten eines Signalisierungsfilters gehen über die beschränkten Möglichkeiten einer Packet Screen ("permit" oder "deny") weit hinaus. Auf der Basis der ausgewerteten Nachrichten kann ein Signalisierungsfilter folgendermaßen verfahren:

Signalisierung gewähren: Wenn die Signalisierungsnachricht den Sicherheitsrichtlinien gehorcht, darf die Signalisierungsnachricht passieren.

Signalisierung ablehnen: Wenn die Signalisierungsnachricht gegen die Sicherheitsrichtlinien verstößt, kann eine Ablehnung der Nachricht erfolgen. Auf eine zum Verbindungsaufbau verwendete SETUP-Nachricht muß der Signalisierungsfilter dann mit einem RELEASE_COMPLETE reagieren.

Entfernen von Informationselementen: Statt einer Ablehnung der Signalisierungsnachricht kann es auch ausreichen, nicht zulässige IEs zu entfernen und die entsprechend verkürzte Signalisierungsnachricht weiterzuleiten. Ein IE kann dabei nicht akzeptabel sein, weil die im IE angegebenen Werte gegen die Sicherheitsrichtlinie verstoßen oder weil das IE insgesamt nicht erlaubt ist. Das Entfernen eines IEs ist natürlich nur bei IEs möglich, die optional in der Signalisierungsnachricht enthalten sind. Erforderliche IEs ("mandatory") dürfen auf diese Weise nicht entfernt werden. Außerdem ist das Entfernen des IEs nur dann empfehlenswert, wenn die Signalisierungsnachricht auch ohne dieses IE sinnvoll ist und eine resultierende Verbindung trotzdem von den Kommunikationspartner für den vorgesehenen Zweck verwendet werden kann. Eine sinnvolle Anwendung dieser Maßnahme ist zum Beispiel das Entfernen einzelner BLLI-IEs aus einer SETUP-Nachricht, wenn vom Sender verschiedene BLLI-IEs zur Auswahl gestellt werden. Auch dem Wunsch eines Endgerätes, einen bestimmten "Provider" zu verwenden, kann durch Entfernen des "Transit Network Selection"-IEs widersprochen werden.

Für folgende Reaktionsmöglichkeiten ist die erweiterte Funktionalität eines Signalisierungskontrollers erforderlich:

Modifizieren von Informationselementen: Das Entfernen eines IEs ist in den meisten Fällen nicht möglich, weil die Verwendung des entsprechenden IE vorgeschrieben ist oder weil die Signalisierungsnachricht ohne dieses IE nicht mehr sinnvoll ist. In diesen Fällen kann das IE durch Modifikationen an die Sicherheitsrichtlinien angepaßt werden. So ist es zum Beispiel

denkbar, die angeforderten Dienstgüteparameter soweit zu reduzieren, bis sie den Sicherheitsrichtlinien entsprechen.[30]

Umleiten von Verbindungen: Das Umleiten einer Verbindung stellt einen Sonderfall des Modifizierens von IEs dar. Verstößt ein Verbindungsaufbau zu einem bestimmten Endgerät gegen die Sicherheitsrichtlinien, kann der Signalisierungskontroller die Verbindung zu einem anderen Endgerät umleiten. Der Signalisierungskontroller kann auch einen alternativen Pfad zum gewünschten Endgerät wählen, um Sicherheitskomponenten, wie z. B. eine Packet Screen, einzubinden. Das Einbinden von Sicherheitskomponenten wird im Abschnitt 3.3.1.2.2 beschrieben.

Beenden von Verbindungen: Eine weitere Reaktionsmöglichkeit eines Signalisierungskontrollers ist das Beenden einer Verbindung. Verstößt zum Beispiel ein System durch das Hinzufügen unberechtigter Teilnehmer zu einer Punkt-zu-Mehrpunkt-Verbindung gegen die Sicherheitsrichtlinien, so kann es empfehlenswert sein, die Punkt-zu-Mehrpunkt-Verbindung insgesamt zu beenden.

Eine weitere sehr sinnvolle Anwendung ist die Isolation von Angreifern. Sobald ein Signalisierungskontroller erkannt hat, daß von einem Endsystem Angriffe ausgehen, kann er dieses Endsystem isolieren, indem der Signalisierungskontroller alle Verbindungen zu diesem System beendet. Die Aufforderung, ein bestimmtes System zu isolieren, kann auch durch eine andere Sicherheitskomponente (z. B. Packet Screen oder Proxy) erfolgen.

Hinzufügen von zusätzlichen Teilnehmern: Das Abbrechen von Verbindungen zum Isolieren eines Systems verhindert in vielen Fällen die Aufklärung des Angriffs. Anstelle des Abbrechens von Verbindungen kann es sinnvoller sein, eine Monitor-Komponente auf eine Verbindung aufzuschalten. Bei einem Monitor auf einer CLIP-Verbindung würde es dann ausreichen, nur die ersten zwei Zellen jedes AAL-5-Rahmens aufzuzeichnen, um die IP-Adressen und die zugegriffenen Dienste identifizieren zu können. Das Aufschalten eines Monitors auf eine Verbindung kann jedoch nicht als normale Punkt-zu-Mehrpunkt-Verbindungen implementiert werden, da diese immer unidirektional sein müssen. Stattdessen ist eine spezielle Konfiguration der virtuellen Kanäle zum Monitor hin erforderlich.

[30] Problematisch ist bei der Reduktion von Dienstgüteparametern, daß der Sender der Signalisierungsnachricht im allgemeinen nicht über die veränderten Dienstgüte-Eigenschaften der virtuellen Verbindung benachrichtigt werden kann. Dies kann zum Beispiel dazu führen, daß der Sender anschließend mit einer zu hohen Zellrate sendet. Ein Ablehnen des Verbindungsaufbaus durch einen Signalisierungsfilter wäre daher in diesem Fall einem Modifizieren von IEs vorzuziehen.

3.3.1.1.3 Realisierungsmöglichkeiten von Signalisierungsfiltern

Für die Realisierung einer Zugriffskontrolle auf Signalisierungsnachrichten in einem Signalisierungsfilter, -proxy oder -kontroller sind zwei verschiedene Ansätze denkbar: integriert in einen Switch oder als zusätzliche Komponente auf der Leitung zwischen zwei Switches.[31] Da der Switch als Komponente bereits für die normale Funktion des ATM-Netzes erforderlich ist und der Switch zudem bereits an der Signalisierung teilnimmt, verspricht die Integration eines Signalisierungsfilters in einen ATM-Switch den geringsten Aufwand. Die bereits in einigen Switches verfügbaren einfachen Filter auf ATM-Adressen sprechen dafür, daß auch die hier vorgestellte erweiterte Filterfunktionalität in Zukunft von Switches erbracht wird.

Bei der Integration eines Signalisierungsfilters in einem Switch muß unterschieden werden nach der Position des Filters. Wie bei der Packet Screen Funktion eines Routers kann auch der Signalisierungsfilter an der eingehenden oder an der ausgehenden Schnittstelle des Switches die Signalisierungsnachrichten kontrollieren. Auch eine Integration des Filterprozesses zwischen den beiden Schnittstellen ist denkbar. Die Erkennung von "Spoofing"-Angriffen macht es beim Signalisierungsfilter (wie auch bei der Packet Screen) erforderlich, bei der Filterung die Schnittstelle zu berücksichtigen, über die die Signalisierungsnachricht (bzw. das IP-Datagramm bei einer Packet Screen) empfangen wurde.

Die Regeln können sich abhängig von der Position des Signalisierungsfilters stark unterscheiden. Offensichtlich wird dies beim ADD_PARTY zum Ergänzen weiterer Teilnehmer einer Punkt-zu-Mehrpunkt-Verbindung. Diese ADD_PARTY-Nachricht wird nur bis zu dem Switch verwendet, bis zu dem die Punkt-zu-Mehrpunkt-Verbindung bereits besteht (Switch B in Abbildung 3.3). Bis zu diesem Switch sind auch bereits die entsprechenden Betriebsmittel reserviert. Ab diesem Switch muß bis zum neuen Teilnehmer (Endsystem Z in Abbildung 3.3) eine neue virtuelle Verbindung aufgebaut werden. Für diesen Verbindungsaufbau wird eine SETUP-Nachricht verwendet. Ein Signalisierungsfilter auf der eingehenden Schnittstelle von Switch B (Position 1) sieht eine ADD_PARTY-Nachricht, während auf der ausgehenden Schnittstelle (Pos. 3) eine SETUP-Nachricht kontrolliert werden muß. Ein Signalisierungsfilter im Switch (Pos. 2) muß hingegen eine Umsetzung von ADD_PARTY nach SETUP machen.

Obwohl zukünftig von der Integration von Signalisierungsfiltern in ATM-Switches auszugehen ist, bietet diese Lösung keine akzeptable Plattform für die Erprobung der hier diskutierten erweiterten Filterungsmöglichkeiten. Die proprietäre Software, die die Signalisierung im Switch steuert, kann nicht ohne erhebliche Schwierigkeiten durch eigene Implementationen ersetzt werden. Daher wurde überprüft,

[31] Soll ein bestimmtes Endgerät kontrolliert werden, wäre entsprechend die Integration als zusätzliche Komponente zwischen Switch und Endgerät erforderlich.

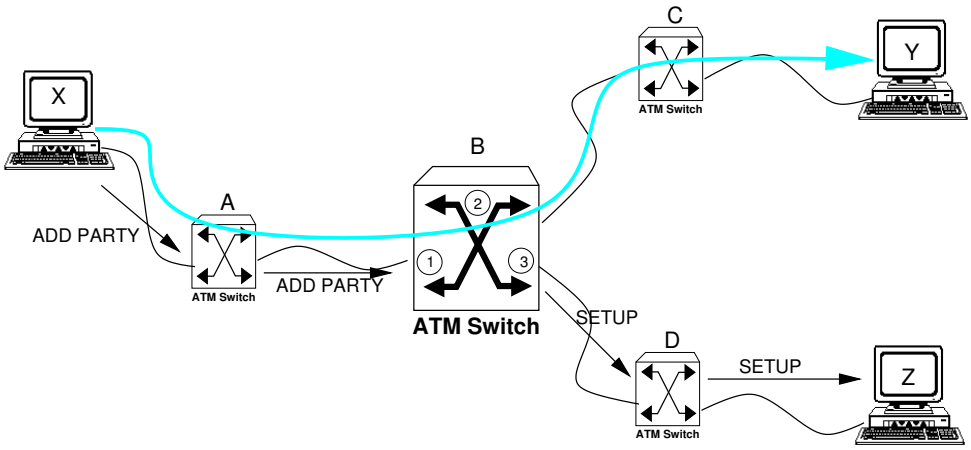


Abbildung 3.3: Alternative Positionen für Signalisierungsfilter im Switch B

ob es möglich ist, Signalisierungsnachrichten an der eingehenden Schnittstelle abzufangen und an eine Workstation weiterzuleiten, auf der ein Prototyp eines Signalisierungsfilters implementiert ist. Nach der Filterung könnte die Signalisierungsnachricht an die Signalisierungssoftware des Switches übergeben werden. Da Signalisierungsnachrichten immer auf vorgegebenen PVCs übertragen werden, erscheint diese Lösung prinzipiell implementierbar zu sein. Für die PNNI-Signalisierung wird immer der PVC (VPI=0, VCI=18) und für UNI der PVC (VPI=0, VCI=5) verwendet. Diese PVCs werden, wie in Abbildung 3.4 dargestellt, im Switch ebenfalls über PVCs von der Schnittstelle zum CPU-Modul weitergeleitet. Durch Ersetzen des PVCs zwischen Schnittstelle und CPU-Modul durch zwei PVCs von der Schnittstelle zum Signalisierungsfilter und vom Signalisierungsfilter zum CPU-Modul könnte theoretisch ein Signalisierungsfilter eingebunden werden (s. Abb. 3.5). Eine Workstation liest die Daten vom PVC ein, zeichnet sie auf (Audit der Signalisierung), überprüft die Signalisierungsnachricht anhand der Sicherheitsrichtlinien (Zugriffskontrolle) und manipuliert die Nachricht gegebenenfalls, bevor die Nachricht wieder an den Switch zurückgesendet wird. Auf diese Weise stünde ein Workstation-basierter Signalisierungsfilter als flexible Plattform für erweiterte Funktionen (zum Beispiel zur Lastverteilung) zur Verfügung. Die gewonnenen Erkenntnisse könnten dann später bei der Integration dieser Funktionen in den Switch verwendet werden. Leider kann diese Idee mit dem zur Verfügung stehenden “Cisco Lightstream 1010” Switch nicht implementiert werden, da das Ersetzen der für die Signalisierung dienenden PVCs auf die zum Signalisierungsfilter führenden Schnittstelle nicht möglich ist. Dieses Ersetzen wird vermutlich deshalb verhindert, weil das CPU-Modul zur Verarbeitung der Signalisierungsnachrichten auch Informationen benötigt, über welche Schnittstelle die Signalisierungsnachricht den Switch erreicht hat. Diese Informa-

tionen leitet das CPU-Modul vermutlich aus dem vorkonfigurierten PVC ab. Da dies nach der Ersetzung durch zwei über den Signalisierungsfilter laufende PVCs nicht mehr möglich wäre, wird das Ersetzen dieser PVCs vom Switch verhindert.

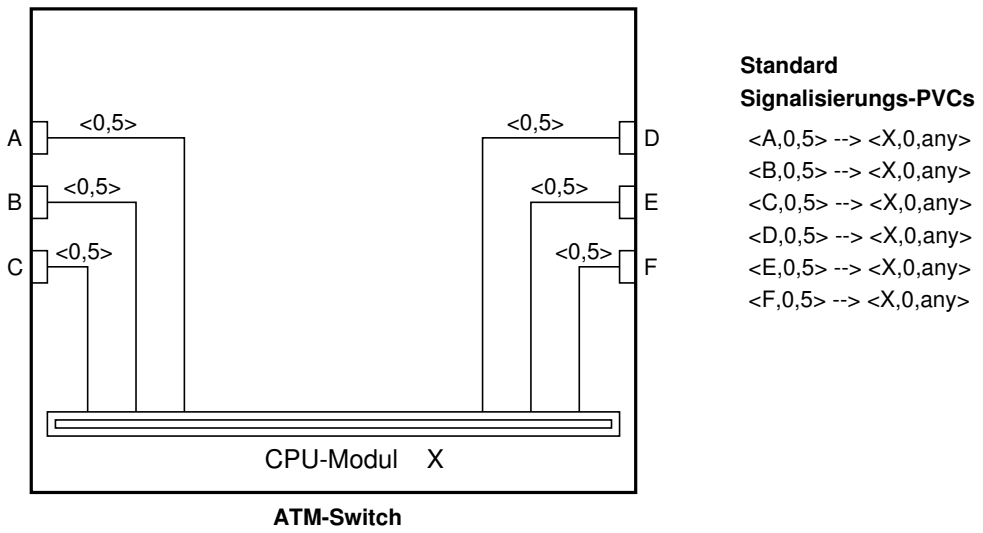


Abbildung 3.4: PVCs für UNI-Signalisierung in einem Switch

3.3.1.1.4 Anwendungsbereiche von Signalisierungsfiltern

Die Zugriffskontrolle eines Signalisierungsfilters beschränkt sich auf den Auf- und Abbau von Verbindungen. Die auf den Verbindungen übertragenen Daten können vom Signalisierungsfilter nicht überwacht werden. Der Einsatzbereich von Signalisierungsfiltern beschränkt sich daher auf zwei wesentliche Aspekte:

- Ein Signalisierungsfilter kann dafür eingesetzt werden, Verbindungen externer Systeme nur zu besonders abgesicherten internen Systemen zuzulassen. Solche besonders abgesicherten Systeme werden bei Firewall-Architekturen als Bastionen bezeichnet. In einem ATM-Netz könnten diese besonders abgesicherten Endsysteme auch andere Firewall-Komponenten sein, die zur Überwachung der übertragenen Daten dienen (z. B. eine Packet Screen).

- Ein Signalisierungsfilter kann den Aufbau von virtuellen Verbindungen unterbinden, die von bestimmten ATM-Adressen ausgehen oder die bestimmte ATM-Adressen als Ziel haben. Dadurch wird es möglich, eine Menge von ATM-Adressen zu einem virtuellen Netz zu vereinen und dieses Netz gleichzeitig gegenüber anderen Systemen abzuschotten, obwohl die Kommunikation über dasselbe physikalische ATM-Netz geführt wird. Auf die-

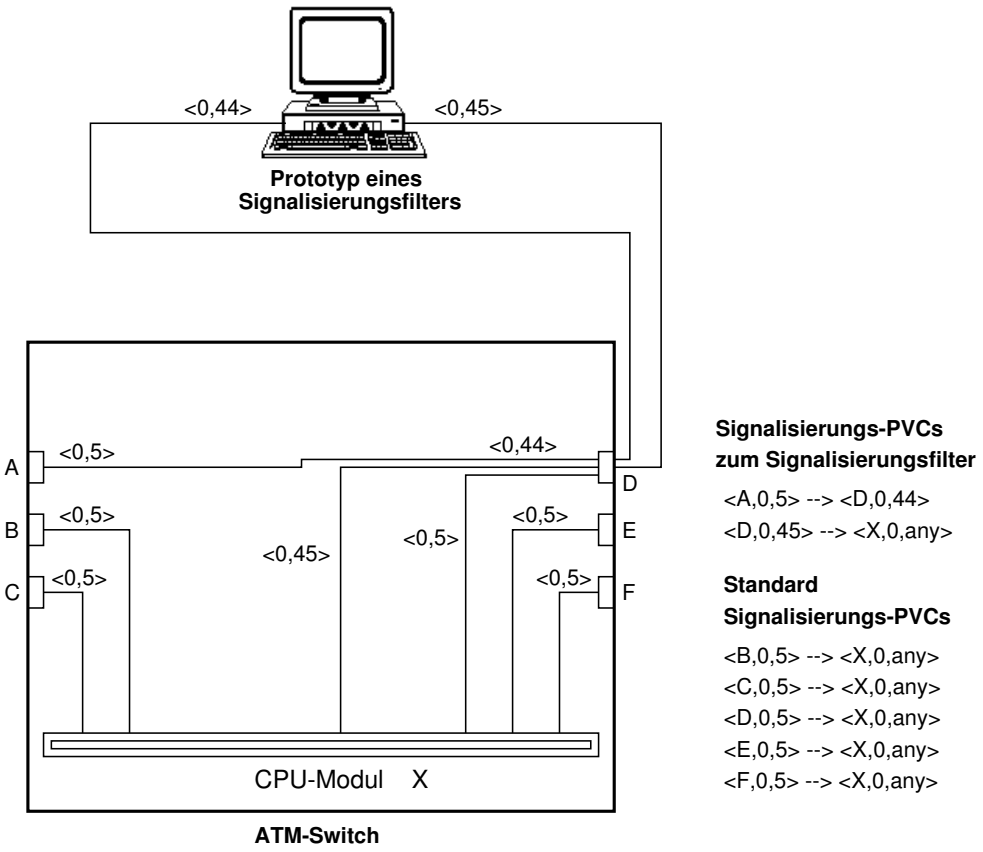


Abbildung 3.5: Anschluß eines Prototyps eines Signalisierungsfilters

se Weise kann ein Signalisierungsfilter auch dafür eingesetzt werden, ausschließlich Verbindungen zwischen Systemen zuzulassen, die auf demselben Sicherheitsniveau liegen. Dadurch wird es möglich, auf demselben physikalischen ATM-Netz mehrere logisch voneinander getrennte Netze zu etablieren. Die Kommunikation zwischen den sogenannten virtuellen Netzen muß dann über Firewalls abgesichert werden. Virtuelle Netze werden im Abschnitt 3.4.5 diskutiert.

Der Signalisierungsfilter kann in einer Firewall-Architektur die Packet Screen ersetzen. Bei Kombinationen von Packet Screen und Bastion liegt die Hauptaufgabe der Packet Screen darin, direkte Verbindungen zwischen internem und externem Netz zu unterbinden und nur Verkehr von und zur Bastion zuzulassen. Die bei einer Packet Screen im Prinzip zur Verfügung stehende Möglichkeit, auch Zugriffe auf bestimmte Ports der Bastion einzuschränken, kann zwar in einigen Fällen eine höhere Sicherheit bieten, ist jedoch nicht zwingend erforderlich, wie

z. B. die Kombination einer Packet Screen mit äußerer Bastion zeigt (s. Abb. 2.9). Der Signalisierungsfilter kann durch die Zugriffskontrolle beim Aufbau virtueller Verbindungen die gleiche Wirkung wie eine Packet Screen erzielen, ohne die auf der virtuellen Verbindung übertragenen Dateneinheiten untersuchen zu müssen.

Da Packet Screens in herkömmlichen Netzen häufig in einen Router integriert werden, ist bei der Kombination von Packet Screen und Bastion in der Regel die Einrichtung eines getrennten Subnetzes für die Bastion(en) erforderlich. Wird ein Signalisierungsfilter statt der Packet Screen mit der Bastion kombiniert, ist hingegen kein eigenes Subnetz mehr für die Bastion erforderlich, da die Filterregeln im Signalisierungsfilter anhand von ATM-Adressen konfiguriert werden können, ohne daß eine Verknüpfung der Filterfunktionalität mit der Funktion eines Routers erforderlich wird.

Die Kontrolle des Signalisierungsfilters erfolgt nur beim Verbindungsauf- und -abbau. Dadurch kann der Signalisierungsfilter bei der eigentlichen Datenübertragung nicht zu negativen Auswirkungen auf die Performanz führen. Ein Signalisierungsfilter wird nur in Umgebungen zu einem Performanzengpaß, in denen sehr häufig virtuelle Verbindungen auf- und abgebaut werden. Da sich ATM-Netze jedoch ohnehin nicht für Umgebungen eignen, in denen kurzlebige Verbindungen häufig auf- und abgebaut werden, hat dieser potentiell vorhandene Engpaß in der Praxis keine Relevanz.

Für den Einsatz in Firewall-Architekturen ist davon auszugehen, daß ein Signalisierungsfilter am Übergang zwischen internem und externem Netz eingesetzt wird, um Verbindungen zwischen internen und externen Systemen lediglich für ausgewählte interne Endsysteme zuzulassen. Auf diesen Endsystemen werden dann die zur Kontrolle der übertragenen Daten erforderlichen Firewall-Komponenten eingesetzt. Dabei müssen diese Firewall-Komponenten nicht zentral in unmittelbarer Nähe des Signalisierungsfilters installiert werden. Sehr leistungsfähige Firewall-Konzepte ergeben sich stattdessen aus einer dezentralen Installation dieser Firewall-Komponenten. Unterschiedliche Firewall-Architekturen für ATM-Netze werden im Abschnitt 3.4 vorgestellt.

Ein Signalisierungs*kontroller* eignet sich zudem hervorragend für die Lastverteilung, da über die Funktionalität von Signalisierungs*filtern* hinausgehend virtuelle Verbindungen auch umgeleitet werden können. Da alle Verbindungen vom Signalisierungskontroller überwacht werden, kann dieser aus der Anzahl der virtuellen Verbindungen und der für die Verbindungen reservierten Dienstgüte Rückschlüsse auf die aktuelle Last der jeweiligen Bastion ziehen. Noch genauere Informationen erhält der Signalisierungskontroller aus den Verkehrsstatistiken des Switches oder durch direktes Abfragen der Last bei der Bastion. Über das Umleiten von neu eintreffenden Verbindungen auf weniger ausgelastete Bastionen kann ein Signalisierungskontroller in die Lastverteilung eingreifen. Die Verbindungsaufbauzeit

muß sich durch diese Lastverteilungsfunktion des Signalisierungskontrollers nicht erhöhen, da die für die Lastverteilung notwendigen Informationen bereits vorliegen und die Entscheidung somit nur geringfügige Verzögerungen verursacht.

3.3.1.1.5 Grenzen von Signalisierungsfiltern

Über die Kontrolle der Signalisierungsnachrichten kann ein Signalisierungsfilter kontrollieren, zwischen welchen Endsystemen virtuelle Verbindungen aufgebaut werden dürfen. Da in einem ATM-Netz eine Kommunikation nur über virtuelle Verbindungen möglich ist, wird durch die Kontrolle der Signalisierung bereits eine weitgehende Kontrolle der Kommunikation erreicht.

Da die Übertragung von Daten entlang einer bestehenden virtuellen Verbindung anschließend bis zum Verbindungsabbau keine weitere Signalisierung mehr erfordert, ist durch einen Signalisierungsfilter keine Überprüfung der übertragenen Daten möglich. Welche Daten auf einer virtuellen Verbindung übertragen werden, kann durch einen Signalisierungsfilter kaum gesteuert werden. Am besten ist noch die Kontrolle bei "Native"-ATM-Anwendungen möglich, wo über das BHLI-IE angegeben wird, durch welche Anwendung die virtuelle Verbindung bearbeitet werden soll. Die Filterungsmöglichkeiten des Signalisierungsfilters entsprechen hier praktisch den Möglichkeiten einer Packet Screen, wenn dort Filter eingesetzt werden, die bestimmte Portnummern spezifizieren. In beiden Fällen unterliegt es aber letztendlich der Kontrolle des Endsystems, durch welche Anwendung die Verbindung bearbeitet wird. Die Informationen im BHLI-IE oder im Transport-Header können daher nicht für eine sichere Identifizierung eines Dienstes genutzt werden.

Noch weniger Kontrolle über die zugegriffenen Dienste hat ein Signalisierungsfilter, wenn die virtuelle Verbindung für die Übertragung von Dateneinheiten der Schicht 2 oder Schicht 3 genutzt wird, wie es bei LANE, MPOA, CLIP oder NHRP der Fall ist. Bei CLIP enthalten zum Beispiel erst die über die virtuelle Verbindung übertragenen IP-Dateneinheiten die IP-Adresse des Empfängers und die Portnummer des zugegriffenen Dienstes. Dabei muß das über die ATM-Adresse identifizierte System nicht zwangsläufig auch der Empfänger der IP-Dateneinheit sein, denn das ATM-Endsystem kann die empfangenen IP-Dateneinheiten an ein anderes System weiterleiten ("Forwarding"). Router und "Edge-Devices" sind in der Regel für mehrere (IP-basierte) Systeme zuständig, an die IP-Dateneinheiten weitergeleitet werden. Auch einfache UNIX-Workstations führen ein "Forwarding" aus, wenn sie über mehrere Netzwerkschnittstellen verfügen.[32] Das "Forwarding" von IP-Dateneinheiten, die eine Empfänger-Adresse haben,

[32] Da eine Workstation heute meist standardmäßig über eine Ethernet-Schnittstelle verfügt, hat eine an ein ATM-Netz angeschlossene Workstation in den meisten Fällen zwei Netzwerkschnittstellen (ATM und Ethernet), so daß dieses Problem sehr häufig auftritt.

die nicht mit dem eigenen System übereinstimmt, ist abschaltbar; jedoch kann sich ein Signalisierungsfilter nicht darauf verlassen, daß diese Funktion bei einem Endgerät abgeschaltet ist. Somit dürfen durch einen Signalisierungsfilter nur virtuelle Verbindungen zu Endsystemen zugelassen werden, die entsprechend auf ihre Sicherheit hin kontrolliert wurden.

Beim Einsatz eines SNAP-Headers, wie er bei CLIP in der Regel verwendet wird, oder bei der Übertragung von Schicht-2-Rahmen bei "LAN Emulation" kann der Signalisierungsfilter nicht einmal feststellen, ob anschließend über die virtuelle Verbindung IP-Dateneinheiten oder Dateneinheiten anderer Protokolle übertragen werden. Auch in diesen Fällen ist die Überwachung der Endsysteme erforderlich, um eine akzeptable Sicherheit erreichen zu können.

Ein weiteres Problem ergibt sich dadurch, daß die Richtung des Zugriffs nicht kontrolliert werden kann. Der Signalisierungsfilter kann zwar die Richtung des Aufbaus der virtuellen Verbindung feststellen. Damit ist jedoch nicht festgelegt, auf welcher Seite anschließend auf Dienste zugegriffen wird. Nachdem eine virtuelle Verbindung für CLIP, LANE, NHRP oder MPOA aufgebaut ist, kann diese Verbindung für den Austausch beliebiger IP-Dateneinheiten in beiden Richtungen genutzt werden.[33] Öffnet zum Beispiel das Endsystem A eine virtuelle Verbindung zum Endsystem B, um einen Zugriff über CLIP auf den dortigen Web-Server durchzuführen, so kann das Endsystem B über dieselbe virtuelle Verbindung auch auf alle IP-basierten Dienste des Endsystems A zugreifen. Dabei ist die Zeitdauer, über die die virtuelle Verbindung geöffnet bleibt, unabhängig von der Länge des ursprünglichen Zugriffs von A auf B. Der Verbindungsabbau erfolgt erst, wenn die Verbindung für eine längere Zeit nicht benutzt wird.[34] B kann somit durch regelmäßiges Senden die Verbindung für weitere Angriffe offenhalten.

Um mit einem Signalisierungsfilter eine ähnliche Funktionalität wie mit einer Packet Screen erzielen zu können, wurde verschiedentlich vorgeschlagen, die "Header" des ersten Pakets der TCP-Verbindung in die SETUP-Nachricht zu integrieren.[35] Ein Signalisierungsfilter könnte dann anhand dieser Header-Informationen eine Zugriffskontrolle wie eine Packet Screen durchführen. Dieser Ansatz setzt eine enge Verknüpfung zwischen der virtuellen Verbindung und der über CLIP geführten TCP-Verbindung voraus. Dies erfordert den Aufbau einer

[33] Virtuelle Verbindungen können auch unidirektional sein. Falls Daten vom Empfänger zurückgesendet werden sollen, wird eine zweite Verbindung benötigt. Beim Einsatz für IP-Verkehr bilden unidirektionale Verbindungen jedoch die Ausnahme.

[34] RFC 1755 [Perez et al. 95] empfiehlt, ungenutzte Verbindungen nach 20 Minuten zu schließen. Implementationsabhängig dürfen hier auch andere Werte gewählt werden.

[35] Alle verfügbaren Referenzen verweisen auf Dokumente, bei denen es sich vermutlich um Entwürfe gehandelt hat, die nicht (mehr) öffentlich verfügbar sind. Der älteste Hinweis auf den hier beschriebenen Ansatz fand sich in den Archiven der Mailing-Liste "`atm@matmos.hpl.hp.com`" in einer Email von Earl E. McCoy (`emccoy@tra.com`) am 2. April 1994.

neuen virtuellen Verbindung für jede neue TCP-Verbindung, und nach dem Abbau der TCP-Verbindung muß auch die virtuelle Verbindung beendet werden. Bereits der TCP-Verbindungsaufbau allein führt bei Anwendungen mit häufigem Verbindungsauf- und -abbau (z. B. Web-Zugriffe) zu einem inakzeptablen Performanzverhalten. Wenn hier zusätzlich zur TCP-Verbindung auch noch die virtuelle Verbindung jedesmal neu aufgebaut werden muß, erhöhen sich die Verbindungsaufbauzeiten noch einmal erheblich, so daß Anwendungen mit häufigem Verbindungsauf- und -abbau nicht mehr sinnvoll eingesetzt werden können. Für UDP-basierte Anwendungen kann dieser Lösungsansatz nicht verwendet werden, da bei UDP kein expliziter Verbindungsabbau stattfindet, nach dem auch die virtuelle Verbindung geschlossen werden könnte. Neben den Performanzproblemen kann dieser Lösungsansatz auch keine ausreichende Sicherheit bieten, da nicht verhindert werden kann, daß ein Endsystem beim Verbindungsaufbau andere Header-Informationen sendet als tatsächlich anschließend über die virtuelle Verbindung übertragen werden. Dieser Ansatz ist somit auch aus Sicherheitsgesichtspunkten sinnlos.[36]

Fazit: Durch die Zugriffskontrolle eines Signalisierungsfilters beim Aufbau einer virtuellen Verbindung kann nur der Zugriff zwischen bestimmten ATM-Endsystemen eingeschränkt werden. Die bei der Signalisierung übertragenen Informationen ermöglichen nur eine schwache Kontrolle über die anschließende Verwendung einer virtuellen Verbindung. Um eine bessere Kontrolle der übertragenen Daten zu ermöglichen, ist die Kombination mit weiteren Firewall-Komponenten wie Packet Screens oder Proxies erforderlich.

3.3.1.2 Erweiterte Möglichkeiten eines Signalisierungskontrollers

Über die Funktionen eines Signalisierungsfilters hinausgehend hat ein Signalisierungskontroller auch die Möglichkeit, Informationselemente beim Aufbau einer virtuellen Verbindung zu ändern und so zum Beispiel alternative Pfade zu wählen, um zusätzliche Sicherheitskomponenten einzubinden.

[36] Die Sicherheitsprobleme können durch die Kombination mit einem geänderten Format zur Übertragung von IP-Dateneinheiten auf der virtuellen Verbindung gelöst werden. Dafür dürfen, ähnlich wie beim "IP-Switching" (s. Abschnitt 2.4.5), die Header-Informationen, die bei Signalisierung gesendet werden, nicht mehr auf der virtuellen Verbindung übertragen werden. Wenn bei der Signalisierung die IP-Adressen und die TCP-Portnummern übertragen werden, so müssen diese Felder vom Sender aus dem Datagramm entfernt werden. Erst der Empfänger fügt die über die SETUP-Nachricht erhaltenen IP-Adressen und Portnummern zu jedem empfangenen Datagramm wieder hinzu. Auf diese Weise hat der Sender nicht mehr die Möglichkeit, beim Verbindungsaufbau andere Adressen anzugeben als später tatsächlich verwendet werden. Die Performanzprobleme durch wiederholten Verbindungsauf- und -abbau bei kurzen Datenflüssen treten auch bei dieser Lösung unverändert auf.

Der Pfad durch ein ATM-Netz wird von dem Switch bestimmt, der vom Endgerät die Aufforderung erhalten hat, eine Verbindung zu einem bestimmten Endgerät aufzubauen. Der Switch berechnet anhand der im voraus über PNNI ermittelten Topologie-Informationen eine Liste möglicher Pfade zum gewünschten Empfänger. Von dieser Liste werden die Pfade gestrichen, die die geforderte Dienstgüte (Zellrate, Zellverzögerung, usw.) nicht erfüllen können. Von den übrigen Pfaden wird der günstigste Pfad ausgewählt. Der günstigste Pfad kann zum Beispiel der Pfad sein, der die niedrigsten Kosten verursacht. Ohne eine Veränderung des dargestellten Ablaufs können Sicherheitskomponenten in den Pfad integriert werden, wenn bei der Dienstgüte neben den Performanzanforderungen auch Sicherheitsaspekte berücksichtigt werden. Aus der Liste möglicher Pfade müssen dann auch die Pfade gestrichen werden, die die geforderte Sicherheitsdienstgüte nicht erreichen.

3.3.1.2.1 Sicherheitsdienstgüte (SQoS)

Die Sicherheitsdienstgüte ("Security Quality of Service", SQoS)[37] ist eine Ergänzung zur herkömmlichen Dienstgüte ("Quality of Service", QoS). Die Dienstgüte beschreibt die Eigenschaften eines Netzes und wird für Knoten ("Nodes") und Leitungen ("Links") angegeben. Für die Dienstgüte von Knoten und Leitungen unterscheidet PNNI jeweils noch einmal zwischen Attributen und Metriken. Die in den Metriken angegebenen Dienstgüteparameter werden für eine virtuelle Verbindung über den gesamten Pfad vom Sender zum Empfänger aufaddiert. Ein Beispiel für eine Metrik ist die maximale Verzögerung bei der Übertragung einer Zelle ("Maximum Cell Transfer Delay", MCTD). Ein Attribut beschreibt hingegen eine bestimmte Eigenschaft eines Pfades. Eine Sicherheitsdienstgüte läßt sich am besten als Attribut spezifizieren, wobei Leitungsattribute ("Link Attribute") für Leitungen mit transparenten Sicherheitskomponenten wie eine zwischen zwei Switches integrierte Packet Screen (s. Abschnitt 3.3.2.2.2) eingesetzt werden sollten, während Knotenattribute ("Nodal Attribute") für Sicherheitskomponenten genutzt werden sollten, die in einem ATM-Netz als Knoten sichtbar sind.

Für die Beschreibung der Sicherheitsdienstgüte werden die im folgenden skizzierten Attribute vorgeschlagen:

[37] SQoS werden bereits in [Varadharajan et al. 97] kurz erwähnt. SQoS bezeichnet dort einen Parameter innerhalb eines bei der Signalisierung verwendeten Informationselements, mit dem kryptographisches Material ausgetauscht werden soll. Die Bezeichnung SQoS beschränkt sich dort daher nur auf Vertraulichkeit, Integrität und Authentizität. Von [Xu et al. 98a] werden hingegen "Firewalling QoS" (FQoS) vorgeschlagen, um virtuelle Verbindungen innerhalb eines Switches zu klassifizieren und durch geeignete Mechanismen (z.B. Packet Screen) zu kontrollieren. Xu verwendet FQoS jedoch ausschließlich innerhalb desselben Switches und erkennt nicht, daß durch die Integration mit PNNI viel flexiblere Lösungen möglich werden.

Protokollüberprüfung: Dieses Attribut beschreibt die Überprüfung der übertragenen Dateneinheiten durch eine Packet Screen.

Inhaltsüberprüfung: Bei einem Pfad oder Knoten mit diesem Attribut werden die übertragenen Dateninhalte überprüft. Diese Aufgabe wird typischerweise von einem Proxy übernommen.

Vertraulichkeit, Integrität und Authentizität: Diese Eigenschaften können durch kryptographische Verfahren erreicht werden. Pfade über ein VPN erhalten abhängig von den eingesetzten kryptographischen Verfahren eines oder mehrere der drei Attribute.

Monitor: Auch die Überwachung einer Verbindung durch einen Netzwerkmonitor kann über die Sicherheitsdienstgüte beschrieben werden.

Die Ergänzung dieser Attribute in PNNI ist relativ einfach. Der wesentliche Unterschied zwischen herkömmlicher Dienstgüte und Sicherheitsdienstgüte entsteht jedoch durch die Art, wie die Dienstgüte angefordert wird. Bei der herkömmlichen Dienstgüte legt das Endsystem die erforderlichen Merkmale fest und das Netz muß daraufhin einen Pfad finden, der den Anforderungen des Endsystems genügt. Die Sicherheitsdienstgüte wird hingegen durch den Signalisierungskontroller bestimmt, der die für das Netz oder für einen Teil des Netzes festgelegten Sicherheitsrichtlinien implementiert. Endsysteme haben lediglich die Möglichkeit, darüber hinausgehende Anforderungen an die Sicherheitsdienstgüte zum Beispiel hinsichtlich der Vertraulichkeit, Integrität und Authentizität anzugeben. Die Anforderungen des Endsystems dürfen dabei nur zu einer Sicherheitsdienstgüte führen, die mindestens das durch die Sicherheitsrichtlinien geforderte Niveau erreicht. Zur Bestimmung der erforderlichen Sicherheitsdienstgüte müssen virtuelle Verbindungen anhand der Sicherheitsrichtlinien klassifiziert werden. Diese Klassifizierung kann am besten durch einen Signalisierungskontroller durchgeführt werden, da dieser bereits eine Sicherheitsrichtlinien-basierte Kontrolle beim Verbindungsaufbau ausübt.

Ein weiterer wichtiger Unterschied zwischen Sicherheitsdienstgüte und herkömmlicher Dienstgüte ist die räumlich begrenzte Bedeutung der Sicherheitsdienstgüte. Während eine angeforderte herkömmliche Dienstgüte für die gesamte Verbindung vom Sender zum Empfänger gilt, wird die Sicherheitsdienstgüte durch die Sicherheitsrichtlinien bestimmt und bezieht sich somit nur auf den Gültigkeitsbereich dieser Sicherheitsrichtlinien. Eine virtuelle Verbindung kann durch verschiedene Teilnetze führen, für die unterschiedliche Sicherheitsrichtlinien festgeschrieben sind. Die Sicherheitsdienstgüte muß dann für jedes Teilnetz erneut bestimmt werden. Dies kann am besten durch die Bildung von "Peer Groups" unterstützt werden, die den gleichen Bereich umfassen wie die Sicherheitsrichtlinien.

Bei der Ergänzung der Sicherheitsdienstgüte in PNNI bleiben mehrere Schwierigkeiten, die noch gelöst werden müssen. Die beiden wichtigsten Punkte werden im folgenden kurz diskutiert:

- Damit die zur Beschreibung der Sicherheitsdienstgüte verwendeten Attribute sicher innerhalb einer "Peer Group" ausgetauscht werden können, ist zunächst eine Authentisierung der zwischen den Knoten ausgetauschten PNNI-"Routing"-Informationen erforderlich. Ohne eine sichere Authentisierung dieser Informationen ist es für einen Angreifer möglich, auf bestimmten Pfaden eine nicht vorhandene Sicherheitsdienstgüte vorzutäuschen. Ein Verfahren zur Authentisierung von PNNI-"Routing"-Informationen muß noch entwickelt werden.

- Der Aufwand für die Auswahl eines Pfades, der der angeforderten Dienstgüte gerecht wird, steigt mit der Anzahl der zu berücksichtigenden Dienstgüteparameter. Da für die Sicherheitsdienstgüte zusätzliche Parameter hinzukommen, entsteht zusätzlicher Aufwand. Eine effiziente Wegwahl unter Berücksichtigung mehrerer unterschiedlicher Dienstgüteparameter ist derzeit noch ein Forschungsgebiet.

Da die Auswahl eines Pfades erst nach der Klassifizierung eines Verbindungsaufbauwunsches möglich ist, entsteht eine enge Verknüpfung zwischen dem Signalisierungskontroller, der die Klassifizierung nach den Sicherheitsrichtlinien durchführt, und PNNI, das einen Pfad in Form einer "Designated Transit List" (DTL) unter Berücksichtigung der erforderlichen Sicherheitsdienstgüte erstellt. Da beide Funktionen in demselben Switch realisiert werden, wäre es möglich, die DTL direkt durch den Signalisierungskontroller verändern zu lassen. Eine Ergänzung der PNNI-Spezifikation bzw. der PNNI-Implementation des Switches wäre in diesem Falle nicht notwendig. Die Topologie-Informationen über die Sicherheitsdienstgüte der jeweiligen Pfade müßten dann jedoch manuell konfiguriert werden. Bei einer Erweiterung von PNNI werden diese Informationen hingegen zusammen mit den übrigen Topologie-Informationen übertragen. Zusätzlich muß der Signalisierungskontroller in beschränktem Rahmen auch Pfade erzeugen können. Somit muß eine wesentliche Komponente von PNNI dupliziert werden. Trotz dieser Einschränkungen ist es sinnvoll, die Sicherheitsdienstgüte-basierte Wegwahl im Rahmen der bestehenden PNNI-Spezifikation zu untersuchen, um die Leistungsfähigkeit dieses neuen Konzeptes zu demonstrieren. Die volle Einsatzfähigkeit einer Sicherheitsdienstgüte-basierten Wegwahl kann jedoch erst durch die Integration in PNNI erreicht werden.

3.3.1.2.2 Sicherheitsdienstgüte-basierte Wegwahl innerhalb der bestehenden PNNI-Spezifikation

Da Erweiterungen von PNNI durch eine Sicherheitsdienstgüte-basierte Wegwahl erst eingesetzt werden können, nachdem eine entsprechend erweiterte Spezifikation verabschiedet wurde, müssen zunächst Übergangslösungen gefunden werden, die eine ähnliche Funktionalität innerhalb der bestehenden PNNI-Spezifikation ermöglichen. In den folgenden Abschnitten soll untersucht werden, wie die Sicherheitsdienstgüte innerhalb der bestehenden PNNI-Spezifikation [ATM Forum 96c] bei der Wegwahl berücksichtigt werden kann. Für die Berücksichtigung der Sicherheitsdienstgüte bei der Wegwahl werden drei verschiedene Ansätze untersucht:

- "Einschleifen" der geforderten Sicherheitskomponente
- Einfügen von Sicherheitskomponenten durch Ersetzen der Empfänger-Adresse
- Sicherheitsdienstgüte-basierte Wegwahl durch Ergänzung oder Änderung der DTL.

3.3.1.2.3 Einschleifen von Sicherheitskomponenten

Das Einbringen von Sicherheitskomponenten in einen Pfad kann beim Verbindungsaufbau ohne Veränderung der PNNI-Nachricht (SETUP-Nachricht) erfolgen, wenn vom Signalisierungskontroller der Aufbau einer virtuellen Verbindung zur Sicherheitskomponente (z. B. einer Packet Screen) hin und wieder zurück zum Signalisierungskontroller veranlaßt werden kann. Die virtuelle Verbindung verläuft dann wie in Abbildung 3.6 dargestellt.[38]

Da der Signalisierungskontroller am Anfang und am Ende des über die Sicherheitskomponente führenden Wegabschnittes steht, kann dieser "Umweg" vom Signalisierungskontroller vor den anderen Switches verborgen werden. Dadurch kann auch die in der DTL vorgegebene Reihenfolge der Switches nach außen hin erhalten bleiben, und selbst die im Signalisierungskontroller enthaltene PNNI-Implementation braucht gegenüber der Implementation in einem herkömmlichen Switch kaum verändert zu werden.[39]

[38] Die Richtung des Verbindungsaufbaus ist in Abbildung 3.6 durch einen Pfeil gekennzeichnet. Die durch die Signalisierung aufgebaute virtuelle Verbindung kann anschließend für unidirektionalen oder bidirektionalen Datenverkehr genutzt werden.

[39] Die Integration von Sicherheitskomponenten stellt ein Problem dar, wenn für eine virtuelle Verbindung QoS-Anforderungen gewährleistet werden müssen. QoS-Aspekte werden an dieser Stelle nicht weitergehend untersucht.

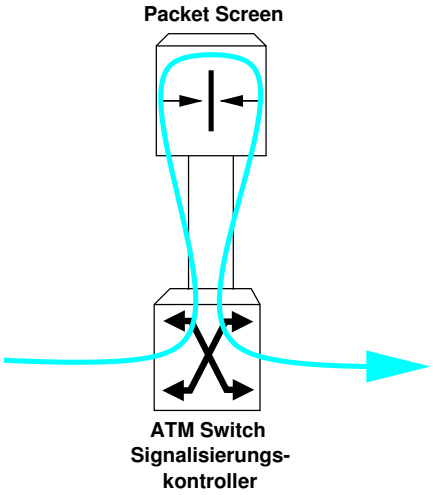


Abbildung 3.6: Sicherheitsdienstgüte-basierte Wegwahl durch "Einschleifen" von Sicherheitskomponenten

Für den Aufbau der über die Sicherheitskomponente führenden Schlaufe sind verschiedene Strategien denkbar. Nimmt die Sicherheitskomponente nicht an der Signalisierung teil, verhält sich also transparent wie die im Abschnitt 3.3.2.2.2 beschriebene "Packet Screen zwischen zwei Switches", dann kann der Signalisierungskontroller einfach ein auf den beiden Schnittstellen zur Sicherheitskomponente ungenutztes VPI/VCI-Paar auswählen und kann dies ohne Signalisierung wie einen implizit konfigurierten PVC benutzen.[40] Eine Signalisierung für einen SVC oder eine explizite Konfiguration eines PVCs ist nicht notwendig, da der Signalisierungskontroller sowohl Initiator als auch Empfänger dieser Verbindung ist.

Hat die Sicherheitskomponente selbst eine Switch-Funktionalität wie zum Beispiel bei der Integration einer Packet Screen in einen Switch (s. Abschnitt 3.3.2.2.3) oder ist diese als ATM-Endgerät implementiert wie zum Beispiel ein Proxy (s. Abschnitt 3.3.3) oder eine klassische Packet Screen (s. Abschnitt 3.3.2.1), dann müssen zwischen Signalisierungskontroller und Sicherheitskomponente virtuelle Verbindungen vereinbart werden. Diese virtuellen Verbindungen können als SVCs über eine Signalisierung[41] aufgebaut werden, oder sie müssen vorab als PVCs konfiguriert werden, von denen der Signalisierungskontroller dann einen

[40] Der Bereich der VPI/VCI-Paare, die vom Signalisierungskontroller ausgewählt werden können, muß der Sicherheitskomponente bekannt sein, wenn die Sicherheitskomponente nicht vollständig transparent ist oder implementationsabhängige Einschränkungen hinsichtlich der VPI/VCI-Paare bestehen, auf denen die Sicherheitskomponente Zellen akzeptieren kann.

[41] Die virtuellen Verbindungen zwischen einem Signalisierungskontroller und einer Sicherheitskomponente auf einem Endsystem müssen über UNI aufgebaut werden.

freien PVC auswählt. Die beiden virtuellen Verbindungen benötigen dann nicht unbedingt zwei getrennte Leitungen zwischen Signalisierungskontroller und Sicherheitskomponente wie in Abbildung 3.6 dargestellt, sondern können auch über dieselbe physikalische Leitung konfiguriert werden. Dies hat den Vorteil, daß ein Port des Signalisierungskontrollers zum Anschluß der Sicherheitskomponente ausreicht, und ist insbesondere dann interessant, wenn eine aufwendige Verarbeitung in der Sicherheitskomponente den Engpaß bildet, so daß die Leitungen von und zur Sicherheitskomponente nicht ausgelastet werden können.

Grundsätzlich stellt sich die Frage, wie eine Verknüpfung zwischen der zur Sicherheitskomponente hingehenden Verbindung und der von der Sicherheitskomponente zurückkommenden virtuellen Verbindung hergestellt werden kann. Für PVCs kann bereits bei der Konfiguration festgelegt werden, welche zwei PVCs jeweils zusammengehören. Bei der Verwendung von SVCs ist die Identifikation zusammengehörender virtueller Verbindungen jedoch problematisch, da gleichzeitig mehrere SVCs vom Signalisierungskontroller zur Sicherheitskomponente hin aufgebaut werden können. Die Sicherheitskomponente muß aber nicht zwangsläufig auch in derselben Reihenfolge die jeweils dazugehörige zweite Verbindung aufbauen. Die Verknüpfung kann daher nur anhand von Informationen in den SETUP-Nachrichten selbst erfolgen. Für diesen Zweck bietet sich die in jeder Signalisierungsnachricht enthaltene "Call Reference" an. Durch das Verwenden derselben "Call Reference" beim Aufbau des zweiten SVCs kann eine Sicherheitskomponente die Zusammengehörigkeit der beiden Verbindungen ausdrücken. Da die "Call Reference" nur lokale Bedeutung zwischen benachbarten Systemen hat, muß die Sicherheitskomponente in diesem Fall direkt an den Signalisierungskontroller angeschlossen sein.

Eine Sicherheitskomponente, die als ATM-Endgerät in die virtuelle Verbindung eingefügt ist, muß nicht für die beiden kommunizierenden Endgeräte sichtbar sein. Wie die Existenz einer als ATM-Endgerät eingefügten Sicherheitskomponente verdeckt werden kann, wird anhand von transparenten Proxies für "Native"-ATM-Anwendungen im Abschnitt 3.3.3.2.3 beschrieben.

Vorteilhaft ist das "Einschleifen" von Sicherheitskomponenten wegen des einfachen Aufbaus des Signalisierungskontrollers. Jedoch geht dies nur auf Kosten der Flexibilität, da der Signalisierungskontroller immer am Anfang und am Ende der Schleife über die Sicherheitskomponente stehen muß.

3.3.1.2.4 Einfügen von Sicherheitskomponenten durch Ersetzen der Empfänger-Adresse

Ein Signalisierungskontroller kann durch Änderung der Empfänger-Adresse ("Called Party Number") in einer SETUP-Nachricht erreichen, daß eine Sicherheitskomponente direkt angesprochen wird. Das die Verbindung initiierende

ATM-Endgerät bemerkt diese Manipulation nicht, da bei der Bestätigung des Verbindungsaufbaus über eine CONNECT-Nachricht keine ATM-Adressen der beteiligten Systeme mehr verwendet werden. Die Adresse des ursprünglich adressierten Empfängers geht hierbei jedoch verloren, so daß die Sicherheitskomponente keine Informationen darüber hat, zu welcher Adresse die zweite Verbindung aufgebaut werden muß. Für die Übermittlung der fehlenden Informationen vom Signalisierungskontroller zur Sicherheitskomponente stehen drei unterschiedliche Wege zur Verfügung:

- Eine Möglichkeit ist es, die fehlenden Informationen in einem zusätzlichen Informationselement zusammen mit der SETUP-Nachricht an die Sicherheitskomponente zu senden. Für die Ergänzung dieses IEs sind nur geringfügige Änderungen der Signalisierungssoftware notwendig. Eine Standardisierung dieses IEs, die einen sehr langwierigen Prozeß voraussetzt, ist nicht zwingend erforderlich, da es sich bei Signalisierungskontroller und Sicherheitskomponente um spezialisierte Komponenten handelt. Bei den zwischen Signalisierungskontroller und Sicherheitskomponente liegenden Switches können Standardkomponenten eingesetzt werden, sofern diese das zusätzliche IE transparent weiterleiten.

- Alternativ zur Übertragung der Adresse des ursprünglich adressierten Empfängers in einem zusätzlichen IE kann vom Signalisierungskontroller mit der "Called Party Subaddress" auch ein bereits spezifiziertes Informationselement verwendet werden. In diesem IE wird normalerweise die Empfänger-Adresse übertragen, wenn beim Verbindungsaufbau über ein öffentliches Netz E.164-Adressen verwendet werden müssen und die Adresse des gewünschten Empfängers nach einem anderen Adressierungsschema vergeben wurde. Ein Signalisierungskontroller kann das "Called Party Subaddress"-IE natürlich nur dann zur Übertragung der ursprünglichen Empfänger-Adresse verwenden, wenn dieses IE nicht bereits zur Adressierung der Sicherheitskomponente benötigt wird. Zur Adressierung der Sicherheitskomponente wird nur dann das "Called Party Subaddress"-IE benötigt, wenn zwischen Signalisierungskontroller und Sicherheitskomponente ein öffentliches Netz liegt, über das die Sicherheitskomponente nicht direkt adressiert werden kann. In den meisten Fällen wird zwischen Signalisierungskontroller und Sicherheitskomponente kein öffentliches Netz liegen, das eine Adressierung der Sicherheitskomponente über die "Called Party Subaddress" erfordert, da eine Manipulation von Signalisierungsnachrichten im öffentlichen Netz nicht ausgeschlossen werden kann und somit die vom Signalisierungskontroller veranlaßte Integration der Sicherheitskomponente nicht gesichert wäre. Innerhalb eines Netzes, in dem Sicherheitsrichtlinien durchgesetzt werden sollen, wird üblicherweise hingegen nur ein Adressierungsschema

verwendet werden, so daß die Verwendung des "Called Party Subaddress"-IEs zur Übertragung der Adresse des ursprünglich adressierten Empfängers kein Problem darstellt.

- Eine weitere Lösungsmöglichkeit ist es, die fehlenden Informationen über eine eigene Verbindung vom Signalisierungskontroller zur Sicherheitskomponente zu senden. Dies hätte den Vorteil, daß keine Ergänzungen der Signalisierungssoftware erforderlich wären. Da zwischen Signalisierungskontroller und Sicherheitskomponente auch der Austausch weiterer Informationen sinnvoll ist, muß diese Lösung nicht unbedingt eine zusätzliche Verbindung zu ausschließlich diesem Zweck erforderlich machen. Zwischen Signalisierungskontroller und Sicherheitskomponente können zum Beispiel Lastinformationen ausgetauscht werden oder die Sicherheitskomponente weist den Signalisierungsfilter an, eine Verbindung zu einem als Angreifer identifizierten System abzubrechen.

Die Einbindung der Sicherheitskomponente ist gegenüber dem die Verbindung initiierenden Endsystem transparent. Gegenüber dem Empfänger ("Called Party") kann die Sicherheitskomponente jedoch als Endgerät auftreten, so daß der ursprüngliche Initiator der Verbindung gegenüber dem Empfänger verdeckt wird. Alternativ kann die Sicherheitskomponente aber auch die ursprüngliche Absender-Adresse ("Calling Party ID") des die Verbindung initiierenden Endsystems verwenden und dadurch auch gegenüber dem Empfänger transparent bleiben.

3.3.1.2.5 Einfügen von Sicherheitskomponenten durch Ergänzung der DTL

PNNI legt mit einer "Designated Transit List" (DTL) den Pfad zum Empfänger fest. Eine DTL wird bereits von dem Switch erzeugt, an den das die Verbindung eröffnende Endgerät angeschlossen ist. Alle weiteren Switches folgen dem in der DTL vorgegebenen Pfad. DTL-Informationselemente sind in SETUP- und in ADD_PARTY-Signalisierungsnachrichten enthalten. Durch Ergänzung der DTL kann ein Signalisierungskontroller beim Verbindungsaufbau die Integration von Sicherheitskomponenten veranlassen.[42] DTLs sind hierarchisch angeordnet, wobei nur in der DTL auf der untersten Hierarchieebene auch ATM-Adressen der Switches angegeben werden. Bei DTLs auf höheren Hierarchieebenen sind hingegen nur logische Bezeichner für die "Peer Groups" auf dem Weg zum Empfänger enthalten. Daher ist das Einfügen von Sicherheitskomponenten nur auf der untersten Hierarchieebene sinnvoll.

[42] Das Einfügen von Sicherheitskomponenten durch Ergänzung einer DTL kann nur beim Verbindungsaufbau eingesetzt werden. Das im Abschnitt 3.3.1.2.3 diskutierte Einschleifen von Sicherheitskomponenten kann im Gegensatz dazu jederzeit erfolgen, da das Verfahren nicht von einem Verbindungsaufbau abhängt.

Beim Einfügen von Sicherheitskomponenten durch Ergänzung einer DTL sind zwei Fälle zu unterscheiden:

- Steht der Signalisierungskontroller im Pfad vor der Sicherheitskomponente, wird dies als *hereinkommende* PNNI-Signalisierungsnachricht bezeichnet. Der Signalisierungskontroller kann hierbei die PNNI-Signalisierungsnachrichten direkt manipulieren.

- Der entgegengesetzte Fall, bei dem der Signalisierungskontroller erst nach dem Passieren der Sicherheitskomponente Zugriff auf die PNNI-Nachricht erhält, wird im folgenden als *herausgehende* PNNI-Signalisierungsnachricht bezeichnet.[43] Hier kann der Signalisierungskontroller die Sicherheitskomponente nicht mehr durch direkte Manipulation der DTL integrieren. Stattdessen muß der Signalisierungskontroller einen anderen Switch dazu veranlassen, daß dieser den Weg über die erforderliche Sicherheitskomponente wählt.

Hereinkommende PNNI-Signalisierungsnachrichten werden vom Signalisierungskontroller direkt verändert, so daß die DTL die Packet Screen[44] enthält, wenn nach Sicherheitsdienstgüteanforderungen dieser Pfad gewählt werden muß. (s. Abb. 3.7)[45]

Die normale Verarbeitung der DTL durch den Switch resultiert dann in einem Pfad, der über die Sicherheitskomponente führt. Dabei ist es nicht mehr erforderlich, daß der Pfad über die Sicherheitskomponente wieder zum Signalisierungskontroller zurückführt. Wie in Abbildung 3.7 dargestellt, kann der Pfad auch über einen anderen Switch weiterführen.[46] Durch die Veränderung der DTL ist es zudem nicht mehr unbedingt notwendig, daß die Sicherheitskomponente direkt an

[43] Diese Richtungsbezeichnungen haben nichts mit herein- oder herausgehendem Verkehr in ein zu schützendes Netz zu tun. Es ist hiermit noch nicht festgelegt, welches Netz ein höheres Schutzniveau haben soll. Der Einsatz von Signalisierungskontrollern und Sicherheitskomponenten zum Aufbau von Firewall-Architekturen für ATM-Netze wird im Abschnitt 3.4 diskutiert.

[44] Die Packet Screen wird in diesem Abschnitt als Beispiel für eine Sicherheitskomponente verwendet, die über die Ergänzung einer DTL eingebunden werden kann. Ein ähnliches Vorgehen ist auch bei anderen Sicherheitskomponenten möglich.

[45] Die Richtung des Verbindungsaufbaus ist in Abbildung 3.7 durch einen Pfeil gekennzeichnet. Die durch die Signalisierung aufgebaute virtuelle Verbindung kann anschließend für unidirektionalen oder bidirektionalen Datenverkehr genutzt werden.

[46] Das Weiterführen des Pfades über einen anderen Switch hat den Vorteil, daß eine Schleife ("Loop") in der DTL vermieden wird, bei dem der Signalisierungskontroller doppelt in der DTL auftaucht. Dieser "Loop" stellt jedoch kein ernsthaftes Problem dar, weil er beim Signalisierungskontroller beginnt und endet, somit nur spezialisierte PNNI-Komponenten betrifft, die an diese Situation angepaßt sind.

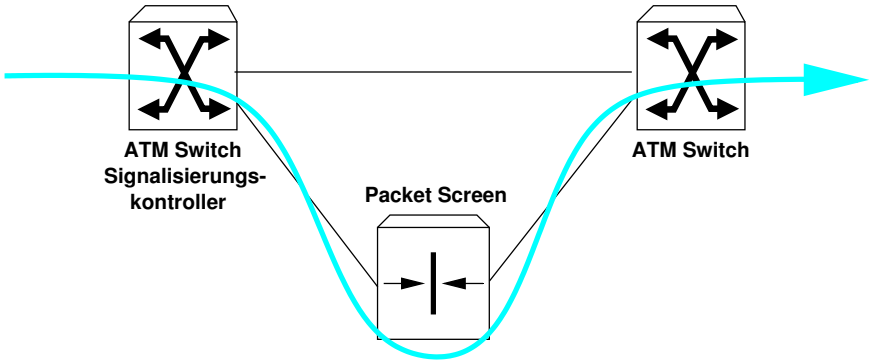


Abbildung 3.7: Sicherheitsdienstgüte-basierte Wegwahl bei hereinkommender Signalisierung

den Signalisierungskontroller angeschlossen ist. Die Sicherheitskomponente kann innerhalb der DTL an beliebiger Stelle eingefügt werden.

Beim Einfügen einer Sicherheitskomponente über das Verändern bzw. Ergänzen einer DTL können sowohl transparente als auch in Switches integrierte Sicherheitskomponenten eingefügt werden. Bei transparenten Sicherheitskomponenten, z. B. einer zwischen zwei Switches angeordneten Packet Screen (s. Abschnitt 3.3.2.2.2), muß in der DTL lediglich der "Logical Port Identifier" (s. [ATM Forum 96c, S. 207]) des vor der Sicherheitskomponente liegenden Switches abgeändert werden, so daß von mehreren parallelen Pfaden der Pfad über die Sicherheitskomponenten gewählt wird. Sicherheitskomponenten, die selbst über Switch-Funktionalität verfügen, müssen in der DTL als zusätzlicher Knoten ("Transit") eingefügt werden. Hierfür ist die Ergänzung der DTL an drei Punkten erforderlich. Der zusätzliche "Transit" besteht aus einem "Logical Node Identifier", der die ATM-Adresse der Sicherheitskomponente enthält, und einem "Logical Port Identifier", der den Port zu dem hinter der Sicherheitskomponente liegenden Switch identifiziert. Darüber hinaus ist auch die Änderung des "Logical Port Identifiers" beim vor der Sicherheitskomponente liegenden Switch erforderlich.

Bei herausgehenden Signalisierungsnachrichten, die den Signalisierungskontroller erst erreichen, nachdem die Sicherheitskomponente passiert ist (s. Abb. 3.8), kann der Pfad nicht nachträglich geändert werden. Der Signalisierungskontroller hat nur zwei Möglichkeiten. Er kann den Verbindungsaufbau gewähren oder ablehnen. Sofern nicht auch ein Signalisierungskontroller in den zweiten Switch in Abbildung 3.8 integriert wird, besteht keine Möglichkeit des direkten Umleitens von Verbindungen wie bei hereinkommenden Signalisierungsnachrichten (s. o.).

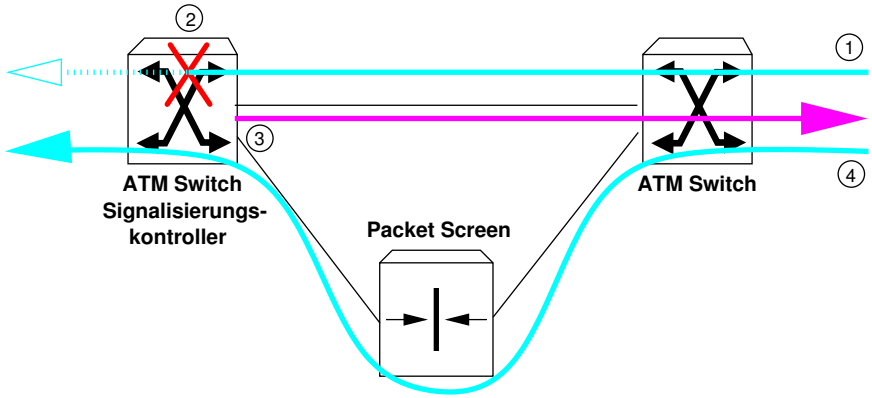


Abbildung 3.8: Sicherheitsdienstgüte-basierte Wegwahl bei herausgehender Signalisierung

Sicherheitskomponenten können bei herausgehenden Signalisierungsnachrichten über den "Crankback"-Mechanismus von PNNI integriert werden. Über eine vom Verwalter vorgegebene unterschiedliche Gewichtung der parallelen Pfade kann entweder der Pfad über die Sicherheitskomponente oder der direkte Pfad als bevorzugter Regelfall vorgegeben werden. Bei dem Szenario in Abbildung 3.8 wird angenommen, daß im Regelfall der Pfad über die direkte Leitung ohne eine Sicherheitskomponente für den Aufbau neuer virtueller Verbindungen genutzt wird (Schritt (1)). Verstößt dieser Pfad für die in der SETUP-Nachricht angegebenen Kommunikationspartner gegen die Sicherheitsdienstgüteanforderungen, muß die Verbindung abgelehnt werden (2). Der Signalisierungskontroller sendet ein "RELEASE" mit einem "Crankback Information Element" ("Crankback"-IE) zurück (3). Als Art der Blockierung wird im "Blocked Transit Type" des "Crankback"-IEs ein "call or party has been blocked at the succeeding end of this interface" signalisiert. Dadurch weiß der Empfänger dieses "Crankback"-IEs, daß die Verbindung von dem nächsten Switch (also in diesem Falle dem Signalisierungskontroller) abgelehnt wurde, weil dieser Pfad aus dem im "Crankback Cause" angegebenen Grund nicht akzeptabel ist. Als "Crankback Cause" kann im "Crankback"-IE bisher nicht der Verstoß gegen Sicherheitsdienstgüteanforderungen als Grund für das Ablehnen einer Verbindung angegeben werden. Mögliche Alternativen innerhalb der bisher spezifizierten Ablehnungsgründe [ATM Forum 96c, S. 205] sind:

"User Cell Rate not available": Hiermit teilt der Initiator dieser Nachricht mit, daß auf der Schnittstelle bzw. dem Link nicht die geforderte Zellrate zur Verfügung steht. PNNI sieht bei dieser Nachricht vor, daß optional auch aktualisierte "Topology State"-Parameter mitgesendet werden können. Diese Möglichkeit darf bei einer wegen mangelnder Sicherheitsdienstgüte abge-

lehnten Verbindung nicht genutzt werden, da die Zellrate an sich in Ordnung ist und somit von Verbindungen mit anderen Sicherheitsdienstgüteanforderungen auch grundsätzlich weiterhin genutzt werden kann.

"Quality of Service unavailable": Diese Nachricht wird für weitere über die Zellrate hinausgehende QoS-Parameter verwendet, wie "Cell Transfer Delay" (CTD), "Cell Delay Variation" (CDV) und "Cell Loss Ratio" (CLR). Wenn in der SETUP-Nachricht jedoch keine QoS-Parameter spezifiziert wurden, macht eine Ablehnung aufgrund unzureichender Betriebsmittel zur Erfüllung der QoS-Anforderungen keinen Sinn.

Neben CTD, CDV und CLR kann auch "Other QoS Parameters unavailable" als Ablehnungsgrund signalisiert werden. Diese Angabe könnte für die Signalisierung von unzureichender Sicherheitsdienstgüte genutzt werden.

"Resource unavailable, unspecified": Diese Nachricht wird bei "Call/Connection Collisions" verwendet (s. [ATM Forum 96c, Abschnitt 6.5.4, S. 223]). Kollisionen können auftreten, wenn beide Seiten gleichzeitig eine PNNI-SETUP-Nachricht senden wollen, die zusammen die verbleibenden Betriebsmittel auf dieser Schnittstelle bzw. dieser Leitung überschreiten. In diesem Fall sollen beide Seiten die Verbindung abbrechen und ein "Resource unavailable, unspecified" als Grund beim "Crankback" angeben. Dieser Ablehnungsgrund erscheint am besten für die Ablehnung einer Verbindung wegen nicht ausreichender Sicherheitsdienstgüte geeignet.

Randbedingungen bei der Ablehnung von Verbindungen: Wichtig beim Ablehnen von Verbindungen wegen unzureichender Sicherheitsdienstgüte ist, daß nicht die generelle Erreichbarkeit verweigert wird,[47] stattdessen darf nur ein temporärer Engpaß durch nicht vorhandene bzw. aufgebrauchte Betriebsmittel oder durch eine Kollision (s. o.) signalisiert werden. Da der alternative Pfad über die Sicherheitskomponente ebenfalls über denselben ATM-Switch verläuft, der diese Verbindung abgelehnt hat, darf auch nicht der Switch selbst als Ursache für die Blockierung angegeben werden. (In diesem Fall stünde in dem "Blocked Transit Type" die Information "Blocked Node".) Durch die Angabe von "call or party has been blocked at the succeeding end of this interface" wird stattdessen nur diese Schnittstelle blockiert, wodurch der alternative Pfad über die Packet Screen zum selben Switch weiterhin möglich bleibt.

Die Wahl eines alternativen Pfades nach einem "Crankback" geschieht grundsätzlich immer durch den Switch, der die DTL erstellt hat. Die einzige Ausnahme von dieser Regel betrifft den hier verwendeten "Blocked Transit Type" "call or party

[47] Dies wird in IP-Netzen zum Beispiel von Packet Screens durch ICMP-Nachrichten "Destination Unreachable" gemacht.

has been blocked at the succeeding end of this interface". Hier darf[48] der unmittelbar vor dieser Leitung liegende Switch einen alternativen Pfad wählen, der aber ebenfalls die in der DTL vorgegebenen Knoten enthalten muß [ATM Forum 96c, S. 242]. Da die DTL beim alternativen Pfad eingehalten werden muß, können nur parallele Leitungen zwischen denselben Switches ausgewählt werden; ein Pfad über andere Switches, die nicht in der DTL angegeben sind, ist nicht zulässig. Daher ist die Wahl des Pfades, der über die Sicherheitskomponente führt, nur dann bereits von diesem Switch auswählbar, wenn die Sicherheitskomponente sich bei der Signalisierung transparent verhält, wie es zum Beispiel bei der im Abschnitt 3.3.2.2.2 beschriebene "Packet Screen zwischen zwei Switches" der Fall ist. Ist die Sicherheitskomponente jedoch mit einem Switch kombiniert (s. Abschnitt 3.3.2.2.3), dann kann der alternative Pfad nur von dem Switch ausgewählt werden, der die DTL bzw. diesen Teil der DTL erstellt hat. Die "Crankback"-Nachricht muß von dem direkt an dieser Verbindung liegenden Switch unverändert weitergeleitet werden. Dabei wird auch die bereits bis hierhin aufgebaute virtuelle Verbindung abgebaut. Da der alternative Pfad bis zum Switch direkt vor der Sicherheitskomponente identisch sein wird, ist es aus Effizienzgründen wichtig, daß der Switch, der die DTL erzeugt hat, nicht zu weit entfernt ist. Der wiederholte Aufbau der virtuellen Verbindung könnte sonst sehr viel Zeit verbrauchen. Zur Vermeidung des wiederholten Aufbaus können die zum Firewall gehörenden Systeme in einer eigenen "Peer-Group" zusammengefaßt werden. Der direkt vor der blockierten Leitung liegende Switch könnte dann als Grenz-Switch ("Border Node") dieser "Peer-Group" bereits den alternativen Pfad über eine neue DTL festlegen.

Fazit: Die direkte Veränderung der DTL bei hereinkommenden Nachrichten ermöglicht eine direkte Kontrolle des Verbindungsaufbaus durch den Signalisierungskontroller. Da PNNI-Nachrichten nicht authentisiert sind, steht einer Veränderung der DTL nichts entgegen. Bei der Kontrolle herausgehender Nachrichten ist nur eine indirekte Veränderung der DTLs über den "Crankback"-Mechanismus in PNNI möglich. Der beim "Crankback" erfolgende teilweise Abbau der Verbindung verzögert den gesamten Verbindungsaufbau erheblich. Da es beim "Crankback" nicht möglich ist, die geforderte Sicherheitskomponente direkt zu spezifizieren, kann inbesondere der Einsatz von verschiedenen Sicherheitskomponenten mit unterschiedlicher Sicherheitsdienstgüte einen mehrfachen "Crankback" erfordern, bis die gewünschte Sicherheitskomponente in die DTL eingefügt wird und die Verbindung vom Signalisierungskontroller akzeptiert wird. Durch eine eigene "Peer-Group" für alle Firewall-Komponenten können die beim "Crankback" auftretenden Verzögerungen verringert werden. Sollte zukünftig eine Authentisierung von PNNI-Nachrichten einschließlich der DTL erfolgen, wird

[48] Der Switch muß nicht von seiner Möglichkeit Gebrauch machen.

ebenfalls eine eigene "Peer-Group" für alle Firewall-Komponenten benötigt. Da der erste Switch einer "Peer-Group" ohnehin eine DTL erzeugen muß, die die Signalisierungsnachricht durch die "Peer-Group" leitet, können hier auch Sicherheitsdienstgüte-Aspekte berücksichtigt werden, ohne daß eine bereits authentisierte DTL verändert werden muß. Die "Crankback"-basierte Einbindung von Sicherheitskomponenten bleibt ohnehin weiterhin möglich, da hier keine direkte Manipulation der Signalisierungsnachrichten erforderlich ist und DTLs immer nur von Switches erzeugt werden, die hierfür auch die notwendigen Authentisierungsinformationen generieren können.[49]

3.3.1.3 Integration von Sicherheitskomponenten über NHRP

Als Alternative zum direkten Einbinden von Sicherheitskomponenten über einen Signalisierungskontroller kann bei IP-basierter Kommunikation über ein ATM-Netz auch der NHRP-Dienst verwendet werden, um eine ähnliche Funktionalität zu erreichen. Die erforderlichen Veränderungen sind dabei viel geringer als für einen Signalisierungskontroller. Anstelle der direkten Manipulation von Signalisierungsnachrichten tritt hier eine Beeinflussung der Endsysteme, so daß diese selbständig Pfade wählen, die den Sicherheitsrichtlinien entsprechen. Die Einhaltung dieser Pfade wird durch einen Signalisierungsfilter überwacht.

NHRP (s. Abschnitt 2.4.3) dient zur Auflösung von IP-Adressen in ATM-Adressen. Im Gegensatz zu einem ATMARP-Server kann ein "Next Hop Server" (NHS) auch Adressen von Systemen außerhalb desselben logischen IP-Subnetzes (LIS) auflösen. Dadurch wird es einem "Next Hop Client" (NHC) ermöglicht, direkte virtuelle Verbindungen zu Systemen anderer LIS aufzubauen. Beim Auflösen der Adressen wird der Pfad verfolgt, den ein IP-Datagramm auf dem Weg zum Empfänger gehen würde. Die mit jedem Router auf dem Pfad kombinierten NHS überprüfen, ob der nächste Router ebenfalls über ATM erreichbar ist und ermitteln so die ATM-Adresse des letzten über ATM direkt erreichbaren Routers oder die ATM-Adresse des Empfängers, falls dieser direkt an das ATM-Netz angeschlossen ist. Verstößt das Durchschalten einer direkten virtuellen Verbindung gegen die Sicherheitsrichtlinien eines zwischen den Kommunikationspartnern gelegenen Netzes, kann der am Übergang zu diesem Netz angeordnete NHS eine NHRP-Anfrage auch mit der ATM-Adresse eines Systems beantworten, das eine virtuelle Verbindung des Senders annehmen kann und für das Weiterleiten von IP-Dateneinheiten an den Empfänger sorgt. Wenn über diesen Weg eine Sicherheits-

[49] Mit einer Authentisierung von PNNI-Nachrichten ist erst in ferner Zukunft zu rechnen. Da jedoch immer vom ersten Switch nach dem Übergang in eine neue "Peer-Group" ein DTL-IE ersetzt wird, muß es unabhängig von der zur Authentisierung von PNNI-Nachrichten gewählten Lösung für diese Switches zukünftig möglich sein, Authentisierungsinformationen für diese hinzugefügten DTL-IEs zu erzeugen.

komponente eingebunden wird, kann vor dem Weiterleiten der IP-Dateneinheiten eine Kontrolle anhand der Sicherheitsrichtlinien erfolgen.

Im Gegensatz zu der im vorigen Abschnitt diskutierten Modifikation von DTLs, durch die eine Sicherheitskomponente direkt in einen Pfad eingebunden wird, liefert NHRP nur die ATM-Adresse eines Systems, das das Weiterleiten von Dateneinheiten zur gewünschten IP-Adresse übernimmt. Der Verbindungsaufbau zu der angegebenen ATM-Adresse unterliegt weiterhin der Kontrolle des Endsystems. Ein Angreifer, der die ATM-Adresse des gewünschten Zielsystems bereits kennt, kann die vom NHS gelieferte ATM-Adresse der Packet Screen ignorieren bzw. braucht gar keine NHRP-Anfrage erst zu stellen und kann stattdessen eine virtuelle Verbindung direkt zur ATM-Adresse des Empfängers aufbauen. Um derartige Angriffe zu verhindern, ist bei dieser Lösung eine Überwachung durch einen Signalisierungsfilter auch weiterhin notwendig. Diese Überwachung des Aufbaus von virtuellen Verbindungen ist aber weit weniger aufwendig als Ergänzungen oder Manipulationen von DTLs durch Signalisierungskontroller.

NHRP-Anfragen enthalten die IP-Adressen des Senders und des Empfängers, sowie die ATM-Adresse des Senders oder, falls der Sender nicht direkt an das ATM-Netz angeschlossen ist, die ATM-Adresse des Routers, der an das ATM-Netz angeschlossen ist und die IP-Dateneinheiten des Senders überträgt. Für die Auswahl einer Sicherheitskomponente stehen einem NHS somit etwas andere Informationen zur Verfügung als einem Signalisierungskontroller, der nur die bei der Signalisierung angegebenen Informationen (z. B. ATM-Adressen) verwenden kann. Bei Systemen, die direkt an das ATM-Netz angeschlossen sind, ist der Unterschied zwischen der Auswertung von ATM-Adressen oder IP-Adressen unwesentlich. Da ein System mit einer ATM-Adresse gleichzeitig für mehrere Empfänger mit unterschiedlichen IP-Adressen zuständig sein kann, ist es somit im Prinzip denkbar, über die IP-Adressen eine genauere Zugriffskontrolle durchzuführen als es über die ATM-Adressen möglich ist. Da nach der NHRP-Anfrage jedoch ein regulärer Aufbau einer virtuellen Verbindung erfolgt, ist für einen Signalisierungsfilter nicht mehr möglich zu unterscheiden, für welche IP-Adressen die virtuelle Verbindung genutzt werden soll. Daher ist es erforderlich, daß alle Endsysteme, zu denen der Signalisierungsfilter einen Verbindungsaufbau gestattet, sicherstellen, daß über eine aufgebaute virtuelle Verbindung keine IP-Dateneinheiten akzeptiert werden, die gegen die Sicherheitsrichtlinien verstoßen. Wenn zum Beispiel ein an ein ATM-Netz angeschlossener Router für die beiden Systeme mit den IP-Adressen A und B zuständig ist und die Sicherheitsrichtlinien vorschreiben, daß IP-Verkehr zum System A durch einen Proxy kontrolliert werden muß, während System B unkontrolliert IP-Dateneinheiten austauschen darf, dann muß der Router IP-Dateneinheiten auf einer direkt vom externen Netz eröffneten virtuellen Verbindung ablehnen, wenn diese IP-Dateneinheiten an das System A gerichtet sind. Eine detailliertere Unterscheidung zwischen unterschiedlichen IP-Adressen ist daher nur dann möglich, wenn die Systeme, die für

IP-Adressen mit unterschiedlichen Sicherheitsanforderungen zuständig sind, mit in die Kontrolle des Firewalls einbezogen werden bzw. Teilaufgaben der Kontrolle vom Firewall übernehmen.

Da der NHS die ATM-Adresse einer beliebigen Sicherheitskomponente auswählen kann, ist hierbei eine Lastverteilung zwischen mehreren replizierten Sicherheitskomponenten möglich. Bei der NHRP-Anfrage wird vom NHS eine Sicherheitskomponente ausgewählt, deren Ressourcen für die Kontrolle der virtuellen Verbindung ausreichen. Diese Lastverteilung erfolgt vor dem Verbindungsaufbau und kann, nachdem die Verbindung aufgebaut wurde, ohne einen erneuten Verbindungsaufbau nicht mehr an geänderte Lastverhältnisse angepaßt werden. Andererseits entsteht durch diese Form der Verteilung kein zusätzlicher "Overhead" während der anschließenden Datenübertragung.

Einen ähnlichen Ansatz verfolgt [Xu et al. 98b] beim logischen Einbinden einer "Cell Screen". Der dort beschriebene Ansatz basiert auf einer Ergänzung von MPOA, beruht jedoch auf der Kontrolle durch "Edge-Devices" an den Übergängen zwischen ATM-Netz und herkömmlichen Netzen. Daher muß bei dem Ansatz von Xu der direkte Anschluß von Endgeräten an das ATM-Netz ausgeschlossen werden. Der hier vorgeschlagene kombinierte Einsatz eines Signalisierungsfilters zusammen mit einer Ergänzung von NHRP hat demgegenüber den Vorteil, auch für direkt an das ATM-Netz angeschlossene Systeme einsetzbar zu sein.

Da das hier vorgestellte Einbinden von Sicherheitskomponenten auf NHRP beruht, bietet der hier beschriebene Ansatz — im Gegensatz zu einem Signalisierungskontroller — keine Lösung für "Native"-ATM-Anwendungen. In einem ATM-Netz, das jedoch primär für die Übertragung von IP-Datenverkehr genutzt wird, ermöglicht die Ergänzung von NHRP hingegen eine einfache Lösung zum Einbinden von Sicherheitskomponenten, da die Implementation erheblich einfacher zu realisieren ist als eine Einbindung über einen Signalisierungskontroller.[50]

Fazit: Bei der Integration von Sicherheitskomponenten über NHRP beruht die Sicherheit des Firewalls auf der Kontrolle des Verbindungsaufbaus durch den Signalisierungsfilter und der Kontrolle der übertragenen Daten in den Sicherheitskomponenten wie Packet Screen oder Proxy-Server. Der NHS liefert zwar auf NHRP-Anfragen die ATM-Adresse der Sicherheitskomponente, jedoch wird die virtuelle Verbindung anschließend durch das Endsystem aufgebaut, für das die Antworten des NHS nicht verbindlich sind. Erst durch den kombinierten Einsatz von NHS mit einem Signalisierungsfilter kann dieselbe Sicherheit erreicht werden wie durch einen Signalisierungskontroller.

[50] Zusätzlich zur Adreßauflösung bei einer IP-Kommunikation wird NHRP auch von MPOA (s. Abschnitt 2.4.7) genutzt, um die ATM-Adresse eines Empfängers zu ermitteln. Welche Ergänzungen notwendig sind, um den hier vorgestellten Ansatz auch bei MPOA anzuwenden, wurde nicht untersucht.

3.3.1.4 Fazit: Kontrolle der Signalisierung

Über die Kontrolle der Signalisierung ergeben sich bei ATM-Netzen neue Möglichkeiten, die in herkömmlichen Netzen noch nicht verfügbar waren. Da jeder Kommunikationsbeziehung der Aufbau einer virtuellen Verbindung vorangehen muß, kann beim Verbindungsaufbau durch eine einmalige Zugriffskontrolle festgelegt werden, welche Kommunikationspartner Daten austauschen dürfen und welche nicht. Für die nachfolgende Datenkommunikation entstehen — im Gegensatz zu einer Packet Screen — keine Performanzeinbußen. Da der Aufbau einer virtuellen Verbindung vergleichsweise selten stattfindet, können bei der Signalisierung sehr viel aufwendigere Zugriffskontrollmechanismen eingesetzt werden als bei einer Packet Screen, die jedes Datagramm untersuchen muß.

Im Abschnitt 3.3.1.1.1 wurden erstmalig Signalisierungsnachrichten und ihre Informationselemente auf die Eignung für eine Zugriffskontrolle hin untersucht. Die wichtigsten Informationen sind — wie erwartet — in der ersten Nachricht des Verbindungsaufbaus (SETUP) enthalten. Interessanter als die bereits heute für eine Zugriffskontrolle verwendeten Sender- und Empfänger-Adressen sind die Pfadinformationen, die in DTL-IEs enthalten sind. Durch eine Überwachung dieser Pfadinformationen wird es nicht nur möglich, "Spoofing"-Angriffe zu erkennen, denn auch zur Identifizierung des Angreifers können diese Pfadinformationen dienen. Durch Kontrolle der Informationen zur Verarbeitung in den Endsystemen (BLLI und BHLI) kann bei "Native"-ATM-Anwendungen eine ähnliche Kontrolle erreicht werden, wie es heute über Packet Screens in herkömmlichen Netzen möglich ist. Für die Kontrolle von Punkt-zu-Mehrpunkt-Verbindungen ist es erforderlich, Zugriffskontrollentscheidungen auf der Basis mehrerer Signalisierungsnachrichten treffen zu können, so daß hierfür Signalisierungs*proxies* benötigt werden. Völlig neuartige Kontrollmöglichkeiten ergeben sich schließlich aus der Modifikation von Signalisierungsnachrichten. Im Abschnitt 3.3.1.2 wurde demonstriert, welche Möglichkeiten für das Einbinden zusätzlicher Sicherheitskomponenten beim Aufbau einer virtuellen Verbindung bestehen. Die Untersuchung in diesem Abschnitt zeigte, daß die Funktionalität eines Signalisierungskontrollers auch ohne Veränderung der bestehenden PNNI-Spezifikation erreicht werden kann. Gleichwohl erfordern die hier beschriebenen Funktionen für die Zugriffskontrolle auf Signalisierungsnachrichten die Integration mit der Steuerungssoftware eines Switches, was die Implementierung eines Prototyps verhindert hat. Im Abschnitt 3.3.1.3 wurde abschließend gezeigt, wie für eine Zugriffskontrolle von IP-basierter Kommunikation mit der Kombination von NHRP und einem einfachen Signalisierungsfilter eine ähnliche Funktionalität erreicht werden kann wie mit einem Signalisierungskontroller. Welche leistungsfähigen Firewall-Architekturen durch die Zugriffskontrolle auf Signalisierungsnachrichten ermöglicht werden, wird der Abschnitt 3.4 aufzeigen.

3.3.2 Packet Screens für ATM-Netze

Durch die im vorigen Abschnitt beschriebene Kontrolle von Signalisierungsnachrichten kann der Aufbau virtueller Verbindungen zwischen bestimmten ATM-Adressen verhindert werden. Zum Zeitpunkt des Verbindungsaufbaus ist jedoch die Art der auf der virtuellen Verbindung übertragenen Daten noch nicht ausreichend vorherzusehen. Eine Kontrolle der übertragenen Nutzerdaten erfordert wie auch bei herkömmlichen Firewalls einen Eingriff in den Datenstrom, der zwischen den beiden Kommunikationspartnern ausgetauscht wird. Auch in ATM-Netzen können hier je nach der geforderten Granularität der Zugriffskontrolle entweder Packet Screens oder Proxy-Server zum Einsatz kommen. In diesem Abschnitt sollen unterschiedliche Ansätze für die Integration von Packet Screens in ATM-Netze untersucht werden.

In herkömmlichen Netzen werden Packet Screens entweder in einen Router oder eine Bridge integriert. Bridges werden in ATM-Netzen nicht eingesetzt. Demgegenüber werden Router auch in ATM-Netzen eingesetzt, um bei CLIP- und LANE-Netzen verschiedene IP-Subnetze miteinander zu verknüpfen. "Klassische Packet Screens" können in diesen Routern zum Einsatz kommen, um den Datenverkehr zwischen den beiden angeschlossenen IP-Subnetzen zu kontrollieren.

3.3.2.1 Klassische Packet Screens für CLIP-Netze

Sowohl bei CLIP als auch bei LANE wird auf dem ATM-Netz ein herkömmliches Ethernet bzw. IP-Subnetz emuliert. Für die Kommunikation mit Systemen außerhalb des jeweiligen Subnetzes wird die Hilfe eines Routers benötigt. Dabei wird die Kommunikation zwischen Systemen verschiedener Subnetze auch dann über den Router geführt, wenn das ATM-Netz den direkten Aufbau einer virtuellen Verbindung ermöglichen würde. Dies hat zur Folge, daß für die Kommunikation zwischen Systemen unterschiedlicher Subnetze zwei virtuelle Verbindungen benötigt werden. Eine virtuelle Verbindung vom Kommunikationspartner im einen Subnetz zum Router und eine weitere virtuelle Verbindung vom Router zum Kommunikationspartner im anderen Netz. Der Router ist also Endpunkt beider virtueller Verbindungen, die als PVC oder SVC konfiguriert sein können. An einem Endpunkt müssen ATM-Zellen wieder zu AAL-Rahmen zusammengefügt werden. Die in den AAL-Rahmen enthaltenen Dateneinheiten werden ausgepackt und eventuell vorhandene SNAP-Header werden verarbeitet. Am Ende dieses Prozesses steht eine IP-Dateneinheit, die dann ebenso wie bei Routern in herkömmlichen Netzen verarbeitet wird. Dies ermöglicht auch die Anwendung von Filterregeln.

Der Einsatz von ATM stellt konzeptionell für die klassische Packet Screen kein Problem dar, weil für die Filterung IP-Dateneinheiten vorliegen. Die Reassemb-

lierung von Zellen zu IP-Dateneinheiten sowie die nach der Verarbeitung durch die Packet Screen notwendige erneute Segmentierung der IP-Dateneinheiten zu ATM-Zellen stellt keine wesentliche Belastung der Packet Screen dar, weil Segmentierung und Reassemblierung effizient in der Hardware der ATM-Schnittstelle implementiert werden. Die Reassemblierung ist jedoch für eine Verzögerung verantwortlich, da eine IP-Dateneinheit erst nach Erhalt der letzten Zelle rekonstruiert werden kann und somit auch erst hiernach die Verarbeitung der IP-Dateneinheit beginnen kann.

Die Performanz dieser Lösung wird durch den maximalen Datagrammdurchsatz der Packet Screen bestimmt, der wiederum von der Anzahl der pro Dateneinheit auszuwertenden Filterregeln bestimmt ist. Die Performanzeigenschaften dieser Lösung werden im Abschnitt 4.3 analysiert.

Auch wenn bei CLIP und LANE keine direkten virtuellen Verbindungen zu Systemen in anderen Subnetzen aufgebaut werden, kann diese Möglichkeit abhängig von der gewählten Firewall-Architektur trotzdem bestehen. Eine Umgehung der Packet Screen durch den Aufbau direkter virtueller Verbindungen zwischen beiden Subnetzen muß über zusätzliche Maßnahmen verhindert werden. Die Auswahl dieser Maßnahmen, etwa der Einsatz von Signalisierungsfiltern oder eine physikalische Trennung beider Subnetze auch auf ATM-Ebene, ist jedoch eine Frage der Firewall-Architektur. Unterschiedliche Firewall-Architekturen werden im Abschnitt 3.4 diskutiert.

Aufgrund der erforderlichen logischen oder physikalischen Trennung auf ATM-Ebene eignen sich klassische Packet Screens nicht besonders für den gleichzeitigen Einsatz von NHRP (s. Abschnitt 2.4.3) oder MPOA (s. Abschnitt 2.4.7). NHRP ist eine Weiterentwicklungen von CLIP, bei der virtuelle Verbindungen nicht nur zwischen Systemen desselben Subnetzes aufgebaut werden können. Zur Entlastung der dazwischenliegenden Router und um einen höheren Durchsatz zu erreichen, wird bei der Kommunikation mit Systemen anderer Subnetze versucht, virtuelle Verbindungen direkt zu Endsystemen außerhalb des Subnetzes aufzubauen. NHRP wird auch bei MPOA eingesetzt, um den Aufbau direkter virtueller Verbindungen auch bei LANE zu ermöglichen. Der Aufbau virtueller Verbindungen direkt zu Endsystemen auf der anderen Seite einer klassischen Packet Screen kann nicht zugelassen werden, da eine Kontrolle dieser Verbindung durch die Packet Screen nicht möglich wäre. Eine klassische Packet Screen muß daher trotz des Einsatzes von NHRP ein Endpunkt der virtuellen Verbindungen sein. Dies ist nur dadurch möglich, daß sich der Router, auf dem die Packet Screen eingesetzt wird, als verantwortlicher Endpunkt für die Systeme des jeweils anderen Netzes ausgibt. Die NHRP-Struktur läßt dies zu, da die "NHRP Resolution Requests" entlang desselben Pfades weitergeleitet werden, den auch die IP-Dateneinheiten von Router zu Router nehmen. Die Packet Screen ist dadurch

in der Lage, einen "NHRP Resolution Request" so zu beantworten, daß der NHC anschließend die virtuelle Verbindung zur Packet Screen aufbaut.

Die Einschränkung von NHRP, daß keine direkten virtuellen Verbindungen zwischen zwei Routern aufgebaut werden dürfen, macht sich besonders stark bemerkbar, da somit beide Endsysteme, die über die Packet Screen kommunizieren wollen, direkt an das ATM-Netz angeschlossen sein müssen. Ist eines der beiden Endsysteme an ein herkömmliches Netz angeschlossen, muß der Router am Übergang zwischen ATM-Netz und herkömmlichen Netz als Endpunkt einer virtuellen Verbindung dienen. Da jedoch auch die Packet Screen ein Router ist, wäre eine direkte virtuelle Verbindung zwischen zwei Routern erforderlich, was bei NHRP zur Vermeidung von "Routing-Loops" verboten ist. Dasselbe Problem tritt auf, wenn beide kommunizierenden Endsysteme jeweils über eine klassische Packet Screen abgesichert sind. Bis die Problematik des Einsatzes von NHRP zwischen zwei Routern gelöst ist, muß wie in einem CLIP-Netz auf ein normales Weiterleiten von IP-Datagrammen von einem Router zum nächsten Router zurückgegriffen werden.

Im Gegensatz zu NHRP und MPOA ist die Integration einer Packet Screen bei "IP-Switching" (s. Abschnitt 2.4.5) bereits im Konzept vorgesehen. Bei "IP-Switching" beginnt eine Kommunikation zunächst mit dem herkömmlichen Weiterleiten von IP-Dateneinheiten von einem Router zum nächsten Router. Soll für einen bestimmten Datenfluß ("Flow") eine virtuelle Verbindung eingerichtet werden, sendet der IP-Switch die den "Flow" identifizierenden Informationen (IP-Adressen und Ports) in einer "Redirection"-Nachricht an seinen benachbarten IP-Switch. Eine Packet Screen wird zwei virtuelle Verbindungen nur dann in der Switch-Komponente des IP-Switches miteinander verbinden, wenn die den jeweiligen "Flow" identifizierenden Sender- und Empfänger-IP-Adressen und Ports von den Filterregeln zugelassen werden. Die den "Flow" identifizierenden Merkmale sind in dem auf dieser virtuellen Verbindung verwendeten Übertragungsformat nicht enthalten und werden erst durch den letzten IP-Switch auf der virtuellen Verbindung wieder hinzufügt. Daher können auf dieser virtuellen Verbindung keine Dateneinheiten mit anderen IP-Adressen oder Ports übertragen werden. Die aufgebaute virtuelle Verbindung kann somit nicht zur Umgehen der Packet Screen benutzt werden. Ein IP-Switch, der abweichende Adreßinformationen in Dateneinheiten einfügt, könnte zwar zur Umgehung der Packet Screen verwendet werden, stellt als "Insider-Angriff" aber keine neue Qualität im Vergleich zu "Insider-Angriffen" gegen Packet Screens in herkömmlichen Netzen dar. Ein weiterer Vorteil beim Einsatz einer klassischen Packet Screen mit dem "IP-Switching" ist die Reduzierung des Filterungsaufwands, da die über die für einen "Flow" eingerichtete virtuelle Verbindung übertragenen Dateneinheiten nicht mehr von der Packet Screen untersucht werden müssen. Für die Integration einer klassischen Packet Screen bietet das "IP-Switching" daher gute Voraussetzungen.

3.3.2.2 Packet Screens auf ATM-Schicht

Bei der bisher beschriebenen klassischen Packet Screen war die Filterfunktion mit einem Router gekoppelt. Die Erfordernis, die technisch aufwendige Router-Komponente auf Schicht 3 des OSI-Modells einsetzen zu müssen, führt bei der klassischen Packet Screen zu Performanzproblemen und verhindert zudem eine transparente Integration in ein ATM-Netz. Die Integration einer Packet Screen in ein ATM-Netz, ohne daß die Packet Screen als Endpunkt der virtuellen Verbindungen sichtbar wird, hätte ähnliche Vorteile wie sie in herkömmlichen Netzen bei der Integration einer Packet Screen in eine Bridge erreicht wird.[51]

Während bei der vorhergehend beschriebenen Integration einer Packet Screen in einem Router die Dateneinheiten über die üblichen "Routing"-Verfahren (Default-Route oder explizite Route) an die Packet Screen geschickt werden, basieren bisherige in Bridges integrierte (Schicht 2) Packet Screens auf der "Broadcast"-Funktionalität von Ethernet oder FDDI. Ohne diese "Broadcast"-Funktionalität kann dieser Ansatz auf das verbindungsorientierte ATM-Konzept daher nicht direkt übertragen werden. Bei ATM sind drei unterschiedliche Lösungsmöglichkeiten denkbar:

1. Einsatz eines Filters auf der Leitung zwischen zwei Switches.
   (s. Abschnitt 3.3.2.2.2)

2. Die Filterfunktionalität wird in den Switch integriert.
   (s. Abschnitt 3.3.2.2.3)

3. Packet Screen wird logisch in den Pfad integriert.
   (s. Abschnitt 3.3.2.2.4)

Allen drei Integrationsmöglichkeiten ist gemeinsam, daß die Dateneinheiten nicht zwangsläufig vor der Filterung reassembliert werden müssen und die Packet Screen die Regeln stattdessen auch auf einzelne ATM-Zellen anwenden kann. Diese Filterung einzelner Zellen im Gegensatz zur Filterung von AAL-5-Rahmen oder IP-Datagrammen gab diesem Konzept den Namen "Cell Screen". Bevor die drei Möglichkeiten für die Integration einer Packet Screen in ein ATM-Netz analysiert werden, soll zunächst das Konzept der "Cell Screen" vorgestellt werden.

[51] In einem Ethernet werden bei einer Bridge-basierten Packet Screen Ethernet-Rahmen gefiltert. Trotz der Integration des Filters auf Schicht 2 des OSI-Modells werden von der Packet Screen auch die Header auf Schicht 3 und 4 untersucht. Die auf einem MS-DOS PC aufsetzende "Drawbridge" ist ein Beispiel für eine derartige Lösung.

3.3.2.2.1 "Cell Screen"

Eine "Cell Screen" filtert einen Strom von ATM-Zellen. Dabei werden die in der ersten Zelle eines IP-Datagramms enthaltenen Informationen (IP-Adressen und Ports) anhand von Packet Screen Regeln ausgewertet. Die getroffene Entscheidung wird dann auch für alle folgenden Zellen, die zu demselben IP-Datagramm gehören, angewendet. Diese Filterung ohne vorherige Reassemblierung ist effizienter als die Filterung in einer klassischen Packet Screen, was insbesondere für Verkehr mit QoS-Anforderungen hinsichtlich der Verzögerung wichtig ist. Zudem kann die Integration einer "Cell Screen" transparent erfolgen.

Für die detaillierte Beschreibung der Filterung von Datagrammen ohne vorheriges Reassemblieren der ATM-Zellen muß zunächst das Übertragungsformat der zu filternden Dateneinheiten im ATM-Netz vorgestellt werden. Als Übertragungsformat wird hier zunächst CLIP (s. Abschnitt 2.4.2) angenommen. Dasselbe Format wird auch bei NHRP (s. Abschnitt 2.4.3) verwendet. Abschließend wird kurz auf weitere Formate eingegangen.

Bei der Kommunikation in einem CLIP-Netz wird der in Abbildung 2.16 (S. 56) dargestellte Protokollstapel verwendet. Vor eine IP-Dateneinheit wird zunächst ein "SNAP-Header" [Heinanen 93] angefügt. Der SNAP-Header enthält Multiplexinformationen, die das enthaltene Paket, in diesem Falle also ein IP-Datagramm, identifizieren. Dadurch ermöglicht der "SNAP-Header" die Übertragung von Dateneinheiten verschiedener Protokolle über dieselbe virtuelle Verbindung. Die "ATM Adaption Layer" ist verantwortlich für das Segmentieren von Datagrammen und das Reassemblieren von Zellen. Für Datenkommunikation wird üblicherweise AAL-5 verwendet. Ein AAL-5-Rahmen besteht aus einem "Trailer" von 8 Byte Länge, der zusammen mit eventuell notwendigem "Padding" an den SNAP-Rahmen angehängt wird. Die ATM-Zelle, die den "AAL-5-Trailer" enthält, wird von der "ATM Adaption Layer" durch das Setzen eines Bits des "Payload Type"-Feldes im ATM-Zellkopf markiert,[52] um dem Empfänger ein Reassemblieren zu ermöglichen. Da die AAL-5-Rahmen aufeinanderfolgend über eine virtuelle Verbindung übertragen werden, ergibt sich durch die Markierung der letzten Zelle implizit auch eine Markierung der ersten Zelle des nachfolgenden AAL-5-Rahmens.[53] Alle folgenden, auf derselben virtuellen Verbindung übertra-

[52] Die letzte Zelle wird durch einen Wert von "5" im "Payload Type"-Feld markiert, während dieses Feld bei allen anderen Zellen des AAL-5-Rahmens den Wert von "4" enthält. [ATM Forum 94, S. 59]

[53] Werden auf einer virtuellen Verbindung (IP-)Dateneinheiten in AAL-5-Rahmen übertragen, muß nach der letzten Zelle einer Dateneinheit immer die erste Zelle der folgenden Dateneinheit kommen. Die hierfür erforderliche Einhaltung der Reihenfolge bei der Übertragung von Zellen wird vom ATM-Netz gewährleistet. In IP-Netzen ist die Einhaltung der Reihenfolge bei der Übertragung demgegenüber nicht gewährleistet, so daß z. B. bei TCP Mechanismen erforderlich sind, um die Reihenfolge der IP-Dateneinheiten wieder herzustellen.

genen Zellen (bis zur nächsten markierten ATM-Zelle) gehören dann zu demselben AAL-5-Rahmen. Da dem Empfänger bei der Reassemblierung zur Identifikation der zu einem AAL-5-Rahmen gehörenden Zellen nur die markierte letzte Zelle eines AAL-5-Rahmens zur Verfügung steht, können AAL-5-Rahmen nur hintereinander auf derselben virtuellen Verbindung übertragen werden. Der verwendete AAL-Typ und auch der Einsatz eines "SNAP-Headers" wird für jede virtuelle Verbindung bei der Signalisierung im Falle eines SVCs oder alternativ bei der Konfiguration eines PVCs festgelegt. Ob ein "SNAP-Header" eingesetzt wird und welcher AAL-Typ verwendet wird, kann für jede virtuelle Verbindung unabhängig von anderen Verbindungen einzeln festgelegt werden.

Eine "Cell Screen" untersucht diesen Zellstrom getrennt für jede virtuelle Verbindung. Unterschiedliche virtuelle Verbindungen sind eindeutig identifizierbar am VPI/VCI-Paar. Auf jeder virtuellen Verbindung wird nach der Markierung der letzten Zelle eines AAL-5-Rahmens gesucht. Trifft eine markierte Zelle ein, so muß es sich bei der folgenden Zelle um den Anfang eines neuen AAL-5-Rahmens handeln.

Bei der Verwendung von CLIP beginnt ein AAL-5-Rahmen mit einem 8 Byte langen SNAP-Header. Gleich danach folgt der IP-Header. Nach dem SNAP-Header stehen in der ersten Zelle somit noch 40 Byte für IP-Header (20 Byte) und TCP-Header (20 Byte) zur Verfügung (s. Abb. 3.9). Der alternativ zu TCP einsetzbare UDP-Header ist mit nur 8 Byte kürzer als der TCP-Header. Beim Versenden von IPv4 Dateneinheiten in AAL-5-Rahmen sind somit alle für eine Packet Screen relevanten Informationen in der jeweils ersten Zelle eines AAL-5-Rahmens enthalten. Die folgenden Zellen brauchen daher von der Filterfunktion nicht untersucht werden und können je nach der für die erste Zelle getroffenen Entscheidung direkt weitergeleitet oder vernichtet werden.

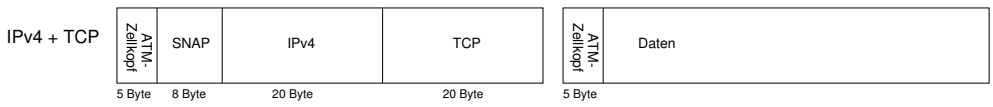


Abbildung 3.9: Aufteilung von IP-Dateneinheiten auf ATM-Zellen bei CLIP und NHRP

Für eine Packet Screen auf Zell-Ebene ("Cell Screen") reicht es in einem CLIP-Netz daher aus, jeweils nur die erste Zelle einer Dateneinheit zu untersuchen und anschließend die getroffene Entscheidung auch auf alle folgenden Zellen derselben Dateneinheit anzuwenden. Dadurch ergibt sich ein sehr einfacher und leistungsstarker Aufbau einer Packet Screen. Da Dateneinheiten nicht vor der Filterung reassembliert werden müssen, sind die durch die Packet Screen bedingten Verzögerungen viel geringer als bei einer klassischen Packet Screen, die zunächst abwarten

muß, bis auch die letzte Zelle eines AAL-5-Rahmens eingetroffen ist.[54] Da zudem nur die erste Zelle einer Dateneinheit untersucht werden muß und folgende Zellen direkt von der "Switch-Fabric" weitergeleitet werden können, sind weniger Daten zu kopieren. Es ist davon auszugehen, daß die "Cell Screen"-Funktionalität zukünftig in ATM-Switches integriert wird. Einige ATM-Switches enthalten bereits eine Funktion, die alle Zellen einer AAL-5-Dateneinheit bis zum Erreichen der letzten Zelle vernichtet. Bisher dient diese "Frame Discard" genannte Funktion dazu, daß der Switch, der eine Zelle vernichten muß,[55] dadurch die sinnlose Übertragung der restlichen Zellen einer vom Empfänger ohnehin nicht mehr rekonstruierbaren Dateneinheit vermeidet. Erst bei Beginn einer neuen Dateneinheit wird wieder mit der Übertragung von Zellen begonnen. Dieselbe Funktion kann in einer "Cell Screen" durch die Filterfunktion aktiviert werden, nachdem die erste Zelle aufgrund von Filterregeln vernichtet wurde. Alternativ können alle Zellen des blockierten Datagramms auch auf eine andere Schnittstelle umgeleitet werden. Durch den Anschluß eines geeigneten Aufzeichnungssystems an diese Schnittstelle ließe sich ein leistungsfähiger Netzwerkmonitor implementieren.

Bisher sind zwei Implementationen des "Cell Screen"-Konzeptes bekannt. Die Firma "StorageTek" bietet seit 1996 ein "ATM Line Access & Security" (ATLAS) System an, das laut Herstellerinformationen auf einer OC-3c ATM-Verbindung (155 MBit/s) als Packet Screen einsetzbar ist, ohne einen Engpaß darzustellen. Neben dem bereits beschriebenen "Cell Screen"-Konzept wird bei ATLAS noch ein sogenannter "Policy Cache" eingesetzt, der bereits bei vorigen IP-Dateneinheiten getroffene Filterentscheidungen schnell wieder zugreifbar macht, ohne erneut eine Auswertung der Filterregeln zu erfordern [Hughes 96, Kowalski 96]. Implementiert wird der "Policy Cache" über "Content Addressable Memory" (CAM), der sehr schnellen Zugriff auf bereits getroffene Filterentscheidungen ermöglicht. Die Kombination aus "Cell Screen" und "Policy Cache" soll auch für höhere Geschwindigkeiten einsetzbar sein, jedoch ist die bereits seit über einem Jahr angekündigte OC-12c (622 MBit/s) Version von ATLAS noch nicht verfügbar.

Der zweite von [Xu et al. 98a] vorgestellte "Cell Screen"-Ansatz basiert auf einem ATM-Switch und ergänzt die Funktionen von ATLAS noch um die sogenannte "Last Cell Hostage"-(LCH)-Technik. Dabei wird ausgenutzt, daß der Empfänger

[54] Die maximale Größe einer AAL-5-Dateneinheit beträgt 64 KByte. Für die Übertragung dieser Dateneinheit auf einer virtuellen Verbindung, für die ein Datendurchsatz von 64 KBit/s reserviert ist, werden etwa 8 Sekunden benötigt. Diese 8 Sekunden betreffen nur die Verzögerung bevor eine Reassemblierung stattfinden kann. Erst danach kann die Filterung durch die Packet Screen beginnen.

[55] Ein Zellverlust kann beispielsweise auftreten, wenn im Switch kein ausreichender Platz mehr in den Puffern ist oder wenn auf verschiedenen Schnittstellen hereinkommende Zellen auf derselben Schnittstelle herausgesendet werden sollen und dadurch die Zellrate dieser Schnittstelle überschritten wird.

einen AAL-5-Rahmen nur dann rekonstruieren kann, wenn er alle Zellen des Rahmens erhalten hat. Daher können mit der LCH-Technik ATM-Zellen sofort weitergeleitet werden, ohne daß zuerst eine Filterentscheidung abgewartet werden muß. Erst vor dem Weiterleiten der markierten letzten Zelle des AAL-5-Rahmens wird überprüft, ob die Auswertung der Filterregeln ein positives Ergebnis geliefert hat. Wenn die Auswertung ergeben hat, daß der AAL-5-Rahmen blockiert werden muß, wird die letzte Zelle nicht weitergeleitet. Stattdessen wird eine ebenfalls als "letzte Zelle" markierte Zelle mit beliebigem Inhalt generiert und an den Empfänger gesendet. Die generierte Zelle garantiert, daß die vom Empfänger durchgeführte Überprüfung der Integrität den rekonstruierten AAL-5-Rahmen als beschädigt erkennt und der AAL-5-Rahmen daher vernichtet wird.[56]

Vorteil dieses Ansatzes sind die geringeren Probleme bei der Gewährleistung der Einhaltung der Reihenfolge von Zellen, da für das Weiterleiten der ersten Zelle nicht auf die Filterentscheidung gewartet werden muß. Ohne die LCH-Technik ist es zur Erhaltung der Reihenfolge notwendig, nachfolgende Zellen einer virtuellen Verbindung zu puffern bis die Filterentscheidung für die erste Zelle erfolgt ist. Die durch die Filterung bedingten Verzögerungen können zu Problemen bei der Einhaltung von etwaigen QoS-Anforderungen führen. Bei der LCH-Technik werden diese Verzögerungen vermieden, da die Zellen sofort weitergeleitet werden.

Die Vorteile der LCH-Technik kommen kaum zum Tragen, wenn sehr kurze AAL-5-Rahmen überprüft werden müssen. Da in zwei Zellen selbst beim Einsatz von TCP bereits bis zu 40 Byte Benutzerdaten übertragen werden, kommt es häufig vor, daß AAL-5-Rahmen lediglich aus zwei Zellen bestehen. Die zwischen dem Eintreffen der ersten und der zweiten Zelle für die Auswertung der Filterregeln zur Verfügung stehende Zeit ist dann gering. Ein Nachteil der LCH-Technik ist, daß alle Zellen mit Ausnahme der letzten Zelle bis zum Endsystem gelangen. Dadurch verbrauchen auch die von der "Cell Screen" blockierten Dateneinheiten noch einen Anteil des im Netz zur Verfügung stehenden Datendurchsatzes. Zudem ist dieses Verfahren davon abhängig, daß der Empfänger den AAL-5-Rahmen tatsächlich vernichtet. Vom Sender können Dateneinheiten generiert werden, die in der letzten Zelle nur den AAL-5-Trailer und Fülldaten ("Padding") enthalten, während die zu übertragenden Daten bereits in den vorher übertragenen Zellen enthalten sind. Ein kooperierender Insider könnte dann auch ohne eine korrekte AAL-5-Prüfsumme versuchen, die gesendete Dateneinheit zu rekonstruieren.

[56] Würde statt der gesendeten generierten Zelle die letzte Zelle von der "Cell Screen" lediglich vernichtet werden, dann würde der Empfänger versuchen, die bereits empfangenen Zellen mit den Zellen des folgenden AAL-5-Rahmens zu einem AAL-5-Rahmen zusammenzubauen. Die Prüfsumme würde auch bei dieser Dateneinheit zu der gewünschten Vernichtung im Endsystem führen. Da das Endsystem jedoch das Ende der blockierten Dateneinheit nicht mehr erkennen kann, wäre von der Vernichtung auch die folgende, möglicherweise erlaubte Dateneinheit betroffen.

Eine "Cell Screen" ist deutlich schwieriger zu realisieren, wenn nicht mehr die gesamten von der Packet Screen benötigten Informationen in der ersten Zelle vorhanden sind. Dies kann etwa bei "LAN Emulation" (s. Abb. 3.10), durch den Einsatz von IP-Optionen oder durch einen IPv6-Header [Deering et al. 95, Hinden et al. 95] bedingt sein. Der längere Header bei LANE führt dazu, daß 8 Byte des TCP-Headers in der zweiten Zelle übertragen werden müssen. Zwar bleiben die TCP-Portnummern in der ersten Zelle, aber das ebenfalls von Packet Screens ausgewertete SYN-Flag wird bei LANE erst in der zweiten Zelle übertragen. IP-Optionen sind hingegen ein geringeres Problem, da Packet Screens auch bisher schon die Übertragung von Dateneinheiten mit gesetzten IP-Optionen üblicherweise nicht gestattet haben. IPv6 wird jedoch zukünftig eine große Rolle spielen [Huitema 96], so daß hierfür eine Lösung gefunden werden muß. Bei IPv6 hat sich die Header-Länge gegenüber IPv4 verdoppelt, so daß bereits der IPv6-Header zusammen mit dem SNAP-Header die erste Zelle ausfüllt. Die für eine Packet Screen wichtigen TCP-/UDP-Portnummern sind dann erst in der zweiten Zelle enthalten (s. Abb. 3.11). Der Aufwand für die Packet Screen steigt, da die Entscheidung nicht mehr allein anhand der ersten Zelle getroffen werden kann. Um trotzdem eine Filterung zu ermöglichen, muß die erste Zelle zwischengespeichert werden bis die zweite Zelle ankommt. Anschließend müssen die beiden Zellen zusammen ausgewertet werden. Wird nicht gleichzeitig auch die "Last Cell Hostage"-Technik (s. o.) eingesetzt, führt diese Zwischenspeicherung zu Verzögerungen.

Diese Zwischenspeicherung verzögert die Übertragung mindestens so lange bis die zweite Zelle angekommen ist. Im Gegensatz dazu bedingt die Reassemblierung in einer klassischen Packet Screen (s. Abschnitt 3.3.2.1) eine Verzögerung, die abhängig von der Länge des AAL-5-Rahmens ist. Die durch eine Reassemblierung bedingte Verzögerung ist nicht nur länger, sondern wegen der variierenden Rahmenlänge weniger konstant als die Verzögerung durch das Abwarten der zweiten Zelle.

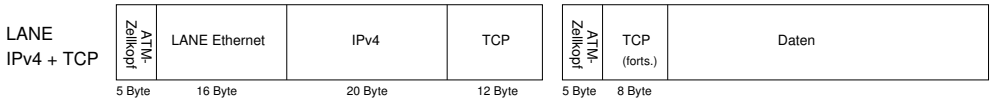


Abbildung 3.10: Aufteilung von IP-Dateneinheiten auf ATM-Zellen bei LANE

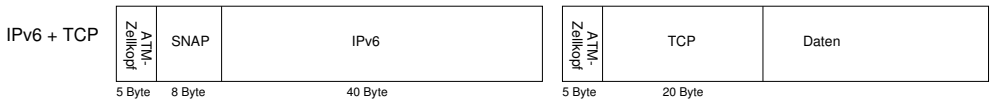


Abbildung 3.11: Aufteilung von IPv6-Dateneinheiten auf ATM-Zellen

Alternativ könnte durch das Weglassen des SNAP-Headers 8 Byte des TCP- oder UDP-Headers in der ersten Zelle übertragen werden. Diese 8 Byte würden für den gesamten UDP-Header ausreichen. Von einem TCP-Header könnte jedoch nur ein Teil übertragen werden. In den ersten 8 Byte eines TCP-Headers sind aber bereits die Sender- und Empfänger-Ports enthalten, die von einer Packet Screen ausgewertet werden müssen. Das ebenfalls von Packet Screens in einem TCP-Header ausgewertete SYN-Flag befindet sich jedoch erst im vierzehnten Byte des Headers [Comer 91, S. 183] und ist somit in jedem Fall erst in der zweiten Zelle enthalten. Durch das Weglassen des SNAP-Headers [Heinanen 93] gehen Multiplex-Informationen verloren, die die Übertragung unterschiedlicher Protokolle auf derselben virtuellen Verbindung ermöglichen. Ohne den SNAP-Header muß für jede virtuelle Verbindung festgelegt werden, welches Protokoll übertragen wird. Für die Übertragung eines weiteren Protokolls muß dann auch zwischen denselben Endgeräten eine weitere Verbindung eröffnet werden. Dieses Vorgehen wird als "VC-basiertes Multiplexing"[57] oder "NULL Encapsulation" bezeichnet. Für die Nutzung von IPv6 über ATM wird der Verzicht auf den SNAP-Header zwar optional zugelassen [Armitage et al. 98], jedoch wird von diesem Vorgehen ausdrücklich abgeraten, weil in diesem Falle für MARS (s. Abschnitt 2.4.4), NHRP (s. Abschnitt 2.4.3) und "Neighbor Discovery" [Narten et al. 96] zusätzliche parallele virtuelle Verbindungen benötigt werden.[58] Diese inflationäre Nutzung von virtuellen Verbindungen sollte vermieden werden, da bei manchen Tarifierungen für jede virtuelle Verbindung Kosten anfallen, der Aufbau virtueller Verbindungen zeitaufwendig ist und zudem die maximale Anzahl virtueller Verbindungen, die von einem Endsystem bedient werden kann, begrenzt ist.

Fazit: Die Filterung von IP-Dateneinheiten in einer "Cell Screen" ist viel effizienter als die Verarbeitung in einer klassischen Packet Screen (s. Abschnitt 3.3.2.1), da AAL-5-Rahmen zur Filterung nicht reassembliert werden müssen und der Hauptanteil der übertragenen Daten mit geringem Aufwand von der Switching-Hardware weitergeleitet werden kann. Da neben dem Weiterleiten der Zellen auch der pro Datagramm anfallende Aufwand für die Auswertung der Filterregeln beträchtlich ist, muß eine "Cell Screen" zusätzlich noch über ein effizientes Verfahren zur Auswertung der Filterregeln wie dem "Policy Cache" bei ATLAS verfügen. Die in [Xu et al. 98a] vorgeschlagene "Last Cell Hostage"-Technik verschafft der "Cell Screen" mehr Zeit für die Auswertung der Filterregeln. Jedoch kommen die blockierten Datagramme fast vollständig beim Empfänger an, so daß die "Cell Screen" bei der "Last Cell Hostage"-Technik davon abhängig ist, daß der Empfänger die Daten vernichtet. Anstelle der derzeitigen Implementation

[57] "VC" steht hier als Abkürzung für eine virtuelle Verbindung ("Virtual Connection").

[58] Wenn der SNAP-Header weggelassen wird, können ausschließlich IPv6 Dateneinheiten auf der virtuellen Verbindung übertragen werden. Eine Mischnutzung etwa mit IPv4, MARS usw. ist dann nicht möglich.

in einer eigenen Netzkomponente (z. B. ATLAS) ist, wegen der teilweise bereits in ATM-Switches vorhandenen Funktionen, zukünftig eine Integration in ATM-Switches zu erwarten, so daß die "Cell Screen"-Funktionalität kostengünstig in ATM-Netzen zur Verfügung stehen wird.

3.3.2.2.2 Filter zwischen zwei Switches

Das Einfügen einer Packet Screen in die Leitung zwischen zwei Switches (s. Abb. 3.12) entspricht vom Ansatz her dem Einfügen einer Bridge-basierten Packet Screen in herkömmliche Netze. Für das bereits erwähnte "ATLAS"-System von "StorageTek" [Hughes 96, Kowalski 96] ist diese Art des Einsatzes vorgesehen. Im Gegensatz zum Einsatz in herkömmlichen "Broadcast"-basierten Netzen erfordert eine Packet Screen in ATM-Netzen keine Unterscheidung zwischen lokalem und Packet Screen überschreitendem Verkehr, da alle bei der Packet Screen ankommenden Daten in jedem Fall für das jeweils andere Netz bestimmt sind.[59]

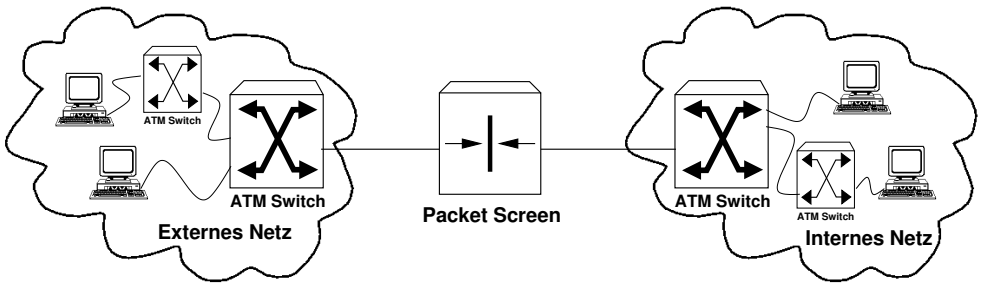


Abbildung 3.12: Packet Screen zwischen zwei Switches

Die Filterung der Dateneinheiten in der Packet Screen kann als "Cell Screen", also ohne Reassemblierung von Zellen zu Datagrammen (s. o.), implementiert werden. Damit die Packet Screen transparent zwischen zwei Switches eingesetzt werden kann, muß bei herausgehenden Zellen dasselbe VPI/VCI-Paar verwendet werden, mit dem die Zelle von der Packet Screen auch empfangen wurde.

Statt der Filterung als "Cell Screen" ist es alternativ auch möglich, die Dateneinheiten vor der Filterung zu reassemblieren und anschließend wieder in Zellen zu segmentieren. Dies würde zum Beispiel die Implementation der Packet Screen auf einer mit zwei ATM-Schnittstellen ausgestatteten Workstation ermöglichen. Die Performanz dieser Workstation-basierten Packet Screen würde der Performanz

[59] Da die Unterscheidung zwischen lokalem und Packet Screen überschreitendem Verkehr in einem ATM-Netz nicht getroffen werden muß, kann eine Packet Screen in einem ATM-Netz effizienter sein als in "Broadcast"-Netzen.

einer klassischen Packet Screen entsprechen. Da sich durch die Reassemblierung und Segmentierung in der Packet Screen jedoch kaum Vorteile[60] gegenüber einer Implementation als "Cell Screen" ergeben, ist eine Implementation als "Cell Screen" vorzuziehen.

Da die Packet Screen transparent zwischen den beiden Switches angeordnet sein soll, kann sie an der Signalisierung nicht aktiv teilnehmen. Die auf den für die Signalisierung verwendeten virtuellen Verbindungen übertragenen Nachrichten müssen daher von der Packet Screen transparent weitergeleitet werden. Die (passive) Auswertung der übertragenen Signalisierungsnachrichten ist jedoch notwendig, da bei der Signalisierung das anschließend auf der virtuellen Verbindung eingesetzte Paket-Format bestimmt wird. Ohne diese Auswertung der Signalisierungsnachrichten muß für jede zu filternde virtuelle Verbindung manuell konfiguriert werden, welcher AAL-Typ verwendet wird und welches Protokoll über der "ATM Adaption Layer" (AAL) eingesetzt wird (z. B. SNAP, CLIP, NHRP oder andere). Für PVCs ist in jedem Fall eine manuelle Konfiguration notwendig. Bei ATM-Netzen mit einer überwiegenden Nutzung von SVCs ist eine manuelle Konfiguration nicht akzeptabel, so daß eine Packet Screen hier die Signalisierungsnachrichten auswerten muß. Der Kontroller, der diese Auswertung der Signalisierungsnachrichten durchführt, macht die Implementation einer Packet Screen für den Einsatz zwischen zwei Switches erheblich komplexer. In diesem Kontroller müssen große Teile der in einem ATM-Switch bereits vorhandenen Signalisierungsfunktionen enthalten sein. Da eine Auswertung von Signalisierungsnachrichten in einem Switch ohnehin vorgenommen werden muß, ist die Integration der Packet Screen Funktionalität in einem Switch vorzuziehen.

3.3.2.2.3 Packet Screen Integration im Switch

Bereits im vorigen Abschnitt wurde dargestellt, daß der Switch durch die Teilnahme an der Signalisierung bereits über wichtige Informationen verfügt, um beim Übergang eines Paketes zwischen logischen Netzen eine Filterung durchzuführen. Bei der Integration in einem Switch ist nur die Implementation als "Cell Screen" also eine Filterung ohne Reassemblierung der Zellen zu Datagrammen sinnvoll. Die Integration einer Packet Screen in einem Switch wird in [Xu et al. 98a] beschrieben.

Der Vorteil einer Packet Screen Integration im Switch ist die dadurch mögliche enge Kopplung mit einem Signalisierungskontroller. Dieser hat einerseits die von der Packet Screen benötigten Informationen über die auf einer virtuellen Verbindung

[60] Eine Implementation auf der Basis einer Workstation wäre lediglich für die Entwicklung von Prototypen interessant.

eingesetzten Protokolle. Andererseits kann der Signalisierungskontroller auch entscheiden, bei welchen virtuellen Verbindungen der Einsatz einer Packet Screen notwendig ist. Bei einer engen Integration kann der Signalisierungskontroller sogar die Filterregeln auswählen, die von der Packet Screen auf einer bestimmten virtuellen Verbindung durchgesetzt werden sollen. Dies hat den Vorteil, daß von der Packet Screen für jedes Paket nur eine sehr geringe Anzahl von Filterregeln ausgewertet werden muß, bevor eine Filterentscheidung getroffen werden kann. Zudem verhindert dies das Auftreten von Nebenwirkungen durch Filterregeln, die für gänzlich andere Kommunikationsbeziehungen konfiguriert wurden. Wird zum Beispiel eine virtuelle Verbindung zwischen zwei Endgeräten aufgebaut, so kann der Signalisierungskontroller Filterregeln generieren, die bereits die IP-Adressen der beiden Kommunikationspartner enthalten und exakt die Ports spezifizieren, zwischen denen eine Kommunikation durchgeführt werden darf. Das Weiterleiten von Dateneinheiten mit anderen IP-Adressen wird dadurch von vornherein verhindert.

Ein weiterer Vorteil der Integration einer Packet Screen in einen Switch[61] ist, daß bereits wesentliche der für eine Packet Screen in einem ATM-Netz notwendigen Komponenten im Switch vorhanden sind. Zu nennen sind hier die Signalisierungskomponente, die Funktionen zum Weiterleiten von Zellen ("Switch Fabric") und das gezielte Vernichten von Zellen ("Packet Discard"). Leider sind die zur Ergänzung der Packet Screen Funktionalität notwendigen Erweiterungen in kommerziellen Switches nicht ohne aufwendige Veränderungen der Hard- und Software möglich. Daher konnten keine Prototypen entwickelt werden.

3.3.2.2.4 Logische Integration einer Packet Screen

Die logische Integration von Sicherheitskomponenten durch einen Signalisierungskontroller wurde bereits im Abschnitt 3.3.1.2.2 vorgestellt. Ein logische Integration funktioniert besonders gut mit transparenten Komponenten, wie zum Beispiel einer auf der Leitung zwischen zwei Switches integrierten Packet Screen oder einer Packet Screen, die in einen Switch integriert ist. Die alternativ zu einem Signalisierungskontroller in Abschnitt 3.3.1.3 vorgestellte logische Integration von Sicherheitskomponenten über eine Kombination von NHRP mit Signalisierungsfiltern eignet sich ebenfalls zur Integration einer Packet Screen.

[61] Die Dokumentation des ATM-Switches "Cisco Lightstream 1010" beschreibt bereits die Konfiguration von Filterregeln über die von Routern desselben Herstellers bekannten "Access Control Lists". Diese Möglichkeit, Filter auf IP-Adressen und TCP/UDP-Ports zu definieren, bezieht sich jedoch nicht auf die ATM-Schnittstellen, sondern ausschließlich auf die Ethernet-Schnittstelle, die als Administrationszugang vorhanden ist.

3.3.2.3 Granularität der Zugriffskontrolle von Packet Screens in ATM-Netzen

Packet Screens werden auch in ATM-Netzen zur Kontrolle von IP-Dateneinheiten eingesetzt. Für die Zugriffskontrolle wird von Packet Screens in ATM-Netzen im wesentlichen auf dieselben Informationen zugegriffen wie von Packet Screens in herkömmlichen Netzen. Zusätzliche Überprüfungen beziehen sich lediglich auf den SNAP-Header, bei dem sich die Kontrolle jedoch auch darauf beschränkt sicherzustellen, daß in der AAL-5-Dateneinheit eine IP-Dateneinheit enthalten ist. Eine Kontrolle von Dateneinheiten anderer Protokolle, zum Beispiel IPX, wäre zwar prinzipiell möglich, aber bereits in herkömmlichen Netzen hat sich hierfür kaum ein Bedarf gezeigt, so daß solche Lösungen auch in ATM-Netzen kaum benötigt werden. Da "Native"-ATM-Anwendungen bisher noch keine Verbreitung gefunden haben, sind Aussagen zum Einsatz von Packet Screens für "Native"-ATM-Anwendungen schwierig. Da Adreßinformationen bei "Native"-ATM-Anwendungen jedoch bei der Signalisierung ausgetauscht werden, eignet sich ein Signalisierungskontroller besser zur Kontrolle von "Native"-ATM-Anwendungen als eine Packet Screen. Die in AAL-Rahmen übertragenen Nachrichten werden bei "Native"-ATM-Anwendungen in der Regel für Packet Screens zu komplex sein, so daß eine Kontrolle von "Native"-ATM-Anwendungen nur über Proxies möglich ist.

3.3.2.4 Packet Screen und Paket-Klassifikation

Ein über die bisherige Funktionalität von Packet Screens hinausgehende Erweiterung ist die Klassifikation von Paketen anhand von Sicherheitsrichtlinien, um hereinkommende Pakete unterschiedlichen herausgehenden virtuellen Verbindungen zuordnen zu können. Dadurch kann eine Packet Screen neben dem Blockieren oder Weiterleiten von IP-Datagrammen auch ähnlich wie ein Signalisierungskontroller einen alternativen Pfad zum Empfänger wählen. Während der Signalisierungskontroller jedoch nur virtuelle Verbindungen über alternative Pfade (s. auch Abschnitt 3.4.4: "Asymmetrische Kontrolle von Verbindungen") leiten kann, wird es durch die Paket-Klassifikation möglich, für unterschiedliche Dateneinheiten innerhalb derselben virtuellen Verbindung unterschiedliche Pfade zu wählen. IP-Dateneinheiten, für die nach den Sicherheitsrichtlinien eine Kontrolle durch einen Proxy erforderlich ist, können über eine virtuelle Verbindung an einen transparenten NHRP-Proxy (s. Abschnitt 3.3.3.3) weitergeleitet werden, während die übrigen IP-Dateneinheiten über eine andere virtuelle Verbindung am Proxy vorbeigeleitet werden.[62]

[62] Die Paket-Klassifikation in einer Packet Screen kann auch zur Lastverteilung auf parallele Bastionen eingesetzt werden (s. Abschnitt 5.2.2.5).

Prinzipiell wäre dieser Ansatz auch bei herkömmlichen Netzen bereits möglich, jedoch ist es dort wenig sinnvoll, IP-Dateneinheiten über eine andere Schnittstelle weiterzuleiten, wenn diese Dateneinheiten dort nicht von einem transparenten Proxy aufgefangen werden. Herkömmliche Proxies können diese Dateneinheiten hingegen nicht verarbeiten, da sie die IP-Adresse des (ursprünglichen) Empfängers und nicht die IP-Adresse der Bastion, auf der der Proxy eingesetzt wird, als Empfänger-Adresse haben. Bei einem ATM-Netz kann hingegen über eine virtuelle Verbindung ein transparenter NHRP-Proxy (s. Abschnitt 3.3.3.3) angesprochen werden. Die Paket-Klassifikation dient dabei für das Aufsplitten der über eine virtuelle Verbindung hereinkommenden IP-Datenströme auf verschiedene herausgehende virtuelle Verbindungen.

3.3.2.5 Fazit: Packet Screens in ATM-Netzen

Der Einsatz von Packet Screens ist auch zur Kontrolle von IP-Kommunikation in ATM-Netzen möglich. Klassische Packet Screens führen zu hohen Verzögerungen aufgrund der in der Packet Screen erforderlichen Segmentierung und Reassemblierung von IP-Dateneinheiten (s. Abschnitt 3.3.2.1). Eine sehr viel geringere Verzögerung verursacht die Filterung in einer "Cell Screen", da eine Reassemblierung nicht erforderlich ist und jeweils nur die erste Zelle einer IP-Dateneinheit untersucht wird. Eine Filterung auf Basis der ersten Zelle wird durch den Einsatz von IPv6 zukünftig nicht mehr möglich sein, da wegen der längeren IPv6-Headers der Transport-Header erst in der zweiten Zelle enthalten ist. Interessante neue Möglichkeiten ergeben sich für Packet Screens in ATM-Netzen, wenn die IP-Adressen der IP-Dateneinheiten bereits im voraus beim Aufbau der virtuellen Verbindung bekannt sind. Somit kann für die Kontrolle einer virtuellen Verbindung bereits eine geringe Anzahl von Filterregeln ausreichen, die beim Aufbau der virtuellen Verbindung ausgewählt wird.

3.3.3 Kontrollkomponenten auf Anwendungsebene

Auf der Anwendungsebene kann eine sehr viel detailliertere Zugriffskontrolle durchgeführt werden als über die einfache Kontrolle der Adreßinformationen durch eine Packet Screen. Bei herkömmlichen Firewalls werden auf der Anwendungsebene Proxy-Server (s. Abschnitt 2.3.1.2) und Anwendungs-Gateways (s. Abschnitt 2.3.1.3) eingesetzt.

3.3.3.1 Proxies und Anwendungs-Gateways in ATM-basierten IP-Netzen

Bei LANE, MPOA, CLIP und NHRP, bei denen das ATM-Netz zur Übertragung von IP-Dateneinheiten genutzt wird, können herkömmliche Proxy-Server und Anwendungs-Gateways unverändert eingesetzt werden. Eine Anpassung dieser Firewall-Komponenten an das ATM-Netz ist nicht notwendig, da bei allen vier Konzepten letztendlich IP-Dateneinheiten übertragen werden und das ATM-Netz als die unterhalb von IP liegende Infrastruktur verdeckt wird. Daher können prinzipiell weiterhin die in Abschnitt 2.3.2 vorgestellten herkömmlichen Firewall-Architekturen verwendet werden. Wobei dann allerdings ATM-spezifische Möglichkeiten und Risiken nicht berücksichtigt werden können.

Die Firewall-Komponenten dienen in ATM-Netzen beim Einsatz von LANE, MPOA, CLIP oder NHRP daher ausschließlich zum Schutz von IP-Netzen. Daß diese IP-Netze auf ATM-Netzen basieren, hat auf die Funktion der Firewall-Komponenten folgende Auswirkungen:

- In erster Linie ergeben sich durch die hohen Datendurchsätze Performanzprobleme für die Firewall-Komponenten. Firewall-Komponenten auf Anwendungsebene — Proxy-Server und Anwendungs-Gateways — werden auf handelsüblichen Workstations eingesetzt. Die Kontrolle der übertragenen Daten in einer normalen Workstation durch einen Prozeß auf Anwendungsebene erfordert bereits bei geringen Datendurchsätzen sehr leistungsfähige Systeme. Die unzureichende Performanz von Proxy-Servern und Anwendungs-Gateways ist jedoch nicht spezifisch für ATM-Netze, da es für andere Hochgeschwindigkeitsnetze wie zum Beispiel Fast- und Gigabit-Ethernet in gleicher Weise auftritt. Die Performanz von Proxy-Servern in einem CLIP-Netz wird im Abschnitt 4.4 untersucht.

- Bedingt durch die hohen Datendurchsätze in einem ATM-Netz werden neue Anwendungen möglich, für die bisher noch keine Proxy-Server verfügbar sind. Beispiele für neue Anwendungen sind Video- und Audio-Übertragungen sowie Konferenzsysteme [Schulzrinne et al. 96, Schulzrinne et al. 98]. Diese durch die hohen Datendurchsätze ermöglichten Anwendungen sind eine sehr indirekte Folge von ATM auf Proxy-Server und werden hier nicht weiter diskutiert.

- Beim Einsatz von Proxy-Servern und Anwendungs-Gateways müssen die im Abschnitt 3.2 diskutierten Schwachstellen von ATM-Netzen zusätzlich zu den bereits bekannten Problemen von IP-Netzen berücksichtigt werden. Die Absicherung dieser Schwachstellen muß durch durch die Kombination mit anderen Mechanismen zu geeigneten Firewall-Architekturen (s. Abschnitt 3.4) erreicht werden.

3.3.3.2 "Native"-ATM-Proxies

Für "Native"-ATM-Anwendungen gibt es noch keine Firewall-Komponenten, auf die zurückgegriffen werden kann. Grundsätzlich kann beim Entwurf von Proxies für "Native"-ATM-Anwendungen jedoch ähnlich vorgegangen werden wie bei herkömmlichen Proxies. Auch bei "Native"-ATM muß der Proxy eine Verbindung zunächst annehmen, den Client authentisieren, die Autorisierung des Clients überprüfen und eine Verbindung zum gewünschten Server aufbauen. Die anschließend übertragenen Daten werden vom Proxy auf der Anwendungsebene überprüft, bevor sie weitergeleitet werden.

In IP-Netzen werden Proxies hauptsächlich für die Absicherung von TCP-basierten Diensten eingesetzt. Für Datagramm-orientierte UDP-basierte Anwendungen sind Proxies hingegen ungeeignet, da es keinen Verbindungsaufbau gibt, nach dem vom Proxy eine Authentisierung des Clients vorgenommen werden kann. Diese wesentliche Einschränkung von Proxies stellt beim Einsatz für "Native"-ATM-Anwendungen kein Problem dar, weil ATM als verbindungsorientiertes Netz in jedem Fall zuerst einen Verbindungsaufbau erfordert.

3.3.3.2.1 Dienstgüte

Eine Herausforderung an zukünftige Firewalls wird die Einführung von Anwendungen sein, die eine garantierte Dienstgüte benötigen. Bei Audio- und Video-Anwendungen müssen zum Beispiel ein festgelegter Datendurchsatz und eine maximale Verzögerung bei der Übertragung eingehalten werden. Im Gegensatz zum "Best-Effort" in heutigen Netzen, bei dem die Hauptbelastung bei den Endsystemen liegt (Fehlererkennung, "Retransmission"), erfordert die Einhaltung der Dienstgüte eine Unterstützung von allen Netzwerkknoten auf dem Pfad zwischen Sender und Empfänger. Firewalls, die eine Einhaltung von ausgehandelten Dienstgüteparametern unterstützen, sind derzeit nicht bekannt.

"Native"-ATM-Proxies sind die einzigen Firewall-Komponenten, die zunächst von einer von ATM ermöglichten garantierten Dienstgüte betroffen sind. Die Problematik der Unterstützung einer garantierten Dienstgüte stellte sich beim Einsatz IP-basierter Proxies in einem ATM-Netz nicht, da IP auch in einem ATM-Netz mit CLIP, NHRP, LANE oder MPOA nur einen "Best-Effort"-Dienst erbringt.[63] Da auch die im Abschnitt 3.3.2 diskutierten Packet Screens für ATM-Netze nur eine Kontrolle von IP-Datenflüssen durchführen, ist auch dort die Unterstützung einer garantierten Dienstgüte nicht erforderlich. Signalisierungsfilter (s. Abschnitt 3.3.1) überwachen lediglich die Signalisierungsnachrichten und keine Datenflüsse,

[63] Ansätze, über die Reservierung von Betriebsmitteln auch bei IP eine angeforderte Dienstgüte zu garantieren (z. B. RSVP), sind prinzipiell unabhängig von ATM und sollen daher an dieser Stelle nicht diskutiert werden.

so daß auch beim Signalisierungsfilter keine garantierte Dienstgüte erforderlich ist.

Ein ATM-Netz kann für eine virtuelle Verbindung eine bestimmte Dienstgüte garantieren, indem auf allen Leitungen und Switches auf dem Pfad zwischen Sender und Empfänger die für die Garantie der Dienstgüte notwendigen Betriebsmittel exklusiv für diese Verbindung reserviert werden. Die Dienstgüte wird für die virtuelle Verbindung zwischen den beiden Endgeräten erbracht. Wenn die virtuelle Verbindung durch einen Proxy überwacht werden muß, existiert mit der Bastion, auf der der Proxy eingesetzt wird, ein drittes Endgerät, das den Pfad zwischen den beiden anderen Endgeräten in zwei Teile aufspaltet. Damit die "Native"-ATM-Anwendung diese Verbindung nutzen kann, muß trotz der Aufspaltung in zwei Verbindungen und der Kontrolle durch den Proxy insgesamt immer noch dieselbe Dienstgüte erzielt werden.

PNNI unterscheidet bei der Dienstgüte zwischen Attributen und Metriken. Ein Attribut beschreibt eine bestimmte Eigenschaft eines Pfades oder eines Knotens, zum Beispiel die zur Verfügung stehende Zellrate ("Available Cell Rate") oder die maximale Zellrate ("Maximum Cell Rate") [ATM Forum 96c, S. 97]. Metriken hingegen sind kumulativ, müssen somit für den Pfad vom Sender zum Empfänger aufaddiert werden. Beispiele für Metriken sind die maximale Zellverzögerung ("Maximum Cell Transfer Delay") und die Zellverzögerungsvarianz ("Cell Delay Variation"). Bei einem Pfad über einen Proxy müssen beide virtuellen Verbindungen und der Proxy den angeforderten Dienstgüteattributen genügen. Sehr viel schwieriger ist es, Metriken bei Verbindungen über einen Proxy zu berücksichtigen. Hier dürfen die beiden Verbindungen und der Proxy in der Summe maximal den bei der Dienstgüte geforderten Wert erreichen. Die Auswahl eines Pfades über einen Proxy, der zum Beispiel die geforderte Zellverzögerung unterschreitet, ist mit PNNI nicht möglich, da PNNI nur Pfade zwischen Endsystemen finden kann, die einer angeforderten Dienstgüte entsprechen. PNNI kann also jeweils nur einen Pfad zwischen Endsystem und Bastion finden, der einer bestimmten Dienstgüte genügt. Da die Entfernung zwischen Endsystem und Proxy in der Regel nicht auf beiden Seiten gleich lang ist, kann das Problem auch nicht dadurch gelöst werden, daß für jede der beiden Verbindungen jeweils die Hälfte der insgesamt akzeptablen Zellverzögerung angefordert wird. Da eine "Native"-ATM-Anwendung nur die insgesamt benötigte Dienstgüte angeben kann, muß die Auswahl eines geeigneten Pfades über einen Proxy durch das ATM-Netz erfolgen. Damit PNNI einen Pfad mit der angeforderten Dienstgüte über den Proxy erstellen kann, darf der Proxy bei der Auswahl des Pfades nicht als Endsystem betrachtet werden, sondern muß wie ein weiterer ATM-Switch auf dem Weg zum Empfänger betrachtet werden, auch wenn schließlich zwei virtuelle Verbindungen aufgebaut werden. Ein vielversprechender Ansatz zur Lösung dieses Problems ist es, Dienstgüte und Sicherheitsdienstgüte zusammen für die Bestimmung eines Pfades zu berücksichtigen (s. Abschnitt 3.3.1.2.1).

Zusätzlich zum Problem der Auswahl eines Pfades, der der geforderten Dienstgüte genügt, ist es bisher auch nicht möglich, eine bestimmte Dienstgüte durch einen Proxy zu garantieren. Proxies werden auf einfachen Workstations eingesetzt, deren Betriebssysteme keine garantierte Dienstgüte zur Verfügung stellen können. Somit kann bisher nur eine "Best-Effort"-Dienstgüte erzielt werden. Selbst wenn das Betriebssystem eine garantierte Dienstgüte ermöglichen würde, kann dies durch heutige Proxies nicht ausgenutzt werden, da die heute bei Proxies eingesetzten Kontrollmechanismen in der Regel nicht die Verarbeitung der Daten innerhalb einer festen Zeitvorgabe bzw. innerhalb einer festen Anzahl von CPU-Taktzyklen garantieren können. Für die Einhaltung einer garantierten Dienstgüte werden daher sowohl die Unterstützung des Betriebssystems als auch verbesserte Kontrollmechanismen des Proxies benötigt.

3.3.3.2.2 Absicherung von "Native"-ATM-Anwendungen durch Proxies

Zur Zeit ist die Verbreitung von "Native"-ATM-Anwendungen noch sehr gering. Das ATM-Forum hat bisher neben einem allgemeinen Modell für "Native"-ATM-Anwendungen [ATM Forum 96b] nur die Telefonie über ATM ("Voice and Telephony over ATM" [ATM Forum 97b] und "Video on Demand" (VoD) [ATM Forum 97a] als "Native"-ATM-Anwendungen spezifiziert.

Bei der Telefonie über ATM wird eine virtuelle Verbindung mit 64 KBit/s für die Übertragung kodierter Audiosignale (nach "ITU-T Recommendation G.711" [ITU 93]) zwischen den Endgeräten verwendet. Die Eröffnung dieser Verbindung erfolgt über die normale Signalisierung mit UNI 4.0 [ATM Forum 96d]. Die Kontrolle der Signalisierungsnachrichten ist daher bereits durch einen Signalisierungsfilter (s. Abschnitt 3.3.1) abgedeckt. Eine Übertragung von Kontrollnachrichten, die über diese Signalisierung hinausgeht, ist für die Telefonie über ATM nicht erforderlich, so daß nach dem Verbindungsaufbau lediglich kodierte Audiosignale übertragen werden. Als Aufgabe eines Proxies bliebe somit nur die Kontrolle dieser Audio-Informationen.

Für "Video on Demand" (VoD) reicht im Gegensatz zur Telefonie eine Verbindung allein nicht aus. Neben dem Video-Server, auf dem Videos gespeichert sind, und dem Client, der ein Video darstellt, wird noch eine Komponente zur VoD-Sitzungssteuerung benötigt, bei dem sich alle Video-Server registrieren und Clients Informationen zu verfügbaren Video-Servern erhalten. Bei diesen drei Komponenten handelt es sich um logische Komponenten, die nicht zwangsläufig auf unterschiedlichen Endsystemen implementiert werden müssen. Zwischen den drei Komponenten werden vier Verbindungen verwendet (s. Abb. 3.13).[64] Die Video-Verbindung dient zum Transport der kodierten Video-Signale (Bild und Ton).

[64] Die Pfeile in Abbildung 3.13 geben die Datenflußrichtung an.

Auf der Kontrollverbindung zwischen Client und Video-Server werden Nachrichten zur Steuerung der Video-Verbindung (Auswahl von Programmen, "Pause/Resume" usw.) übertragen. Die Verbindungen zur VoD-Sitzungssteuerung dienen der Registrierung und der Anforderung von Betriebsmitteln. Das auf den Kontrollverbindungen eingesetzte Protokoll wird nicht direkt in [ATM Forum 97a] spezifiziert. Vorgeschlagen wird der Einsatz der Protokolle "Digital Storage Media – Command and Control" (DSM-CC) [Balabanian et al. 96, ISO 97] oder "ITU-T Recommendation H.245" [ITU 97]. Von diesen Verbindungen ist nur für die Video-Verbindung eine garantierte Dienstgüte erforderlich. Für die unidirektionale Video-Verbindung werden Zellrate, Zellverzögerung und Zellverzögerungsvarianz sowie mehrere Qualitätsmerkmale bezüglich des Zellverlustes angefordert [ATM Forum 97a, S. 17].

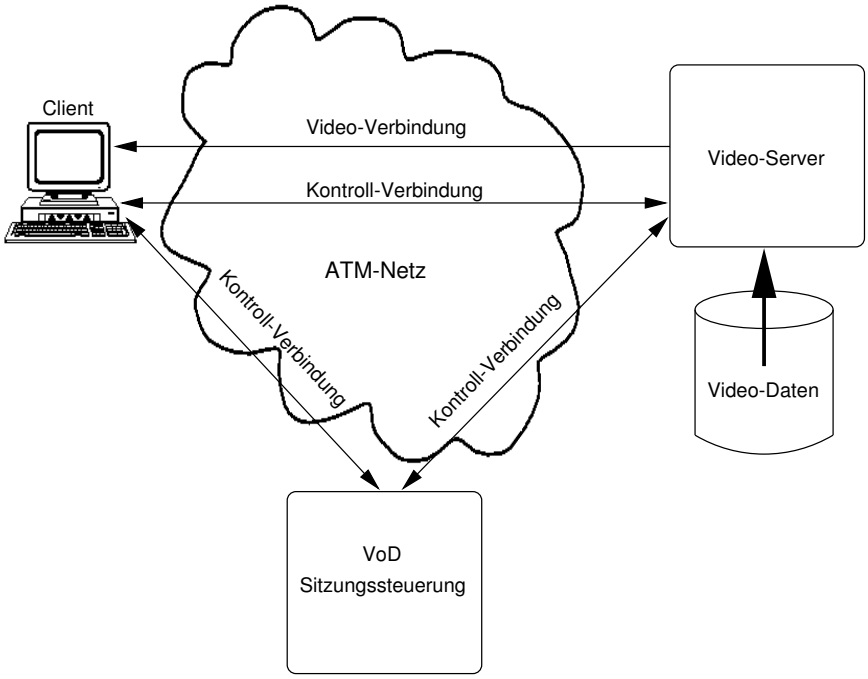


Abbildung 3.13: "Video on Demand" [ATM Forum 97a]

Bei den für VoD benötigten Verbindungen ist die Überwachung der Kontrollverbindungen am wichtigsten, da hierüber eine Steuerung des Video-Servers, der VoD-Sitzungssteuerung und auch des Clients durchgeführt wird. DSM-CC ist ein sehr leistungsfähiges Protokoll für verschiedenste Anwendungen vom VoD bis zum "Home Shopping". Es enthält eine Reihe von Funktionen, die ein sehr hohes Risiko darstellen. Zu denen in DSM-CC spezifizierten Funktionen zählt eine automatische Konfiguration des Clients, das Herunterladen von Programmen zum Client und die Fernausführung von Funktionen ("Remote Procedure Call", RPC) durch den Client auf dem Server und auch in Gegenrichtung durch den Server auf

dem Client [Balabanian et al. 96]. Durch die Kombination der Funktionen zum Herunterladen von Programmen mit der Funktion der Fernausführung enthält dieses Protokoll sehr weitreichende Angriffsmöglichkeiten, da prinzipiell beliebige Programme übertragen und anschließend ausgeführt werden können.[65] Die Überwachung dieser Verbindungen durch einen Proxy beinhaltet keine Anforderungen an eine zu erbringende Dienstgüte, so daß der Entwurf von Proxies für die auf den Kontrollverbindungen eingesetzten Protokolle herkömmlichen Ansätzen folgen kann. Ein Proxy zur Überwachung der Video-Verbindung muß hingegen die angeforderte Dienstgüte einhalten, so daß hierfür neue Ansätze entwickelt werden müssen.

Bei der Überwachung der Video-Verbindung ist zunächst zu überprüfen, welche Risiken durch eine Überwachung beseitigt oder zumindest verringert werden können. Die Video-Verbindung enthält nur Bild- und Tondaten einschließlich der notwendigen Synchronisationsinformationen. Die Kontrolle der Video-Verbindung kann sich also nur gegen die unberechtigte Nutzung von Betriebsmitteln oder gegen die Nutzung der Video-Verbindung zur Übertragung verbotener Inhalte wenden. Die unberechtigte Nutzung von Betriebsmitteln, etwa durch Reservieren von sehr hohen Zellraten, muß bereits beim Verbindungsaufbau durch einen Signalisierungsfilter überwacht werden und liegt somit außerhalb der Aufgaben eines Proxies. Durch die Kontrolle der übertragenen Videodaten kann ein Proxy gegen den unerlaubten Export von vertraulichen Informationen oder zum Schutz der internen Nutzer vor ungeeigneten Inhalten eingesetzt werden. Abgesehen von den Problemen bei der Einhaltung der Dienstgüte, die durch diese Filterung entstehen, ist die Erkennung und Filterung bestimmter Gesten oder Bild-Sequenzen noch immer ein Forschungsproblem. Der erreichte Sicherheitsgewinn durch die Filterung bleibt dabei gering, so daß bezweifelt werden muß, daß die Filterung der Video-Verbindung bei "Video on Demand" überhaupt sinnvoll ist.

Bei Video-Konferenzsystemen und bei der Telefonie über ATM stellt sich bei der Überwachung der Datenverbindungen zusätzlich die Frage der rechtlichen Zulässigkeit dieser Überwachung.

Fazit: Die Untersuchung von "Native"-ATM-Anwendungen hat gezeigt, daß derzeit kein Bedarf für die Garantie bestimmter Dienstgüteparameter durch Proxies besteht. Bei der "Native"-ATM-Anwendung "Video on Demand" reicht eine Überwachung der Kontrollverbindungen aus. Bei der Datenverbindung ist eine Kontrolle hingegen nicht sinnvoll, und es kann daher eine direkte virtuelle Verbindung zwischen Video-Server und Client zugelassen werden. Da nur bei

[65] Die Übermittlung von Programmen und das Fernausführen von Programmen ist zusammen hinreichend für die Implementierung eines "Wurms" [Ellermann et al. 91].

diesen Datenverbindungen Dienstgüteanforderungen eingehalten werden müssen, erstreckt sich die Kontrolle des Proxies nur auf Kontrollverbindungen, für die keine Dienstgüteanforderungen bestehen. Auch bei der Telefonie über ATM kann durch die Überwachung der Datenverbindung keine Erhöhung der Sicherheit erreicht werden. Eine zusätzliche Kontrollverbindung wird für die Telefonie nicht benötigt, so daß Steuerungsinformationen, die zur Filterung geeignet sind, ausschließlich in den Signalisierungsnachrichten beim Auf- und Abbau der virtuellen Verbindung vorhanden sind.

3.3.3.2.3 Transparenzanforderungen an "Native"-ATM-Proxies

Die Kommunikation über ein ATM-Netz erfolgt in zwei getrennten Phasen. Zuerst muß eine virtuelle Verbindung zum gewünschten Empfänger eröffnet werden, über die in der zweiten Phase Daten übertragen werden. Die Adressen der Kommunikationspartner werden nach einem erfolgreichen Verbindungsaufbau nicht mehr benötigt und die Daten können direkt über die virtuelle Verbindung ausgetauscht werden. Da die Adressen bei der Kommunikation über eine bestehende virtuelle Verbindung nicht mehr verwendet werden, ist es für die kommunizierenden Endsysteme nicht sichtbar, wenn die Kommunikation nicht direkt mit dem Kommunikationspartner, sondern über einen Proxy erfolgt. Daher muß nur beim Verbindungsaufbau für die transparente Einbindung des Proxies in die virtuelle Verbindung gesorgt werden. Dies ist der entscheidende Unterschied zwischen IP-Netzen und ATM-Netzen. Anders als bei ATM-Netzen müssen bei IP-Netzen alle Dateneinheiten über Adreßinformationen verfügen, so daß die Einbindung eines Proxies während der gesamten Kommunikation erfolgen muß. Dies wird in IP-Netzen dadurch erreicht, daß der Client den Proxy über dessen IP-Adresse direkt adressiert. Da der Client die IP-Adresse und TCP-Portnummer des Proxies benötigt,[66] kann die Einbindung von Proxies in IP-Netzen üblicherweise nicht transparent erfolgen.

Zunächst wird die transparente Einbindung eines Proxies für eine "Native"-ATM-Anwendung diskutiert. Im Abschnitt 3.3.3.3 wird anschließend auf die transparente Einbindung IP-basierter Proxies bei CLIP und NHRP eingegangen.

Der Verbindungsaufbau einer virtuellen Verbindung besteht aus drei Signalisierungsnachrichten — SETUP, CONNECT und CONNECT_ACKNOWLEDGE (s. Abb. 2.14, S. 50). Der Client sendet eine SETUP-Nachricht über den UNI-Signalisierungskanal zu dem Switch, an den der Client angeschlossen ist. In den Informationselementen der SETUP-Nachricht sind die ATM-Adresse des Empfängers ("Called Party Number"), Angaben zur Verarbeitung in den End-

[66] Genaugenommen wird die IP-Adresse der Bastion benötigt, auf der der Proxy eingesetzt wird.

systemen (AAL-Parameter, optional auch BLLI und BHLI),[67] Parameter zur benötigten Dienstgüte sowie optional die ATM-Adresse des Absenders ("Calling Party Number") enthalten. Durch die CONNECT-Nachricht wird dem Client bestätigt, daß der Verbindungsaufbau erfolgt ist. In der CONNECT-Nachricht sind keine ATM-Adreßinformationen enthalten. Ob die CONNECT-Nachricht von dem gewünschten Endsystem oder von einer anderen Komponente im Netz (z. B. einem Proxy) erzeugt wurde, ist für den Client daher nicht erkennbar. Den Empfang der CONNECT-Nachricht und den Verbindungsaufbau bestätigt der Client schließlich mit einer CONNECT_ACKNOWLEDGE-Nachricht, die nur zum lokal angeschlossenen Switch aber nicht bis zum Endgerät des Kommunikationspartners übertragen wird.

Auf der Seite des Servers sieht der Verbindungsaufbau ähnlich aus. Der Server erhält eine SETUP-Nachricht, die seine ATM-Adresse als Empfänger-Adresse ("Called Party Number") enthält. Die Absender-Adresse ("Calling Party Number") kann ebenfalls in der SETUP-Nachricht enthalten sein, ist aber für den Verbindungsaufbau nicht unbedingt erforderlich. Mit einer CONNECT-Nachricht wird der Verbindungsaufbau vom Server bestätigt. Als Bestätigung der CONNECT-Nachricht erhält der Server vom Switch schließlich ein CONNECT_ACKNOWLEDGE. Die beim SETUP angegebene Absender-Adresse wird von Anwendungen verwendet, um den Kommunikationspartner zu identifizieren. Die Absender-Adresse dient dabei nur zur Information des Empfängers, wird aber nicht für den Verbindungsaufbau benötigt. Insbesondere wird die Absender-Adresse nicht für das Zurücksenden der CONNECT-Nachricht benötigt. Daher kann als "Calling Party Number" eine beliebige Adresse angegeben werden oder die Absender-Adresse kann sogar ganz weggelassen werden. Für einen Proxy bietet sich beim Verbindungsaufbau zum Server an, die Adresse des Clients oder alternativ die eigene Adresse zu verwenden. Der Proxy wählt seine eigene ATM-Adresse als Absender-Adresse, wenn die Existenz des Clients gegenüber dem Server verdeckt werden soll. Der Proxy tritt dann gegenüber dem Server als Client auf. Im anderen Fall, wenn der Proxy die Adresse des Clients als "Calling Party Number" angibt, ist für den Server nicht erkennbar, daß die virtuelle Verbindung durch den Proxy kontrolliert wird. Unabhängig davon, ob der Proxy *gegenüber dem Server* als Absender-Adresse die Adresse des Clients oder seine eigene Adresse verwendet, bleibt die Kontrolle des Proxies *gegenüber dem Client* transparent.

In einem IP-Netz kann ein Proxy gegenüber einem Server nicht wie hier beschrieben eine beliebige Absender-Adresse verwenden, da die Absender-Adresse vom Server als Empfänger-Adresse für die zurückgesendeten Dateneinheiten benötigt wird. In einem ATM-Netz wird die vom Server zurückgesendete CONNECT-Nachricht über den UNI-Signalisierungskanal zurück an den Switch gesendet.

[67] Diese Informationselemente wurden im Abschnitt 3.3.1.1.1 vorgestellt.

Der Switch identifiziert die Nachricht über die "Call Reference". Alle Signalisierungsnachrichten, die zum Aufbau einer virtuellen Verbindung verwendet werden, verfügen mit der "Call Reference" über einen lokal eindeutigen Bezeichner. Die "Call Reference" wird vom Sender einer SETUP-Nachricht und von jedem Switch, der die SETUP-Nachricht weiterleitet, erneut vergeben. Dadurch gibt es für eine Signalisierung auf jedem Abschnitt des Pfades eine eindeutige "Call Reference". Für das Zurücksenden von Signalisierungsnachrichten (CONNECT, RELEASE usw.) an den Absender kann anhand der "Call Reference" der Pfad der SETUP-Nachricht zurückverfolgt werden, ohne daß hierfür die Absender-Adresse benötigt wird.[68] Benutzerdaten sendet der Server ausschließlich über die aufgebaute virtuelle Verbindung, ohne daß hierfür noch die ATM-Adressen der Kommunikationspartner benötigt werden.

Die Signalisierung beim Aufbau einer virtuellen Verbindung läßt einen großen Spielraum für die Einbindung von Proxies. Die Entscheidung, ob ein Proxy zur Kontrolle einer virtuellen Verbindung benötigt wird, kann daher vom Signalisierungskontroller beim Verbindungsaufbau getroffen werden. Die Einbeziehung des Clients oder des Servers in diese Entscheidung ist nicht notwendig. Für den Signalisierungskontroller bestehen alle in Abschnitt 3.3.1.2.2 dargestellten Möglichkeiten zur Einbindung eines Proxies. Beim "Einschleifen" einer Sicherheitskomponente und beim Einfügen einer Sicherheitskomponente durch Ersetzen der Empfänger-Adresse kann der Proxy als Endsystem direkt adressiert werden. Der Proxy bzw. die Bastion, auf der der Proxy eingesetzt wird, kann sich wie ein normales Endsystem verhalten. Dadurch werden interne Systeme durch den Proxy verdeckt, da der Proxy nach außen hin das anfragende Endsystem ist.

Beim Einbinden von Sicherheitskomponenten über die Ergänzung der DTL wird der in der DTL kodierte Pfad vom Signalisierungskontroller so verändert, daß der Proxy, der eigentlich ein Endsystem ist, als weiterer Knoten in der DTL auftaucht. Der Proxy bzw. die Bastion darf bei dieser Lösung nicht als Endsystem der Endpunkt einer Verbindung sein und muß sich stattdessen zu den benachbarten Netzkomponenten hin wie ein Switch verhalten. Für die Bastion bedeutet dies, daß die Signalisierung nicht über UNI sondern über PNNI erfolgt. Die Bastion muß hierfür jedoch nicht alle Funktionen von PNNI implementieren, da an die Bastion keine Endgeräte angeschlossen werden, die Bastion nur über

[68] Bei einer Verwendung der Absender-Adresse anstelle der "Call Reference" könnte nicht garantiert werden, daß für die als Antwort zurückgesendeten Signalisierungsnachrichten (CONNECT, RELEASE usw.) erneut derselbe Pfad verwendet wird. Signalisierungsnachrichten, die sich auf dieselbe virtuelle Verbindung beziehen, müssen immer auf demselben Pfad übertragen werden, da sie nicht nur für die Endsysteme von Bedeutung sind. Die Signalisierungsnachrichten werden auch von den Switches ausgewertet, damit zum Beispiel die für die Verbindung reservierten VPI/VCI-Paare und weitere Betriebsmittel reserviert beziehungsweise anschließend wieder freigegeben werden können. Außerdem ergäbe sich für den Absender selbst ein Zuordnungsproblem, wenn gleichzeitig mehrere Verbindungen aufgebaut werden.

zwei Schnittstellen verfügt und die Bastion zudem auch keinen Übergang zu einer anderen "Peer-Group" bildet. Da die Bastion sich im ATM-Netz logisch wie ein Switch verhält, soll hier die Bezeichnung "**Bastion Switch**" verwendet werden.

Fazit: Mit der Einführung von "Native"-ATM-Anwendungen wird es zukünftig auch einen Bedarf für "Native"-ATM-Proxies geben. Die Möglichkeit von "Native"-ATM-Anwendungen, Dienstgüteparameter anzufordern, stellt eine neue Herausforderung für Proxies dar, jedoch kann bei den heute spezifizierten "Native"-ATM-Anwendungen kein Sicherheitsgewinn durch die Überwachung von Video- oder Audio-Verbindungen erzielt werden. Wesentlicher Unterschied von "Native"-ATM-Proxies im Vergleich zu herkömmlichen Proxies ist die Möglichkeit zur transparenten Einbindung des Proxies. Während herkömmliche Proxies direkt angesprochen werden müssen und daher weder für den Client noch den Server transparent sein können, ist eine Kontrolle durch einen "Native"-ATM-Proxy möglich, ohne daß Client oder Server Kenntnis von dem Proxy erlangen müssen.

3.3.3.3 Transparente Proxies in NHRP-Netzen

Bisher wurde das transparente Einbinden von Proxies für die Absicherung von "Native"-ATM-Anwendungen diskutiert. Ein Signalisierungskontroller kann auf die gleiche Weise auch virtuelle Verbindung umleiten, die zur Übertragung von IP-Dateneinheiten zum Beispiel von CLIP oder NHRP genutzt werden. Da die übertragenen IP-Dateneinheiten jedoch mit den IP-Adressen selbst ebenfalls Adreßinformationen enthalten, ergeben sich zwei wesentliche Unterschiede zu den bisher diskutierten "Native"-ATM-Anwendungen. Einerseits enthalten die vom Client gesendeten IP-Dateneinheiten die IP-Adresse des vom Client gewünschten Empfängers. Der Proxy muß also in der Lage sein, zusätzlich zu seiner eigenen Adresse auch an beliebige andere IP-Adressen gerichtete IP-Datagramme anzunehmen. Der zweite Unterschied ist, daß der IP-basierte Proxy keinen direkten Zugriff auf die virtuellen Verbindungen hat und daher eine zweite, bereits vom Signalisierungskontroller zum Server aufgebaute Verbindung gar nicht nutzen kann. Stattdessen sendet der Proxy lediglich IP-Dateneinheiten an die IP-Adresse des Servers, für die die Bastion selbst über NHRP die ATM-Adresse des Servers auflöst und zu diesem Server eine virtuelle Verbindung aufbaut. Es macht somit keinen Sinn, die Verbindung zum Server bereits durch den Signalisierungskontroller aufbauen zu lassen.

Die Einbindung eines IP-basierten Proxies durch einen Signalisierungskontroller erfolgt über das Ändern der Empfänger-Adresse (s. Abschnitt 3.3.1.2.4). Statt zum Server wird die virtuelle Verbindung zur Bastion aufgebaut. Der Signalisierungskontroller baut keine zweite virtuelle Verbindung von der Bastion zum Server auf, da diese Verbindung vom Proxy nicht verwendet werden kann. Auch

die Benachrichtigung des Proxies über die ursprünglich in der Signalisierungsnachricht enthaltene ATM-Empfänger-Adresse ist nicht erforderlich, da die Bastion die ATM-Adresse des Servers über eigene NHRP-Anfragen selbständig ermittelt.[69] Das Einbinden eines IP-basierten Proxies erfordert beim Signalisierungskontroller einen geringeren Aufwand als das Einbinden eines Proxies für "Native"-ATM-Anwendungen, bei denen der Signalisierungskontroller auch für den Verbindungsaufbau zwischen Bastion und Server sorgen muß. Der Eingriff des Signalisierungskontrollers zum Einbinden eines IP-basierten Proxies führt zu der in Abbildung 3.14 dargestellten virtuellen Verbindung zwischen Client und Proxy.

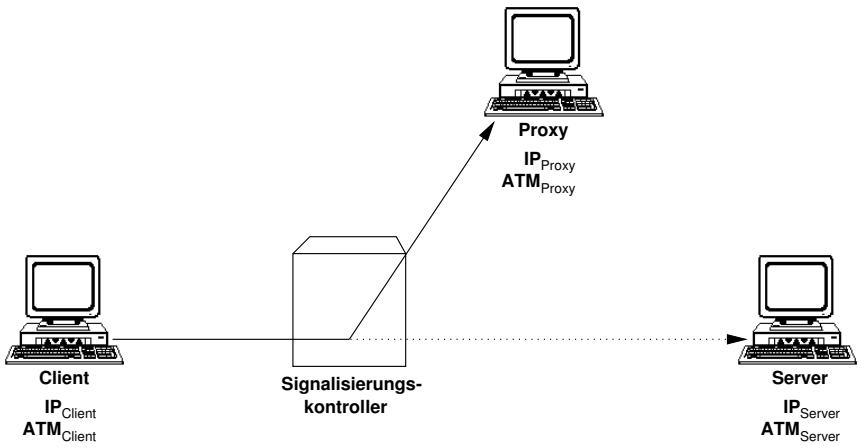


Abbildung 3.14: Einbindung eines IP-basierten Proxies durch einen Signalisierungskontroller

Der Client verwendet die virtuelle Verbindung anschließend zur Übertragung von an den Server gerichteten IP-Dateneinheiten. Dabei ist es für den Client unerheblich, daß die virtuelle Verbindung nicht direkt zum Server führt. Über NHRP aufgebaute virtuelle Verbindungen führen entweder direkt zum gewünschten Empfänger oder zu einer Netzkomponente, die die Übertragung der IP-Datagramme zum Empfänger übernimmt. Dieses Weiterleiten von IP-Datagrammen ist zum Beispiel auch dann erforderlich, wenn der Empfänger nicht selbst an ein ATM-Netz angeschlossen ist und daher keine direkte virtuelle Verbindung aufgebaut werden kann. Bei der Bastion, auf der der IP-basierte Proxy eingesetzt wird, kommen somit über die virtuelle Verbindung vom Client IP-Datagramme mit der Empfänger-Adresse IP_{Server} und der Absender-Adresse IP_{Client} an. IP-Datagramme, die nicht an eine IP-Adresse des Systems selbst gerichtet sind,

[69] Die Bastion muß vom Signalisierungskontroller als ATM-Endgerät angesprochen werden. Der Einsatz eines IP-basierten Proxies auf einem "Bastion Switch" (s. Abschnitt 3.3.3.2.3) ist somit nicht möglich.

werden von der IP-Implementation eines Systems entweder vernichtet oder über eine andere Schnittstelle des Systems an den Empfänger weitergeleitet ("Forwarding"). Beide Fälle sind auf der Bastion nicht erwünscht. Um dem Proxy den Empfang von Daten des Clients zu ermöglichen, muß die Bastion IP-Datagramme annehmen und verarbeiten können, die an andere IP-Adressen als die IP-Adresse der Bastion selbst gerichtet sind.

Die Verarbeitung von IP-Datagrammen, die an fremde IP-Adressen gerichtet sind, kann auf zwei Wegen ermöglicht werden. Entweder muß eine spezielle IP-Implementation den Empfang dieser Dateneinheiten ermöglichen oder eine zusätzliche "Network Address Translation"-(NAT)-Komponente muß die IP-Adressen durch die IP-Adresse der Bastion austauschen, bevor die Verarbeitung durch eine Standard IP-Implementation erfolgt. Zunächst wird der Einsatz einer NAT-Komponente zum Austausch der IP-Adresse diskutiert. Anschließend wird kurz dargelegt, welche Unterschiede bei einer spezialisierten IP-Implementation gegenüber der NAT-Lösung bestehen.

Der Einsatz von Proxies beschränkt sich üblicherweise auf TCP-basierte Anwendungen. Daher wird im folgenden von einem TCP-basierten Proxy ausgegangen, die hier beschriebenen Konzepte können jedoch auch auf UDP-basierte Proxies ausgedehnt werden. Der Client beginnt nach dem Aufbau der virtuellen Verbindung mit dem Senden einer IP-Dateneinheit an den Server. Die IP-Dateneinheit enthält die IP-Adressen von Client (IP_{Client}) und Server (IP_{Server}) sowie die TCP-Portnummern vom Client (TCP_{Client}) und vom angesprochenen Server (TCP_{Server}). Insgesamt ergeben sich die Adreßinformationen als 4-Tupel $< IP_{Client}, IP_{Server}, TCP_{Client}, TCP_{Server} >$. Unter der Annahme, daß der Proxy dieselbe Portnummer verwendet wie der angesprochene Server, kann die TCP-Portnummer (TCP_{Server}) erhalten bleiben und durch die NAT-Komponente muß nur die IP-Adresse (IP_{Server}) durch die Adresse des Proxies (IP_{Proxy}) ersetzt werden. Die NAT-Komponente speichert die Abbildung zwischen $< IP_{Client}, IP_{Server}, TCP_{Client}, TCP_{Server} >$ und $< IP_{Client}, IP_{Proxy}, TCP_{Client}, TCP_{Server} >$ für die Verarbeitung nachfolgender IP-Dateneinheiten und für die Verarbeitung der zum Client zurückgesendeten Dateneinheiten. Die aus der Abbildung resultierende IP-Dateneinheit kann von der TCP/IP-Implementation der Bastion wie eine herkömmliche Dateneinheit verarbeitet werden. Die in der entgegengesetzten Richtung gesendeten Dateneinheiten werden ebenfalls von IP und TCP wie in einem herkömmlichen System verarbeitet. Bevor die IP-Dateneinheit über die virtuelle Verbindung an den Client gesendet werden kann, ändert die NAT-Komponente die IP-Absender-Adresse von (IP_{Proxy}) auf (IP_{Server}). Die einzusetzende IP-Adresse des Servers erhält die NAT-Komponente aus einem einfachen "Lookup" mit den IP- und TCP-Adreßinformationen, der von IP übergebenen Dateneinheit. Auf diese Weise kann eine TCP-Verbindung zwischen Client und Proxy aufgebaut werden.

Für den Aufbau der zweiten TCP-Verbindung vom Proxy zum Server benötigt der Proxy die IP-Adresse des Servers. Die Adresse des Servers wird bei herkömmlichen Proxies entweder manuell konfiguriert, interaktiv durch den Benutzer eingegeben oder dem Proxy durch den Client mitgeteilt. Bei der manuellen Konfiguration der Server-Adresse kann über jeden Proxy nur ein einziger Server angesprochen werden. Die Übertragung der Adresse durch den Client an den Server ist im Gegensatz zur interaktiven Eingabe der Adresse zumindest gegenüber dem Benutzer transparent, erfordert jedoch den Einsatz spezialisierter Clients. Im Gegensatz zu diesen herkömmlichen Proxies ist bei der hier beschriebenen Lösung die IP-Adresse des Servers in der beschriebenen NAT-Komponente vorhanden und muß daher vom Client nicht zusätzlich übertragen werden. Die NAT-Komponente wird um eine Schnittstelle ergänzt, an der der Proxy mit den ihm vorliegenden Adreßinformationen ($< IP_{Client}, IP_{Proxy}, TCP_{Client}, TCP_{Server} >$) die IP-Adresse und die TCP-Portnummer des Servers abfragen kann. Die TCP-Portnummer wird nur benötigt, wenn Proxy und Server nicht an derselben Portnummer installiert sind und die NAT-Komponente daher auch die TCP-Portnummer des Servers ersetzen muß. Der Datenfluß über einen IP-basierten Proxy in einem ATM-Netz ist in Abbildung 3.15 wiedergegeben.

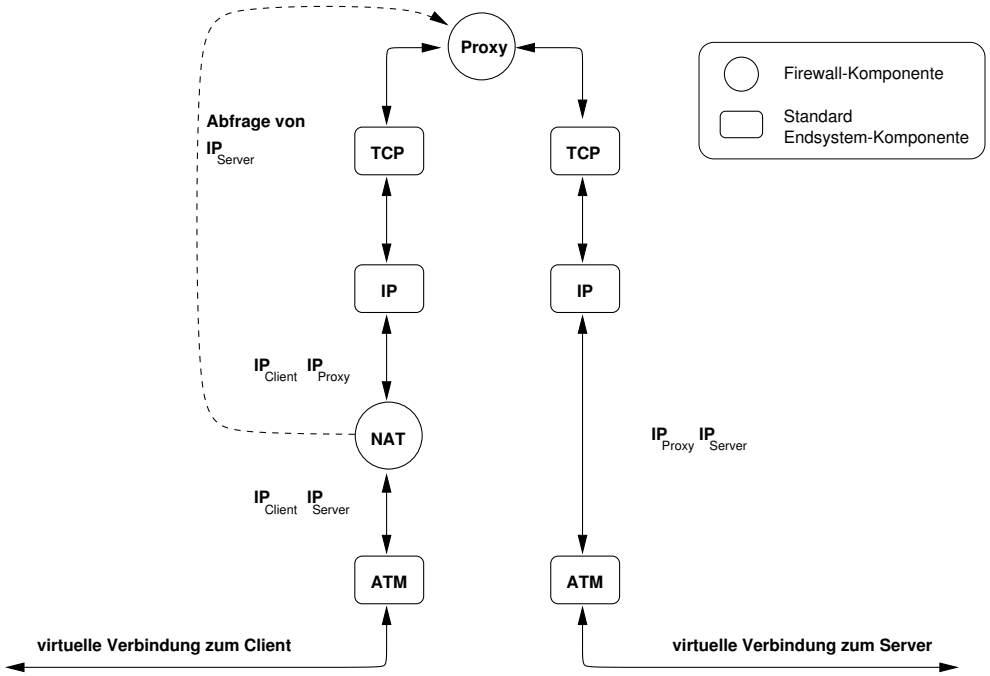


Abbildung 3.15: Datenfluß über einen IP-basierten Proxy in einem ATM-Netz

Damit die Kommunikation zwischen Client und Proxy zustande kommen kann, muß die zum Ersetzen der IP-Adressen in der NAT-Komponente angewendete Funktion bijektiv sein. Die eindeutige Abbildung von $< IP_{Client}, IP_{Server},$

$TCP_{Client}, TCP_{Server}$ > auf < $IP_{Client}, IP_{Proxy}, TCP_{Client}, TCP_{Server}$ > für die von einem Client hereinkommenden Dateneinheiten ist in jedem Fall gewährleistet, da eine beliebige IP-Adresse (IP_{Server}) durch eine einzige IP-Adresse (IP_{Proxy}) ersetzt werden muß. In der Gegenrichtung beim Ersetzen der IP-Adresse vor dem Heraussenden zum Client kann es potentiell mehrere Adressen geben (IP_{Server}), die für die IP-Adresse des Proxies (IP_{Proxy}) eingesetzt werden können. Dieser Fall tritt ein, wenn derselbe Client zu zwei verschiedenen Servern zwischen denselben Ports TCP-Verbindungen aufbaut.[70] Die Wahrscheinlichkeit für so eine "Kollision" ist nicht besonders hoch, da die TCP-Implementationen gebräuchlicher Systeme bei aufeinander folgenden Verbindungen die Portnummer des Clients inkrementieren. Trotzdem ist es nicht ausgeschlossen, daß eine "Kollision" auftritt oder sogar von dem Client herbeigeführt wird, um die Kommunikation der bereits bestehenden Verbindung zu stören. Eine "Kollision" kann von der NAT-Komponente erkannt werden, indem eine neu hinzukommende Abbildung mit bereits etablierten Abbildungen verglichen wird.

Durch zusätzliches Austauschen der IP-Adresse (IP_{Client}) oder der TCP-Portnummer des Clients (TCP_{Client}) kann die Kollision durch die NAT-Komponente vermieden werden. Das Austauschen der TCP-Portnummer ist dem Austauschen der IP-Adresse vorzuziehen, da die IP-Adresse von den meisten Proxies zur Identifikation des Clients benötigt wird. Der Austausch der TCP-Portnummer reicht aus, um über 60.000 Verbindungen vom selben Client auf denselben TCP-Port bei unterschiedlichen Servern zu ermöglichen. Der Austausch der Portnummer ist im Falle einer Kollision also ausreichend.[71]

Alternativ zum Einsatz der NAT-Komponente kann auch eine spezialisierte IP-Implementation eingesetzt werden, die in der Lage ist, IP-Dateneinheiten mit fremden IP-Empfänger-Adressen anzunehmen. Bei der spezialisierten IP-Implementation reicht es nicht allein aus, nur zusätzlich auch IP-Dateneinheiten mit fremden IP-Empfänger-Adressen akzeptieren zu können, sondern es müssen auch IP-Dateneinheiten mit der ursprünglichen Adresse als Absender-Adresse zurückgesendet werden können. Hierfür ist wie in der NAT-Komponente eine Speicherung der Adressen notwendig, für die IP-Dateneinheiten akzeptiert werden. Der Zugriff auf gespeicherte Adressen muß über die IP-Adresse des Clients und die bei-

[70] Dieser Fall tritt zum Beispiel ein, wenn vom Client `192.168.14.1` und Portnummer `1500` eine Verbindung zum Server `192.168.15.15` und Port `23` aufgebaut wird und derselbe Client gleichzeitig ebenfalls vom Port `1500` aus eine zweite Verbindung zu einem anderen Server `192.168.33.33` zum gleichen Port `23` aufbaut. Durch einfaches Ersetzen der IP-Adresse des Servers durch die IP-Adresse des Proxies könnten beide Verbindungen vom Proxy nicht mehr unterschieden werden.

[71] Beim Austausch der IP-Adresse des Clients wäre eine zusätzliche Abfrage analog zu dem Vorgehen bei der IP-Adresse des Servers auch zur Ermittlung der ursprünglichen IP-Adresse des Clients möglich. Da die IP-Adresse des Clients zusammen mit der IP-Adresse des Servers bei der NAT-Komponente mit einem einzigen Aufruf erfragt werden kann, ist auch bei dieser Lösung der Aufwand kaum höher.

den TCP-Portnummern erfolgen. Auch bei der spezialisierten IP-Implementation ist eine Schnittstelle notwendig, über die der Proxy die ursprünglich angegebene IP-Adresse erfragen kann. In der spezialisierten IP-Implementation müssen somit die gleichen Funktionen wie in der NAT-Komponente vorhanden sein; einzig das explizite Ersetzen der Adressen durch Manipulation am Datagramm kann entfallen, da diese Informationen innerhalb der IP-Implementation verwaltet werden können. Gegenüber der NAT-Lösung wird nur ein geringfügig niedrigerer Verarbeitungsaufwand erreicht, jedoch ist der Implementationsaufwand für die Integration der zusätzlichen Funktionen höher als bei der NAT-Lösung, die als eigenständiges Modul entwickelt werden kann. Von der erreichten Funktionalität sind beide Lösungen gleichwertig.

3.3.3.3.1 Performanz und Skalierbarkeit

Da zusätzlicher Verarbeitungsaufwand in der NAT-Komponente anfällt, liegt die erwartete Performanz dieser Lösung geringfügig unterhalb der Performanz eines herkömmlichen Proxies in einem CLIP-Netz (s. Abschnitt 4.4). Jedoch ist der Aufwand in der NAT-Komponente gering im Vergleich zum Verarbeitungsaufwand im Proxy, so daß mit keinem hohen Performanzverlust im Vergleich zu einem herkömmlichen Proxy gerechnet wird.

Im Vergleich zu herkömmlichen Proxies in CLIP-Netzen, bei denen der Proxy immer vom Client direkt adressiert werden muß, kann bei der hier beschriebenen Lösung die Entscheidung, ob ein Proxy notwendig ist, allein vom Signalisierungskontroller getroffen werden, da die Benutzung des Proxies für den Client vollkommen transparent erfolgt. Zusätzlich hat der Signalisierungskontroller dadurch auch die Möglichkeit, eine Lastverteilung zwischen mehreren parallelen Bastionen durchzuführen. Da die Verwendung des Proxies gegenüber dem Client transparent ist, kann der Signalisierungskontroller eine beliebige Bastion für die Bearbeitung der Verbindung eines Clients auswählen und ist nicht wie bei herkömmlichen IP-Proxies an einen bestimmten von Client adressierten Proxy gebunden. Der Signalisierungskontroller kann die Entscheidung, von welcher Bastion eine neue Verbindung bearbeitet werden soll, von der aktuellen Last jeder einzelnen Bastion, von der für die virtuelle Verbindung angeforderten Dienstgüte oder auch von den ATM-Adressen in der SETUP-Nachricht abhängig machen. Eine Unterstützung dieser Lastverteilung durch den Client ist wegen der Transparenz dieser Lösung nicht erforderlich. Der Eingriff des Signalisierungskontrollers erfolgt nur beim Aufbau der virtuellen Verbindung, so daß anders als bei einer Lastverteilung über eine zentrale NAT-Komponente (s. Abschnitt 5.2.2.3) während der anschließenden Datenübertragung keine Verarbeitung zur Lastverteilung mehr erforderlich ist und somit auch kein zentraler Performanzengpaß entsteht.

Da IP-Adressen oder Portnummern beim Aufbau der virtuellen Verbindung noch nicht vorliegen, kann der Signalisierungskontroller die Bastion nicht anhand der (anschließend) zugegriffenen Dienste auswählen. Da über die aufgebaute virtuelle Verbindung beliebige IP-basierte Dienste angesprochen werden können, müssen alle durch den Signalisierungskontroller auswählbaren Bastionen Proxies für die gleichen Dienste anbieten. Es kann also zum Beispiel nicht eine dedizierte Bastion für Telnet und eine dedizierte Bastion für FTP eingesetzt werden. Selbst wenn beim Aufbau einer virtuellen Verbindung bereits entschieden werden könnte, auf welchen IP-basierten Dienst zugegriffen werden soll,[72] können über eine bereits aufgebaute virtuelle Verbindung anschließend auch andere IP-basierte Dienste zugegriffen werden.

Das nach Aufbau der virtuellen Verbindung keine alternativen Bastionen mehr für die Kommunikation des Clients mit demselben Server ausgewählt werden können, hat neben den bereits beschriebenen Anforderung an die auf der Bastion verfügbaren Proxies auch eine Einschränkung der Leistungsfähigkeit der Lastverteilung zur Folge. Da die gesamte Kommunikation des Clients mit einem bestimmten Server immer über dieselbe virtuelle Verbindung übertragen wird, kann die Kontrolle der Kommunikation zwischen diesem Client und diesem Server nicht mehr auf mehrere Bastionen aufgeteilt werden. Die Einschränkung, daß die gesamte Kommunikation eines Clients mit demselben Server über dieselbe Bastion abgewickelt werden muß, ist aus Sicherheitsgesichtspunkten nicht als Nachteil zu sehen, da es Zusammenhänge zwischen unterschiedlichen Verbindungen gibt, die eine gemeinsame Kontrolle beider Verbindungen auf derselben Bastion erforderlich machen.[73] Aus Performanzgesichtspunkten kann es hingegen in Einzelfällen wünschenswert sein, die Kommunikation zwischen einem Client-Server-Paar von mehreren parallelen Bastionen überwachen zu lassen.

3.3.3.3.2 Prototyp

Zur Demonstration der Realisierbarkeit des vorgestellten Konzeptes wurde ein Prototyp eines transparenten NHRP-Proxies implementiert. Als NAT-Modul wurde die frei verfügbare Packet Screen "IP-Filter" eingesetzt, mit der ebenfalls die Performanzmessungen von Packet Screens in ATM-Netzen im Abschnitt 4.3 durchgeführt werden. "IP-Filter" verfügt bereits über Funktionen, um zusätzlich zur Filterung auch die IP-Adressen und gegebenenfalls Portnummern von IP-Datagrammen zu ändern. Durch die Angabe von NAT-Regeln ist es möglich,

[72] Durch eine Ergänzung der SETUP-Nachricht um ein zusätzliches IE, in dem die TCP-Portnummer des anschließend zugegriffenen Dienstes enthalten ist, ließe sich ein IP-basierter Dienst bereits bei der Signalisierung identifizieren. Dieser Ansatz ist für eine Zugriffskontrolle allein nicht ausreichend. (s. Seite 123f).

[73] Bei FTP existieren zum Beispiel Zusammenhänge zwischen den Datenverbindungen und der Kontrollverbindung.

IP-Datagramme, die an beliebige Zieladressen gerichtet sein können, auf dieselbe Zieladresse umzuleiten. Beispielsweise können Zugriffe auf den TCP-Port 8888 beliebiger Zielsysteme über die folgende Regel zum Port 8888 auf dem System mit der IP-Adresse `192.168.14.71` umgeleitet werden:

```
rdr ba0 0.0.0.0/0 port 8888 -> 192.168.14.71 port 8888 tcp
```

Wenn die Datagramme durch das NAT-Modul zur IP-Adresse der Bastion selbst umgeleitet werden, verarbeitet die IP-Implementation der Bastion die Dateneinheiten so als wären sie ursprünglich bereits an die IP-Adresse der Bastion gerichtet gewesen. Die für einen transparenten Proxy erforderliche NAT-Funktionalität kann somit bereits von der Software "IP-Filter" erbracht werden.

"IP-Filter" verfügt über ein Hilfsprogramm "`ipnat`", mit dem NAT-Regeln konfiguriert, aber auch Informationen aus dem NAT-Modul ausgelesen werden können. Über die Liste der aktiven Verbindungen kann dabei ermittelt werden, zwischen welchen Adressen das NAT-Modul momentan eine Abbildung durchführt. In der im folgenden wiedergegebenen Ausgabe von "`ipnat`" haben zwei Clients mit den IP-Adressen `192.168.14.73` und `192.168.14.77` mehrere Verbindungen zu den Zieladressen `192.168.15.75` und `192.168.15.76` aufgebaut, die durch das NAT-Modul auf die Adresse `192.168.14.71` umgeleitet werden.

```
mistral% /sbin/ipnat -l
List of active MAP/Redirect filters:
rdr ba0 0.0.0.0/0 port 8888 -> 192.168.14.71 port 8888 tcp

List of active sessions:
RDR 192.168.14.71   8888  <- -> 192.168.15.75   8888  [192.168.14.73 38072]
RDR 192.168.14.71   8888  <- -> 192.168.15.76   8888  [192.168.14.73 38070]
RDR 192.168.14.71   8888  <- -> 192.168.15.75   8888  [192.168.14.77 38310]
RDR 192.168.14.71   8888  <- -> 192.168.15.76   8888  [192.168.14.77 38309]
RDR 192.168.14.71   8888  <- -> 192.168.15.75   8888  [192.168.14.73 38069]
RDR 192.168.14.71   8888  <- -> 192.168.15.75   8888  [192.168.14.73 38068]
RDR 192.168.14.71   8888  <- -> 192.168.15.75   8888  [192.168.14.77 38308]
RDR 192.168.14.71   8888  <- -> 192.168.15.75   8888  [192.168.14.77 38307]
```

Mit dem Hilfsprogramm "`ipnat`" können somit die IP-Adressen ausgelesen werden, zu denen der Client ursprünglich eine Verbindung aufbauen wollte. Über die dem Proxy bekannte IP-Adresse des Clients und der vom Client benutzten Portnummer kann über die letzten beiden Spalten der Ausgabe von "`ipnat`" die Zeile selektiert werden, in deren mittlerer Spalte sich die ursprünglich vom Client angegebene IP-Adresse und Portnummer befindet. Der Client `192.168.14.73` mit der Portnummer `38072` (erste Zeile) hat somit ursprünglich eine Verbindung zur IP-Adresse `192.168.15.75` aufbauen wollen.

Mit dem NAT-Modul und dem Hilfsprogramm "`ipnat`" stehen bereits die wesentlichen Komponenten zur Verfügung, die für einen transparenten Proxy benötigt werden. Abschließend muß nur noch die Abfrage der IP-Adresse über "`ipnat`" in einem Proxy implementiert werden. Als Ausgangspunkt für den transparenten Proxy wurde "MGproxy" verwendet, der eigens als Experimentierplattform entwickelt wurde und auch für die Messungen der Performanz von Proxies im Abschnitt 4.4 eingesetzt wird. Um für den Aufruf von "`ipnat`" nicht über ein "`system()`" oder "`popen()`" aus dem Proxy heraus eine "Shell" starten zu müssen, wurde eine Funktion "`get_real_addr()`" implementiert, die aus "`ipnat`" abgeleitet dieselben Funktionen verwendet, um die im NAT-Modul gespeicherten Adressen auszulesen. Zusätzlich zum Aufruf der "`get_real_addr()`"-Funktion waren im Proxy keine wesentlichen Ergänzungen erforderlich.[74]

Dieser transparente Proxy kann wie bereits beschrieben über einen Signalisierungskontroller (s. Abschnitt 3.3.1.2) oder über NHRP (s. Abschnitt 3.3.1.3) eingebunden werden. Um den Prototyp eines transparenten Proxies auch ohne diese beiden Komponenten testen zu können, wird die in Abbildung 3.16 dargestellte Konfiguration aufgebaut. Die Clients werden so konfiguriert, daß sie die Bastion mit dem transparenten Proxy als Router verwenden. IP-Dateneinheiten für Systeme außerhalb des eigenen Subnetzes (`192.168.14.0`) werden dadurch an die Bastion gesendet. Auf der Bastion werden bei diesen Dateneinheiten durch das NAT-Modul die Empfänger-Adressen durch die IP-Adresse der Bastion (`192.168.14.71`) ersetzt, so daß der transparente Proxy die Dateneinheiten erhält. Die zweite Verbindung zu einem der Server kann vom Proxy dann direkt ohne Verwendung des NAT-Moduls aufgebaut werden. Bei diesem Aufbau konnten die Clients über den transparenten Proxy problemlos auf die beiden Server (`192.168.15.75` und `192.168.15.76`) zugreifen, bei denen jeweils am TCP-Port 8888 ein HTTP-Server installiert war.

In einem zweiten Experiment wird die zweite Schnittstelle der Bastion entfernt, so daß die Bastion nur noch in einem Subnetz (`192.168.14.0`) angeschlossen ist. Die Verknüpfung der beiden Subnetze erfolgt über einen Router (s. Abb. 3.17).

[74] Ein zusätzlicher Funktionsaufruf wurde benötigt, um die IP-Adresse und Portnummer des Clients zu ermitteln. Die Annahme von Verbindungen erfolgt bei "MGproxy" durch einen Hauptthread, der für die anschließende Bearbeitung einen Arbeitsthread auswählt, dem die neue Verbindung als "Socket" übergeben wird. Da die IP-Adresse und Portnummer des Clients zwar dem Hauptthread bekannt sind, aber nicht an den Arbeitsthread weitergegeben werden, muß der Arbeitsthread vor dem Aufruf der "`get_real_addr()`"-Funktion die IP-Adresse des Clients über den "Socket" durch den Aufruf der Standardfunktion "`getpeername()`" ermitteln. Alternativ zu diesem zusätzlichen "`getpeername()`"-Aufruf wäre es auch möglich gewesen, zusammen mit dem "Socket" auch die Adreßinformationen vom Hauptthread an den Arbeitsthread zu übergeben. Da hierfür jedoch umfangreichere Änderungen erforderlich gewesen wären, wurde mit dem "`getpeername()`"-Aufruf der geringere Implementationsaufwand gegenüber einer voraussichtlich besseren Performanz vorgezogen.

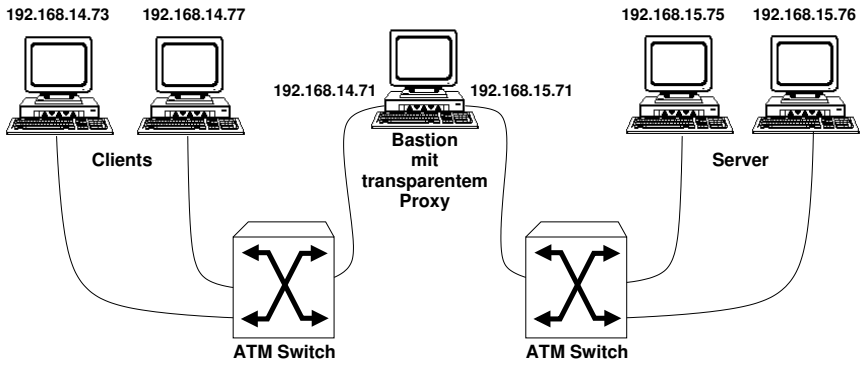


Abbildung 3.16: Testnetz-Konfiguration für erstes Experiment mit transparenten Proxies

Weiterhin verwenden die Clients die Bastion als Router, um Systeme außerhalb des eigenen Subnetzes anzusprechen. Die Bastion kann aber nicht mehr wie im ersten Experiment über eine zweite Schnittstelle direkt mit diesen Servern kommunizieren, sondern muß Dateneinheiten über dieselbe Schnittstelle an den Router weiterleiten, um Systeme in anderen Subnetzen zu erreichen. Auch dieses Experiment verlief erfolgreich. Die Clients konnten wie auch im ersten Experiment Verbindungen über den transparenten Proxy zu den Servern aufbauen, um dort einen HTTP-Server auf dem TCP-Port 8888 anzusprechen.

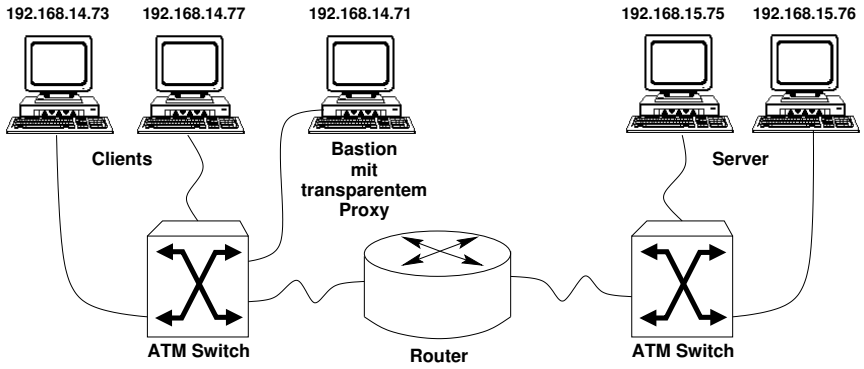


Abbildung 3.17: Testnetz-Konfiguration für zweites Experiment mit transparenten Proxies

3.3.3.3.3 Einsatz in herkömmlichen Netzen

Die hier beschriebene Lösung zur transparenten Integration eines Proxies über einen Signalisierungskontroller läßt sich nur unzureichend auf herkömmliche Netze übertragen. Transparente Proxies können in herkömmlichen Netzen nicht in gleicher Weise eingesetzt werden, weil die Wegwahl und die Datenübertragung in

herkömmlichen Netzen über die in jeder Dateneinheit enthaltenen Adreßinformationen eng miteinander gekoppelt sind.[75] Zwar können auch in herkömmlichen Netzen Dateneinheiten über eine in einem Router oder einer Packet Screen implementierten NAT-Funktion an eine Bastion gesendet werden, jedoch ist diese Lösung aus mehreren Gründen nicht so leistungsfähig, wie das Einbinden über einen Signalisierungskontroller in einem ATM-Netz.

- Die NAT-Funktion muß zentral in einem Router oder einer Packet Screen implementiert werden und kann nicht dezentral in jeder Bastion erfolgen. Bei einer hohen Last wird die zentrale NAT-Funktion zum Engpaß.
- Da in jeder Dateneinheit Adreßinformationen enthalten sind, muß die zentrale NAT-Funktion an der Verarbeitung jeder Dateneinheit beteiligt sein und kann nicht wie der Signalisierungskontroller einmalig beim Verbindungsaufbau eingreifen.
- Die Manipulation der Adreßinformationen in herkömmlichen Netzen macht enge Restriktionen bei der Netztopologie erforderlich, so daß die transparenten Proxies direkt an die zentrale NAT-Komponente angeschlossen werden müssen. Demgegenüber können die durch einen Signalisierungskontroller eingebundenen Sicherheitskomponenten an fast beliebiger Stelle des Netzwerkes angeschlossen werden.[76]
- Da die NAT-Funktion in herkömmlichen Netzen nicht erst in der Bastion selbst eingesetzt werden kann, ist die Adresse des ursprünglich angesprochenen Servers nicht auf der Bastion vorhanden. Die Übertragung der IP-Adresse erfordert bei herkömmlichen Netzen eine zusätzliche Kommunikation zwischen dem Proxy und der zentralen NAT-Komponente.

Fazit: Die Kombination einer NAT-Komponente mit einem Proxy ermöglicht das transparente Einbinden von Proxies über einen Signalisierungskontroller. Dieses neue Konzept wird durch die in einem ATM-Netz vollzogene Trennung der Wegwahl von der Datenübertragung ermöglicht. Diese hat zur Folge, daß das Einbinden eines Proxies beim Aufbau einer virtuellen Verbindung durch einen Signalisierungskontroller veranlaßt werden kann und bei der anschließenden Übertragung die Adreßinformationen der übertragenen Dateneinheiten nicht mehr zur

[75] Der Einsatz eines transparenten Proxies in herkömmlichen Netzen wird in [Chatel 96] beschrieben. Die Anwendbarkeit des dort beschriebenen Ansatzes ist jedoch stark eingeschränkt, da praktisch nur ein Einsatz auf einem Gateway-Firewall möglich ist. Bei Kombinationen von Packet Screen und Bastion können die IP-Dateneinheiten, die die IP-Adresse des Servers als Empfänger-Adresse enthalten, die Packet Screen nicht passieren und erreichen daher die Bastion nicht.

[76] Die Positionierung von Sicherheitskomponenten wird natürlich durch Sicherheitsanforderungen eingeschränkt.

Wegfindung berücksichtigt werden müssen. Durch die hier vorgestellte Lösung wird eine Transparenz gegenüber den Clients erreicht, die die bisher bei Zugriffen über Proxies notwendige Unterstützung durch den Client oder den Benutzer überflüssig macht. Ohne diese Transparenz wäre auch die beschriebene Lastverteilung durch den Signalisierungskontroller nicht möglich. Schließlich stellt auch die bei herkömmlichen Proxies schwierige Kaskadierung von Proxies durch die Transparenz dieser Lösung kein Problem mehr dar.

3.3.3.4 Fazit: Kontrollkomponenten auf Anwendungsebene

Der Einsatz von IP-basierten Proxies in ATM-Netzen stellt kein Problem dar, weil CLIP oder LANE die Eigenschaften des ATM-Netzes verbergen und die Firewall-Komponenten auf Anwendungsebene genauso eingesetzt werden können wie bei herkömmlichen IP-basierten Firewalls. Die Nutzung von ATM-spezifischen Eigenschaften ist bei dieser direkten Migration nicht möglich, so daß sich in erster Linie Performanzprobleme ergeben.

Für "Native"-ATM-Anwendungen sind bisher noch keine Proxies entwickelt worden. Die konzeptionellen Unterschiede zwischen ATM- und IP-Netzen wirken sich auf Proxies kaum aus, wie Untersuchungen der "Native"-ATM-Anwendungen Telefonie und "Video on Demand" zeigten. Insbesondere hat sich, anders als zunächst erwartet, die Überwachung von für Audio und Video eingesetzten virtuellen Verbindungen als nicht erforderlich herausgestellt, so daß vorerst noch keine Proxies benötigt werden, die eine garantierte Dienstgüte zur Verfügung stellen.

Die wesentliche Neuerung bei "Native"-ATM-Proxies ist, daß die Kontrolle transparent erfolgen kann. Weder Client noch Server müssen Kenntnis vom Proxy haben. Erst diese Transparenz ermöglicht das Einbinden eines Proxies über einen Signalisierungskontroller. Demgegenüber ist es bei IP-Proxies bisher immer notwendig gewesen, daß sowohl Client als auch Server jeweils direkt mit dem Proxy kommunizieren.

Aus der Kombination eines Proxies mit einer NAT-Komponente wurde das neue Konzept eines transparenten IP-Proxies für NHRP-Netze entwickelt. Erst mit diesen transparenten Proxies kann die Überwachung einer virtuellen Verbindung durch einen Signalisierungskontroller veranlaßt werden, ohne daß der Client den Proxy selbst ansprechen muß. Da der transparente Proxy zudem die Adresse des Servers über die NAT-Komponente ermitteln kann, muß der Client die Adresse des gewünschten Servers nicht wie bei herkömmlichen Proxies an den Proxy senden. Dieses neue Konzept wurde mit einem Prototyp erprobt.

3.3.4 Kontrolle des Netzwerkmanagements

Die für das Netzwerkmanagement in ATM-Netzen eingesetzten Protokolle sind ILMI, SNMP und PNNI.

ILMI: Der Anwendungsbereich des "Integrated Local Management Interfaces" (ILMI, s. Abschnitt 2.4.1) ist der Austausch von Managementinformationen zwischen Switch und angeschlossenen Endgeräten. Firewalls werden nicht zwischen Switch und Endgerät eingesetzt und können daher die mit ILMI verbundenen Sicherheitsprobleme nicht lösen.

SNMP: Das zum Management von Netzwerkkomponenten auch in ATM-Netzen eingesetzte "Simple Network Management Protocol" (SNMP) ermöglichte bisher nur eine schwache Authentisierung über Paßworte, die unverschlüsselt übertragen werden. Erst SNMPv2 (RFC 1901 – RFC 1910)[77] und das derzeit in der Entwicklung befindliche SNMPv3 (RFC 2272 – RFC 2275)[78] enthalten ausreichende Sicherheitsmechanismen. SNMP ist aus zwei Gründen für Firewall-Konzepte wichtig. Zum einen bedürfen SNMP-Nachrichten wegen der bekannten Schwachstellen dieses Protokolls einer sehr genauen Kontrolle durch einen Firewall. In der Regel lassen es Sicherheitsrichtlinien nicht zu, daß SNMP-Nachrichten durch einen Firewall hindurchgelassen werden. Auf der anderen Seite machen immer komplexere Firewall-Architekturen eine manuelle Konfiguration aller Komponenten sehr aufwendig. Dezentrale Firewall-Konzepte und der Aufbau virtueller Netze mit komplexen Filterkonfigurationen in allen ATM-Switches können ohne den Einsatz eines sicheren Netzwerkmanagementprotokolls nicht verwaltet werden. Nicht zuletzt ist es auch denkbar, Steuerinformationen zwischen Firewall-Komponenten über das Netzwerkmanagementprotokoll zu übertragen. Ein Proxy, der Angriffsversuche eines Systems erkannt hat, kann zum Beispiel über das Managementprotokoll den Signalisierungskontroller anweisen, alle virtuellen Verbindungen zu diesem System abzubrechen und keinen weiteren Verbindungsaufbau mehr zuzulassen. Ein Signalisierungskontroller könnte in der anderen Richtung zum Beispiel einer Packet Screen die Filterregeln mitteilen, die auf einer bestimmten virtuellen Verbindung angewendet werden sollen. Hier ergeben sich vielfältige Möglichkeiten, die jedoch nur mit einem sicheren Netzwerkmanagementprotokoll genutzt werden können.

---

[77] RFC 1901 [Case et al. 96a], RFC 1902 [Case et al. 96b], RFC 1903 [Case et al. 96c], RFC 1904 [Case et al. 96d], RFC 1905 [Case et al. 96e], RFC 1906 [Case et al. 96f], RFC 1907 [Case et al. 96g], RFC 1908 [Case et al. 96h], RFC 1909 [McCloghrie 96], RFC 1910 [Waters 96].

[78] RFC 2272 [Case et al. 98], RFC 2273 [Levi et al. 98], RFC 2274 [Blumenthal et al. 98], RFC 2275 [Wijnen et al. 98].

Die Kontrolle von SNMP durch einen Firewall ist kein ATM-spezifisches Problem, da SNMP ein Protokoll der TCP/IP-Protokollfamilie ist und bereits in herkömmlichen Netzen zur Verwaltung von Netzkomponenten eingesetzt wird. Da keine neuen Aspekte durch den Einsatz in einem ATM-Netz entstehen, wird von einer weiteren Diskussion abgesehen.

PNNI: Neben der durch den Signalisierungskontroller überwachten Signalisierung zum Aufbau virtueller Verbindungen ist auch der Austausch von "Routing"-Informationen eine weitere Aufgabe von PNNI. Da die "Routing"-Informationen für die Ermittlung der in DTLs angegebenen Wege verwendet werden, können manipulierte "Routing"-Informationen zum Umleiten des Verbindungsaufbaus über einen vom Angreifer kontrollierten Pfad oder auch für vielfältige "Denial of Service"-Angriffe ausgenutzt werden. PNNI-"Routing"-Informationen werden zwischen benachbarten Switches über das VPI/VCI-Paar <0,18> ausgetauscht. Die "Peer Group Leader" sind auf der höheren Hierarchiestufe in ähnlicher Weise verbunden, jedoch müssen hier SVCs zur Verbindung der nur logisch benachbarten Systeme verwendet werden. Ein Firewall kann diese SVCs kontrollieren, um Verbindungen von Systemen zu verhindern, die nicht zur "Peer-Group" gehören. Angreifer können so von Systemen außerhalb der "Peer-Group" keine gefälschten "Routing"-Informationen einspeisen. Diese Art der Absicherung von PNNI durch einen Signalisierungskontroller ist jedoch nur von sehr begrenztem Nutzen, da manipulierte "Routing"-Informationen bei jedem beliebigen Knoten einer "Peer-Group" eingespielt werden können und sich von dort aus auf die anderen Systeme der "Peer-Group" ungehindert ausbreiten können. Eine weitergehende Absicherung erfordert jedoch eine Ergänzung von PNNI um zusätzliche Mechanismen zur Authentisierung von übertragenen "Routing"-Informationen. Diese Absicherung ist unabhängig vom Einsatz eines Firewalls erforderlich und wird daher hier nicht weitergehend untersucht.

3.4 Firewall-Architekturen für ATM-Netze

Unter Berücksichtigung der im Abschnitt 3.2 untersuchten Schwachstellen von ATM-basierten Netzen werden im folgenden Abschnitt Firewall-Architekturen für ATM-Netze entwickelt. Für den Aufbau der Firewall-Architekturen werden die im Abschnitt 3.3 vorgestellten Firewall-Komponenten eingesetzt.

Zunächst wird im Abschnitt 3.4.1 gezeigt, wie eine Migration herkömmlicher Firewall-Architekturen durchgeführt werden kann und welchen Einschränkungen solche Lösungen unterliegen. Durch den Einsatz virtueller Netze wird eine flexible Basis für den Aufbau unterschiedlicher Firewall-Architekturen geschaffen (s. Abschnitt 3.4.2). Dieses Konzept erlaubt eine weitaus bessere Ausnutzung

von ATM-spezifischen Eigenschaften und kann auch als Basis für den Aufbau von Firewalls für "Native"-ATM-Anwendungen dienen. Hierbei wird sich zeigen, daß die bei herkömmlichen Firewalls oft eingesetzten Packet Screens in ATM-Netzen kaum noch benötigt werden (s. Abschnitt 3.4.3). Durch die Ausdehnung des Konzeptes der virtuellen Netze auf das gesamte interne Netz (Abschnitt 3.4.5) wird schließlich eine Aufteilung des internen Netzes nach Sicherheitsgesichtspunkten ermöglicht. Der Einsatzbereich von Firewalls verschiebt sich durch den Einsatz virtueller Netze von der ursprünglichen Nutzung am Übergang zwischen Internet und LAN. Zukünftig werden "virtuell zentrale Firewalls" die transparente Kontrolle beliebiger Kommunikationsbeziehungen nicht nur externer Systeme mit Systemen des internen Netzes, sondern auch zwischen Systemen des internen Netzes ermöglichen.

3.4.1 Firewall-Architekturen für CLIP-Netze

Durch den Einsatz von "Classical IP over ATM" (CLIP) nach [Laubach 94, Laubach et al. 98] können ATM-Netze zur Übertragung von IP-Dateneinheiten genutzt werden (s. Abschnitt 2.4.2). In CLIP-Netzen faßt man ATM lediglich als ein Übertragungsmedium (bzw. eine OSI-Schicht-2 Technologie) auf, über die IP-Dateneinheiten transparent übertragen werden, ohne daß ATM-spezifische Vorteile wie die Reservierungen von Betriebsmitteln ausgenutzt werden. Durch das Konzept des "logischen IP-Subnetzes" (LIS) wird von CLIP die Struktur herkömmlicher Netze auf ATM-Netze übertragen, indem eine direkte Kommunikation zwischen Systemen desselben LIS ermöglicht wird, während die Kommunikation zwischen unterschiedlichen Subnetzen einen Router erfordert. Da in einem CLIP-Netz die IP-Schicht und alle darüberliegenden Schichten unverändert erhalten bleiben, können im ersten Ansatz die herkömmlichen IP-basierte Firewall-Konzepte eingesetzt werden. So kann etwa ein Router, der zwei CLIP-Netze verbindet, wie in herkömmlichen Netzen als Packet Screen eingesetzt werden. Der Einsatz von ATM als Übertragungsmedium erfordert keine Veränderungen an den Filterfunktionen eines Routers, da die Filter auf IP-Dateneinheiten und nicht auf ATM-Zellen angewendet werden. Sowohl für das Weiterleiten im Router als auch für das Filtern müssen die IP-Dateneinheiten aus den einzelnen ATM-Zellen zusammengefügt werden (Reassemblierung). Nach der Filterung müssen die Dateneinheiten erneut auf Zellen aufgeteilt werden (Segmentierung) (s. Abschnitt 3.3.2.1).

Neben CLIP bieten auch "LAN Emulation" (LANE) die Möglichkeit, IP-Dateneinheiten über ATM-Netze zu übertragen, indem z. B. ein Ethernet vom ATM-Netz emuliert wird. Für die Kommunikation zwischen unterschiedlichen emulierten Ethernet-Segmenten muß bei LANE ein Router eingesetzt werden. Daher ist es auch bei LANE möglich, herkömmliche Firewall-Konzepte anzuwenden. Die

im folgenden für CLIP vorgestellten Netzwerktopologien können auch für LANE eingesetzt werden. Die Maßnahmen zur Absicherung der jeweiligen Netztopologien werden sich bei den für die jeweiligen Netze spezifischen Diensten — ATMARP bei CLIP sowie BUS, LECS und LES bei LANE[79] — im Detail unterscheiden. Die erzielbare Performanz hängt vom Protokoll-"Overhead" ab, der sich bei den beiden Lösungen deutlich unterscheidet. Die Untersuchungen beschränken sich im folgenden auf CLIP, da CLIP für den Einsatzbereich von Firewalls die größere Relevanz hat.

Im folgenden werden unterschiedliche Netzwerktopologien für die Integration von Firewalls in CLIP-Netze untersucht. Prinzipiell können alle im Abschnitt 2.3.2 vorgestellten Firewall-Architekturen auf CLIP-Netze übertragen werden. Da sich das Herangehen bei der Anpassung der jeweiligen Architektur an ein CLIP-Netz kaum voneinander unterscheidet, werden beispielhaft eine einfache Packet Screen (s. Abschnitt 2.3.2.1) und eine beidseitig abgeschirmte Bastion (s. Abschnitt 2.3.2.2) vorgestellt.

Im Gegensatz zu einem "Legacy"-LAN wie Ethernet, das lediglich ein Kabel für den Aufbau eines Netzes erforderte, benötigen ATM-Netze eine aktive Komponente, den Switch.[80] Für eine Netztopologie, bei dem ein Firewall eingesetzt wird, um zwei Netze — ein externes Netz und ein internes, zu schützendes Netz — voneinander zu trennen, werden daher zwei Switches benötigt. Für den Einsatz einer Packet Screen am Übergang zwischen einem internen Netz und einem externen Netz ergibt sich die in Abbildung 3.18 wiedergegebene Netztopologie.[81]

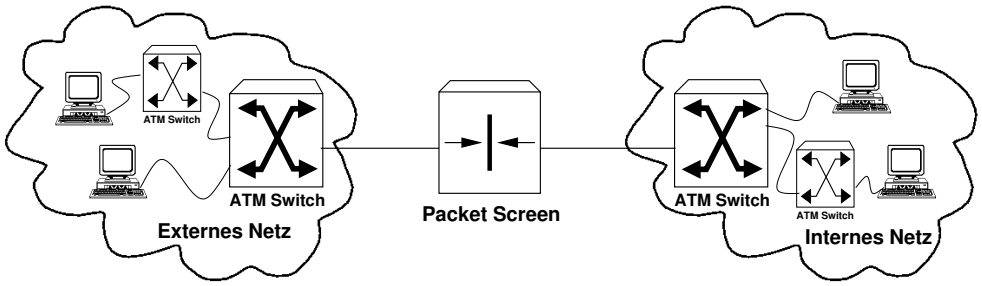


Abbildung 3.18: Packet Screen in einem CLIP-Netz

Vorteil dieser Lösung ist die klare Trennung der beiden Netze. Bei der in Abbildung 3.18 dargestellten Absicherung über eine Packet Screen ist zwischen beiden

[79] S. Abschnitt 2.4.6

[80] Switches werden heute auch bei Ethernet eingesetzt, um eine bessere Performanz zu erreichen. Im Gegensatz zur ATM kann der Aufbau eines Netzes bei Ethernet jedoch weiterhin ohne Switches erfolgen.

[81] Das interne und das externe Netz kann jeweils beliebig komplex sein. Dies soll in der Abbildung durch die kleineren Switches und Endsysteme angedeutet werden.

Netzen nur eine Kommunikation zwischen den in der Packet Screen freigeschalteten IP-Adressen und Ports möglich. Der Aufbau direkter virtueller Verbindungen zwischen internem und externem Netz ist nicht möglich. Angriffe auf die ATMARP-Server des jeweils anderen Netzes erfordern den Aufbau einer virtuellen Verbindung zum ATMARP-Server und sind somit bei dieser Firewall-Architektur nicht möglich. Die strikte Trennung der Netze hängt hier, wie auch bei Firewalls in "Legacy"-LANs, nur von der Sicherheit des Firewalls ab. Statt der Packet Screen kann in derselben Netztopologie auch ein Gateway-Firewall eingesetzt werden. Auf dem Firewall können dieselben Dienste wie bei Firewalls in herkömmlichen Netzen angeboten werden. Die Packet Screen bzw. der Gateway-Firewall ist bei der in Abbildung 3.18 dargestellten Firewall-Architektur allein für die Durchsetzung der Sicherheitsrichtlinien verantwortlich.

Der Nachteil dieser Lösung ist die ineffiziente Ausnutzung von teuren Hardware-Komponenten. Während der interne Switch zum Anschluß von Endgeräten oder weiteren Switches des internen Netzes genutzt werden kann, wird der externe Switch bei vielen Einsatzszenarien kaum ausgenutzt. Bei diesem Switch werden im Extremfall nur zwei Schnittstellen genutzt: Eine Schnittstelle zum Anschluß des Firewalls und die andere Schnittstelle zum Anschluß an ein Weitverkehrsnetz.

Die ineffiziente Ausnutzung der Switches wird noch deutlicher, wenn für die gewählte Firewall-Architektur zusätzlich ein weiteres IP-Netz benötigt wird, wie es bei einer beidseitig abgeschirmten Bastion (s. Abschnitt 2.3.2.2) der Fall ist. Für die Realisierung des in Abbildung 2.10 dargestellten Konzeptes einer Firewall-Architektur mit beidseitig abgeschirmter Bastion wird der Einsatz eines weiteren Switches erforderlich. Die Realisierung dieser Architektur ist in Abbildung 3.19 dargestellt. In dem zwischen den beiden Packet Screens gelegenen Grenznetz sind üblicherweise nur ein oder zwei Bastionen und gegebenenfalls ein Informationsserver (WWW) enthalten, so daß auch dieser Switch nur wenig ausgelastet ist. Da es sich bei Switches um vergleichsweise teure Komponenten handelt, wäre die Einsparung dieser ineffizient eingesetzten Switches wünschenswert, sofern dadurch die Sicherheit nicht reduziert wird.

3.4.2 Einsatz virtueller Netze für Firewall-Architekturen

Der Aufbau logisch voneinander getrennter Subnetze erfordert bei ATM nicht grundsätzlich auch getrennte Switches. Verschiedene Subnetze können auch auf demselben ATM-Switch konfiguriert werden. Bei den im vorigen Abschnitt beschriebenen Firewall-Architekturen dient der Einsatz getrennter Switches der Durchsetzung der Trennung von Netzen, die auf einem unterschiedlichen Sicherheitsniveau liegen. Ein Signalisierungsfilter ermöglicht eine logische Trennung innerhalb eines Switches anstelle der physikalischen Trennung beim Einsatz von

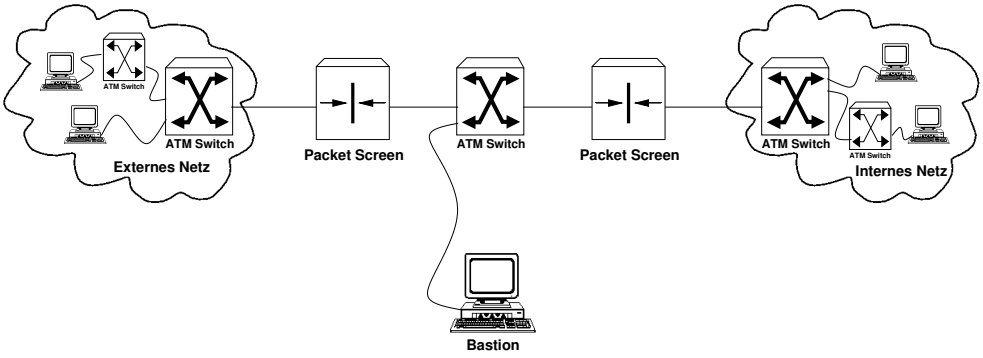


Abbildung 3.19: Beidseitig abgeschirmte Bastion für ein CLIP-Netz

zwei unabhängigen Switches. Die im vorigen Abschnitt vorgestellte Firewall-Architektur in Abbildung 3.18 vereinfacht sich dann durch die Realisierung des internen und des externen Netzes über denselben Switch zu der in Abbildung 3.20 dargestellten Architektur.[82]

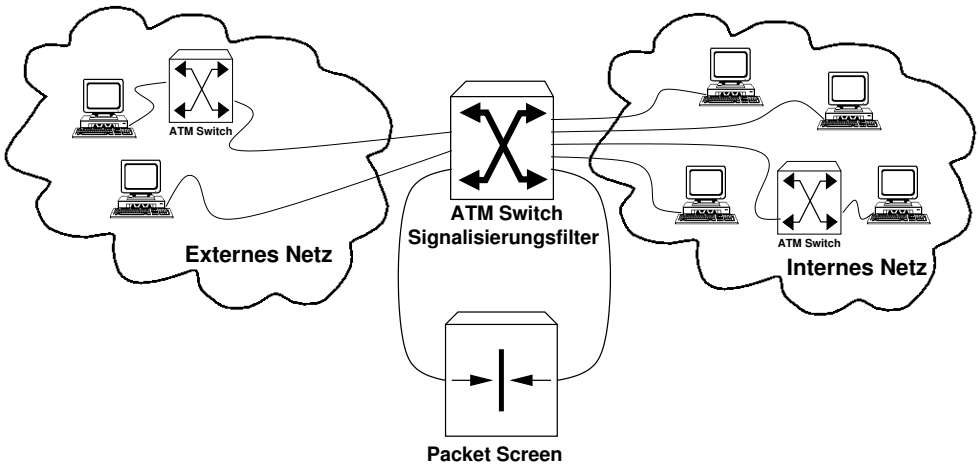


Abbildung 3.20: Kombination von Packet Screen mit Signalisierungsfilter

Für die Durchsetzung der Trennung der beiden Netze muß der Signalisierungsfilter sicherstellen, daß Signalisierungsnachrichten, die auf internen Schnittstellen des Switches eintreffen, nur für den Aufbau von virtuellen Verbindungen zu anderen internen Systemen benutzt werden können und ebenso externe Systeme nur virtuelle Verbindungen zu anderen externen Systemen aufbauen können. Erst bei der strikten Trennung der beiden Subnetze durch den Signalisierungsfilter

[82] Bei der in Abbildung 3.20 dargestellten Architektur kann statt der Packet Screen auch ein Gateway-Firewall eingesetzt werden.

kann von "virtuellen Netzen" gesprochen werden.[83] Da die beiden Schnittstellen, an denen die Packet Screen angeschlossen ist, ebenfalls zum internen bzw. externen Netz gehören, bildet die Packet Screen den einzigen Übergang, an dem IP-Dateneinheiten zwischen internem und externem Netz ausgetauscht werden können. Die Packet Screen ist bei der in Abbildung 3.20 dargestellten Architektur somit ein *logischer Engpaß*[84] zwischen beiden Netzen, während die Packet Screen beim Einsatz getrennter Switches (Abbildung 3.18) einen *physikalischen Engpaß* bildet.

Zusätzlich zum Sicherstellen der Trennung von internem und externem Netz durch einen Signalisierungsfilter ist auch die Sicherung der Konfiguration gegen Manipulationen erforderlich. Hierfür müssen unberechtigte Veränderungen der Filterregeln des Signalisierungsfilters und andere Manipulationen am Switch ausgeschlossen werden. Dies erfordert Einschränkungen der administrativen Zugänge zum Switch, die zum Beispiel über Telnet oder SNMP eine Schwachstelle des Switches bilden können.

Derzeit verfügbare Signalisierungsfilter ermöglichen noch nicht die Gruppierung von Schnittstellen zu geschlossenen Benutzergruppen, zwischen denen ausschließlich untereinander virtuelle Verbindungen aufgebaut werden können. Stattdessen müssen die Filterregeln auf der Basis von ATM-Adressen definiert werden. Dabei besteht, ähnlich wie bei einer Packet Screen, die Gefahr, daß der Signalisierungsfilter durch gefälschte ATM-Adressen ("Spoofing") umgangen wird. Für die Abwehr derartiger "Spoofing"-Angriffe müssen die Filterregeln so strukturiert werden, daß die Durchsetzung der Trennung der beiden Netze nur von Filterregeln auf ATM-Adressen abhängt, deren Korrektheit aufgrund der Netzstruktur ebenfalls überwacht werden kann. Bei den Schnittstellen zum internen Netz hin, können beim Signalisierungsfilter die ATM-Adressen der direkt angeschlossenen Systeme genau spezifiziert werden. Alle anderen Absender-Adressen dürfen auf diesen Schnittstellen nicht akzeptiert werden. Gleichzeitig dürfen Absender-Adressen interner Systeme nicht auf den externen Schnittstellen akzeptiert werden, da es sich hierbei nur um Angriffe externer Systeme handeln kann, die zur Umgehung von Filterregeln interne ATM-Adressen verwenden. Die Spezifikation von Filterregeln

[83] Die Bezeichnung "virtuelles Netz" oder "virtuelles LAN" wird häufig auch verwendet, wenn mehrere Netze auf demselben ATM-Netz realisiert werden, ohne daß damit in jedem Fall auch eine strikte Trennung dieser Netze verbunden ist. Die Konfiguration verschiedener "logischer Netze" auf demselben ATM-Netz ist mit relativ geringem Aufwand verbunden. Dagegen erfordert die Durchsetzung einer strikten Trennung erheblichen Aufwand zur Konfiguration der Filterregeln und zur Absicherung gegen Angriffe auf den Switch. In dieser Arbeit wird die Bezeichnung "virtuelles Netz" nur dann verwendet, wenn die Trennung der logischen Netze durch eine Zugriffskontrolle durchgesetzt wird. Bei der Realisierung mehrerer Netze auf demselben ATM-Netz *ohne* eine sichere Trennung wird zur Unterscheidung die Bezeichnung "logisches Netz" verwendet.

[84] Mit einem Firewall wird bewußt ein Engpaß geschaffen, um an dem Engpaß die Kommunikation überwachen zu können.

im Signalisierungsfilter kann ausschließlich durch die Angabe von Ausdrücken[85] auf ATM-Adressen interner Systeme erfolgen, indem auf internen Schnittstellen der Aufbau virtueller Verbindungen von internen Systemen zu anderen internen Systemen gestattet wird und alle anderen virtuellen Verbindungen pauschal verboten werden. Auf den externen Schnittstellen werden virtuelle Verbindungen von und zu internen Systemen verboten und alle anderen virtuellen Verbindungen pauschal erlaubt.

Die Konfiguration der Filterregeln eines Signalisierungsfilters für die Trennung virtueller Netze ist nicht trivial, da für jede Schnittstelle des Switches ein speziell angepaßter Satz von Filterregeln konfiguriert werden muß. Zur Vermeidung von Konfigurationsfehlern ist der Einsatz von Werkzeugen zur Erstellung von Filterregeln sehr empfehlenswert. Leider sind geeignete Werkzeuge noch nicht verfügbar.

Weiterhin ist zu beachten, daß für getrennte virtuelle Netze nicht derselbe ATM-ARP-Server genutzt werden darf. Der für die internen Systeme eingesetzte ATM-ARP-Server darf nicht für externe Systemen zugreifbar sein, um die in Abschnitt 3.2.2 beschriebenen Angriffe auf den ATMARP-Server zu verhindern. Deshalb sollte der ATMARP-Server vorzugsweise auf einem internen Switch oder Endsystem installiert werden. Falls dies nicht möglich ist, kann auch der ATMARP-Server auf dem Switch eingesetzt werden, wenn dort gleichzeitig der Signalisierungsfilter einen Zugriff externer Systeme auf den ATMARP-Server verhindert.

Durch den Einsatz des Signalisierungsfilters ist es nicht mehr erforderlich, getrennte Switches für das interne und das externe Netz einzusetzen. Hierdurch wird nicht nur die Einsparung eines Switches ermöglicht, sondern es entsteht auch eine sehr flexible Basis für den Aufbau komplexer Firewall-Architekturen. Abbildung 3.21 zeigt, wie eine beidseitig abgeschirmte Bastion mit nur einem Switch aufgebaut werden kann.

Die Flexibilität dieser Lösung zeigt sich besonders beim Ausfall einzelner Firewall-Komponenten. Durch Umkonfigurierung des Signalisierungsfilters kann problemlos ein logischer Engpaß durch einen anderen logischen Engpaß ersetzt werden. So kann der Betrieb aufrecht erhalten werden, ohne daß nach dem Ausfall einer Firewall-Komponente die Verkabelung geändert werden muß, bzw. die Komponente (physikalisch) ausgetauscht werden muß.

Ein weiterer Vorteil dieser Lösung ist, daß Systeme auch leicht zeitweise dem externen, ungeschützten Netz zugeordnet werden können, wenn deren Sicherheits- und Kommunikationsanforderungen dies erlauben bzw. erfordern. Das System braucht dafür nicht an einen anderen Standort verlegt werden. Auch die Verkabelung braucht nicht verändert zu werden. Ein dann externes System muß für

[85] Die Spezifikation von Filterregeln kann beispielsweise über reguläre Ausdrücke erfolgen.

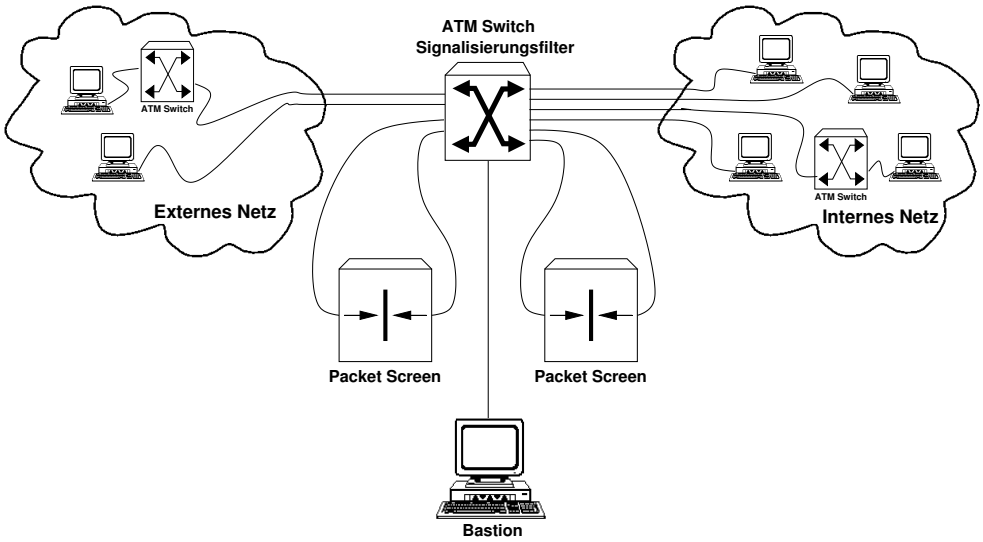


Abbildung 3.21: Beidseitig abgeschirmte Bastion mit Packet Screens und Signalisierungsfilter

den Zugriff auf interne Systeme den Firewall überqueren, auf der anderen Seite ist der Zugang zum Internet leichter. Dadurch kann einzelnen Systemen übergangsweise — etwa für spezielle Projekte — ein direkter, unkontrollierter Zugang zum Internet gewährt werden. Dies ist möglich ohne eine Veränderung in der Verkabelung. Bei klassischen Netzen wäre der Aufwand für die neue bzw. zusätzliche Verkabelung zu hoch, als daß sich dieser Ansatz in der Praxis durchsetzen ließe. Vorsicht ist jedoch bei regelmäßigem Wechsel der Sicherheitsdomänen[86] geboten, da sich daraus auch Sicherheitsprobleme ergeben können.[87]

Die Nutzung desselben Switches für verschiedene Netze ist aus Performanzgesichtspunkten kein Problem, da die Leistungsfähigkeit eines Switches so ausgelegt ist, daß gleichzeitig der maximale Durchsatz auf mehreren Anschlüssen des Switches übertragen werden kann, ohne daß die Datenübertragung auf einem An-

[86] Als (Sicherheits-)Domäne bezeichnet [DoD 85] eine Menge von Objekten, auf die ein Subjekt zugreifen kann. Wird ein Firewall zur Trennung von zwei Netzen eingesetzt, entstehen mit dem internen und dem externen Netz zwei Domänen.

[87] Für den Wechsel zwischen verschiedenen Domänen müssen in den Sicherheitsrichtlinien geeignete Maßnahmen wie Sicherheitsüberprüfungen oder Neuinstallation vorgesehen werden. Bei einem Wechsel vom externen zum internen Netz, also von einem niedrigen zu einem höheren Sicherheitsniveau, muß das Einbringen von "Trojanischen Pferden" und Viren verhindert werden. In der Gegenrichtung, beim Wechsel vom höheren zum niedrigeren Sicherheitsniveau, besteht die Gefahr, daß auf dem System gespeicherte vertrauliche Informationen das interne Netz verlassen. In der Regel wird der Aufwand für die Umkonfigurierung des Signalisierungsfilters weitaus geringer sein als der Aufwand für die Sicherheitsüberprüfungen oder Neuinstallation des Systems.

schluß die Performanz des anderen Anschlusses beeinträchtigt. Somit kann eine gegenseitige Beeinflussung der Performanz weitestgehend ausgeschlossen werden.

Alternativ zu einem Signalisierungsfilter kann die Trennung verschiedener virtueller Netze auch durch die Konfiguration von PVCs erreicht werden. Mit den PVCs werden die virtuellen Verbindungen zwischen allen Systemen von vornherein festgelegt. Der Aufbau von zusätzlichen SVCs durch die Signalisierung wird vollständig unterbunden, damit von einem Angreifer keine SVCs aufgebaut werden können, die gegen die Sicherheitsrichtlinien verstoßen. Da für jede Kommunikationsbeziehung ein PVC in den beteiligten Switches und Endsystemen konfiguriert werden muß, steigt der Konfigurationsaufwand mit dem Hinzufügen weiterer Kommunikationspartner erheblich. Wegen dieses Skalierungsproblems stellt die Konfiguration von PVCs nur bei sehr kleinen Netzen eine echte Alternative zu einem Signalisierungsfilter dar.

Fazit: Die Sicherheit des hier beschriebenen Ansatzes hängt von der strikten Trennung der beiden virtuellen Netze im Switch ab. Sollte es gelingen, direkt ohne Überquerung des Firewalls IP-Dateneinheiten von einem Netz in das andere zu senden, wäre die Sicherheit dieses Aufbaus nicht mehr gewährleistet. Da das externe Netz direkt an den Switch angeschlossen ist, besteht zudem das Risiko eines Angriffs auf den Switch. Daher muß bei dieser Topologie auch der Switch gegen Angriffe abgesichert werden. Der Switch muß daher mit in das Sicherheitskonzept des Firewalls integriert werden. Die Trennung der virtuellen Netze ließe sich sehr viel einfacher durchsetzen, wenn anstelle der Filterregeln auf ATM-Adressen eine Zuordnung der Schnittstellen des Switches zu geschlossenen Benutzergruppen möglich wäre. Insgesamt entsteht durch den Einsatz virtueller Netze jedoch eine sehr flexible Basis zum Aufbau unterschiedlicher Firewall-Architekturen.

3.4.3 Kombination von Bastion und Signalisierungskontroller

Bastionen werden in herkömmlichen Firewall-Architekturen mit einer oder mehreren Packet Screens kombiniert um sicherzustellen, daß eine Kommunikation zwischen internem und externem Netz ausschließlich über die Bastion stattfinden kann. Die Packet Screen kann zwar zusätzlich auch Zugriffe auf die Bastion einschränken, so daß nur ausgewählte Ports auf der Bastion erreichbar sind. Die entscheidende Aufgabe der Packet Screen ist hingegen, eine Umgehung der Bastion zu verhindern. In einem ATM-Netz ist eine Packet Screen nicht erforderlich, wenn stattdessen ein Signalisierungsfilter dafür sorgt, daß keine direkten virtuellen Verbindungen zwischen internem und externem Netz aufgebaut werden können und somit ausschließlich virtuelle Verbindungen zur Bastion für die Kommunikation zwischen internem und externem Netz verwendet werden können. Die hieraus entstehende Firewall-Architektur ist in Abbildung 3.22 wiedergegeben.

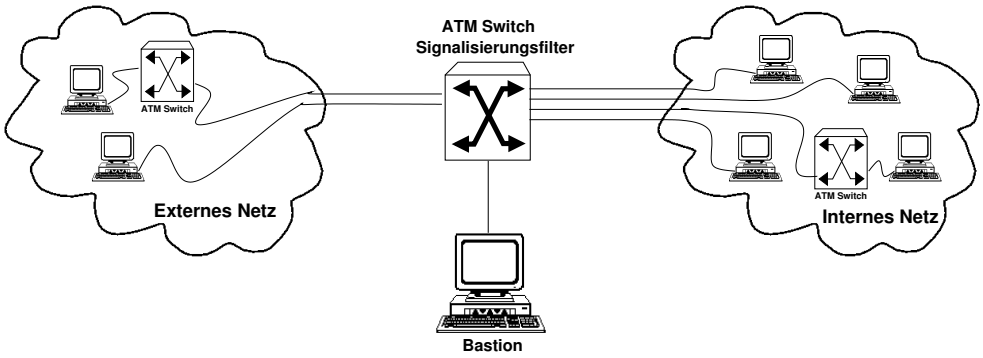


Abbildung 3.22: Kombination von Bastion und Signalisierungsfilter

Diese Firewall-Architektur eignet sich zur Absicherung von IP-basierten Diensten ebenso wie für "Native"-ATM-Dienste, da anders als bei der Kombination von Packet Screen und Bastion keine IP-spezifischen Firewall-Komponenten eingesetzt werden. Abhängig von den eingesetzten Proxy-Servern kann eine Bastion sowohl zur Absicherung von IP-basierten Diensten (s. Abschnitt 3.3.3.3) als auch zur Absicherung von "Native"-ATM-Diensten (s. Abschnitt 3.3.3.2) eingesetzt werden.

Eine noch leistungsfähigere Firewall-Architektur entsteht, wenn statt des Signalisierungsfilters ein Signalisierungskontroller eingesetzt wird. Durch den Signalisierungskontroller werden zwei wesentliche Ergänzungen erreicht. So wird durch das Umleiten von Verbindungen eine transparente Integration von Sicherheitskomponenten erzielt, da der Client beim Verbindungsaufbau den Proxy nicht mehr direkt adressieren muß. Neben dieser Transparenz ermöglicht der Signalisierungskontroller zusätzlich auch eine Lastverteilung zwischen replizierten Bastionen beim Verbindungsaufbau.

Während der Signalisierungsfilter nur überprüft, daß virtuelle Verbindungen nur zur Bastion hin eröffnet werden, kann der Signalisierungskontroller eine Verbindung beim Verbindungsaufbau umleiten, so daß die Bastion zur Kommunikation verwendet wird. Beim Einsatz eines Signalisierungskontrollers ist es daher nicht mehr erforderlich, daß der Initiator einer virtuellen Verbindung die Adresse der Bastion verwenden muß, damit die virtuelle Verbindung zur Bastion hin aufgebaut wird. Diese Transparenz erleichtert den Einsatz des Firewalls, da keine speziellen Konfigurationen der Systeme erforderlich sind, die über den Firewall kommunizieren müssen.

Eine Lastverteilung zwischen replizierten Bastionen kann ein Signalisierungskontroller durchführen, indem er beim Verbindungsaufbau eine Bastion auswählt, deren Ressourcen zur Bearbeitung der neuen virtuellen Verbindung ausreichen. Wenn vom Signalisierungskontroller eine Lastverteilung zwischen mehreren Ba-

stionen durchgeführt werden soll, ist es bei der Einrichtung von Bastionen nicht möglich, einen bestimmten IP-basierten Dienst nur auf einer Bastion anzubieten, während andere IP-basierte Dienste von anderen Bastionen angeboten werden, weil der Signalisierungskontroller die hierfür notwendige Auswahl einer Bastion nicht treffen kann. Da der Signalisierungskontroller nur die in der Signalisierungsnachricht enthaltenen Informationen zur Verfügung hat, können bei der Auswahl der Bastion noch keine TCP- oder UDP-Ports herangezogen werden, so daß Bastionen, die IP-basierte Dienste anbieten, immer für alle TCP- und UDP-basierten Dienste Proxies anbieten müssen. Eine Unterscheidung zwischen "Native"-ATM und IP-basierten Diensten kann hingegen vom Signalisierungskontroller durchgeführt werden, so daß für "Native"-ATM und IP-basierte Dienste getrennte Bastionen eingesetzt werden können.

Je nachdem ob ein Signalisierungsfilter oder ein Signalisierungskontroller eingesetzt werden soll, muß das "Routing" auf IP-Ebene unterschiedlich gelöst werden. Beim Einsatz eines Signalisierungsfilters braucht aus beiden Netzen nur die Bastion erreichbar zu sein. Bei NHRP muß der NHS somit nur die IP-Adresse der Bastion in die ATM-Adresse auflösen können, während der Zugriff auf Systeme im jeweils anderen Netz nicht erforderlich ist. Beim Einsatz eines Signalisierungskontrollers muß es hingegen möglich sein, die ATM-Adressen von Systemen im jeweils anderen Netz über NHRP zu ermitteln, damit anschließend eine Signalisierungsnachricht für den Aufbau einer direkten virtuellen Verbindung erzeugt werden kann. Für den Einsatz eines Signalisierungskontrollers für IP-basierte Kommunikation ist es daher erforderlich, daß IP-Adressen von Kommunikationspartnern im jeweils anderen Netz in die jeweiligen ATM-Adressen aufgelöst werden können. Der hierfür erforderliche Austausch von NHRP-Nachrichten über den Firewall hinweg ist wegen der sich hieraus ergebenden Angriffsmöglichkeiten nicht wünschenswert (s. Abschnitt 3.2.3). Eine Lösung, bei der es nicht erforderlich ist, NHRP-Nachrichten über den Firewall hinweg zu übertragen, ist die im Abschnitt 3.3.1.3 beschriebene Integration von Sicherheitskomponenten über NHRP. Dabei wird bereits auf die NHRP-Anfrage die ATM-Adresse der Bastion zurückgegeben, so daß der Client selbständig eine virtuelle Verbindung zur Bastion aufbaut und dieser Verbindungsaufbau somit nur über einen Signalisierungs*filter* kontrolliert werden braucht.

Fazit: Durch die Kombination einer Bastion mit einem Signalisierungsfilter ist der Einsatz von Packet Screens in ATM-Netzen nicht mehr erforderlich. Ein weiterer Vorteil dieser Lösung ist, daß sie auch für die Absicherung von "Native"-ATM-Anwendungen geeignet ist. Wird anstelle des Signalisierungsfilters ein Signalisierungskontroller eingesetzt, kann die Absicherung transparent erfolgen, und der Signalisierungskontroller kann zudem für eine Lastverteilung zwischen mehreren Bastionen sorgen.

3.4.4 Asymmetrische Kontrolle von Verbindungen

Wenn Firewall-Komponenten vom Signalisierungskontroller eingebunden werden, ist es möglich, abhängig von der Richtung des Verbindungsaufbaus unterschiedliche Firewall-Komponenten einzubinden. Bei von innen aufgebauten virtuellen Verbindungen können dann andere Firewall-Komponenten für die Kontrolle eingesetzt werden als bei von außen aufgebauten Verbindungen. Die Sicherheitsrichtlinien können zum Beispiel vorschreiben, daß virtuelle Verbindungen, die vom externen zum internen Netz hin eröffnet werden ("hereinkommende Verbindungen"), durch einen Proxy kontrolliert werden müssen, während herausgehende Verbindungen, also mit einem Verbindungsaufbau vom internen zum externen Netz, nur durch eine Packet Screen kontrolliert zu werden brauchen. Ein Signalisierungskontroller kann diese Sicherheitsrichtlinie einfach durchsetzen, indem er beim Aufbau der virtuellen Verbindung die jeweilige Firewall-Komponente in Abhängigkeit von der Richtung des Verbindungsaufbaus einbindet. Bei hereinkommenden Verbindungen wird dann durch den Proxy eine detaillierte Zugriffskontrolle der übertragenen Daten durchgeführt, während in der Gegenrichtung dem internen System soweit vertraut wird, daß die übertragenen Daten nicht durch einen Proxy kontrolliert zu werden brauchen. Bei herausgehenden Verbindungen stellt die eingebundene Packet Screen nur sicher, daß die virtuelle Verbindung nur für die nach den Sicherheitsrichtlinien vorgesehenen Protokolle (Filterung anhand von TCP-/UDP-Ports) verwendet werden und daß insbesondere über die Verbindung nicht gleichzeitig auch Zugriffe des externen Systems auf IP-basierte Dienste des internen Systems erfolgen.[88]

Das Problem beim asymmetrischen Einsatz unterschiedlicher Firewall-Komponenten ist, daß davon ausgegangen wird, daß Zugriffe grundsätzlich nur von dem System aus erfolgen, das die virtuelle Verbindung aufgebaut hat. Diese Festlegung der Richtung existiert bei virtuellen Verbindungen, die für eine IP-basierte Kommunikation aufgebaut werden, jedoch nicht, da IP die virtuelle Verbindung unabhängig von der Richtung des ursprünglichen Verbindungsaufbaus einfach als Transportmedium zur Übertragung von IP-Dateneinheiten verwendet. Daher ist das Ausschließen von Zugriffen in Gegenrichtung eine zu starke Einschränkung des vorgestellten Firewall-Konzeptes. Vielmehr werden es einige Sicherheitsrichtlinien erfordern, daß Zugriffe durch einen Proxy kontrolliert in das interne Netz hereingelassen werden, obwohl der Aufbau der virtuellen Verbindung von einem

[88] Das interne System könnte zum Beispiel die virtuelle Verbindung aufbauen, um auf einen (IP-basierten) Web-Server auf dem externen System zuzugreifen. Auf der bidirektionalen virtuellen Verbindung können vom externen System zusätzlich zu den Antworten des Web-Servers auch beliebige IP-basierte Dienste des internen Systems zugegriffen werden. Statt ein internes System direkt anzugreifen, könnte ein Angreifer auf diese Weise warten, bis eine virtuelle Verbindung durch das interne System aufgebaut wurde, über die die Angriffe erfolgen können. Die Gefährdung durch diese Zugriffe des externen Systems sollen durch die Packet Screen verhindert werden.

internen System initiiert wurde und daher nur eine Packet Screen vom Signalisierungskontroller eingebunden wurde. Dasselbe Problem besteht prinzipiell auch bei Verbindungen, die von außen aufgebaut wurden und bei denen daher ein Proxy zur Kontrolle verwendet wird. Bei Zugriffen des internen Systems auf IP-basierte Dienste des externen Systems wäre auch hier eine Kontrolle durch eine Packet Screen ausreichend, so daß auf die zusätzliche Belastung des Firewalls durch eine aufwendige Kontrolle in einem Proxy verzichtet werden könnte.

Folgende Lösungsansätze sind denkbar, um auch nach dem Aufbau einer virtuellen Verbindung noch andere Firewall-Komponenten einbinden zu können.

- Die bestehende virtuelle Verbindung wird abgebrochen. Nach dem Abbruch der virtuellen Verbindung kann beim erneuten Verbindungsaufbau eine andere Firewall-Komponenten eingebunden werden. Nachteil dieses Ansatzes sind die Verzögerungen, die durch den Verbindungsab- und -aufbau entstehen. Der Ansatz versagt vollständig, wenn die Richtung der Zugriffe sehr häufig wechselt oder wenn gleichzeitig Zugriffe in beiden Richtungen erfolgen, für deren Kontrolle unterschiedliche Firewall-Komponenten benötigt werden.

- Alternativ zum Abbrechen der virtuellen Verbindung könnte durch eine direkte Manipulation der virtuellen Verbindung durch den Signalisierungskontroller eine andere Firewall-Komponenten eingebunden werden. Dieser Ansatz ist effizienter als ein Abbruch der Verbindung, hat jedoch grundsätzlich dieselben Probleme bei häufig wechselnden oder gleichzeitigen Zugriffen.

- Gleichzeitige Zugriffe in beiden Richtungen erfordern eine getrennte Behandlung durch unterschiedliche Firewall-Komponenten. Die hierfür notwendige Klassifikation der Dateneinheiten kann die Packet Screen durchführen (s. Abschnitt 3.3.2.4). Dateneinheiten, die durch einen Proxy kontrolliert werden müssen bzw. die zu einer bereits durch einen Proxy kontrollierten TCP-Verbindung gehören, werden von der Packet Screen an den Proxy weitergeleitet. Alle anderen Dateneinheiten werden ausschließlich durch die Packet Screen kontrolliert.

  Für diesen Lösungsansatz können Packet Screen und Proxy entweder in demselben System kombiniert werden oder die Packet Screen muß Dateneinheiten, die durch einen Proxy kontrolliert werden sollen, über eine virtuelle Verbindung an eine Bastion weiterleiten, auf der ein Proxy installiert ist. Der kombinierte Einsatz von Packet Screen und Proxy auf demselben Endsystem stellt eine einfache Ergänzung des im Abschnitt 3.3.3.3 vorgestellten Proxies für NHRP-Netze dar. Selbst wenn Packet Screen und Proxy auf unterschiedlichen Systemen implementiert werden, unterscheidet sich

dieses Konzept kaum, da das Klassifizieren und Weiterleiten von Dateneinheiten unabhängig von der Realisierung in gemeinsamen oder getrennten Systemen nach dem gleichen Prinzip abläuft. Wichtig ist bei der Realisierung in getrennten Systemen lediglich, daß der Proxy auf der Bastion nur über die Packet Screen mit den Systemen im internen und im externen Netz kommuniziert. Eine direkte Kommunikation der Bastion mit anderen Systemen als der Packet Screen führt dazu, daß bei diesen Systemen anschließend zwei virtuelle Verbindungen (über die Packet Screen oder über die Bastion) existieren, über die IP-Dateneinheiten mit demselben System des anderen Netzes ausgetauscht werden können. Da IP-Dateneinheiten jedoch in beiden Übertragungsrichtungen grundsätzlich erst durch die Packet Screen klassifiziert werden müssen, ist es wichtig, daß neben der Verbindung über die Packet Screen keine direkte Kommunikation mit der Bastion stattfindet.

Fazit: Die neuen Firewall-Komponenten "Signalisierungskontroller" (s. Abschnitt 3.3.1.2) und der transparente Proxy für NHRP-Netze (s. Abschnitt 3.3.3.3) ermöglichen zusammen mit einer einfachen Erweiterung der Packet Screen eine neue Form der Zugriffskontrolle, die in herkömmlichen Netzen bisher nicht möglich war. Während der Signalisierungskontroller das Einfügen von Firewall-Komponenten in eine virtuelle Verbindung beim Verbindungsaufbau ermöglicht, kann eine Packet Screen durch die Klassifikation von Dateneinheiten auch noch für einzelne IP-Dateneinheiten innerhalb bereits bestehender virtueller Verbindungen zusätzliche Sicherheitskomponenten einbinden.

3.4.5 Zentrale Kontrolle und virtuelle Netze

Durch den Einsatz von Signalisierungsfiltern können mehrere virtuelle Netze auf demselben ATM-Netz eingerichtet werden. Dieses Konzept läßt sich nicht nur, wie im Abschnitt 3.4.2 beschrieben, für den Aufbau flexibler Firewall-Architekturen verwenden, sondern kann auch auf das gesamte interne Netz ausgedehnt werden. Durch den Einsatz von Signalisierungsfiltern in jedem Switch des internen Netzes wird es möglich, Systeme nach Sicherheitsanforderungen zu gruppieren und in jeweils eigenen virtuellen Netzen zusammenzuschließen.

Eine physikalische Netzstruktur muß sich an den baulichen Gegebenheiten und an dem Verkehrsaufkommen orientieren. Die gebräuchliche "strukturierte Verkabelung" ist mit jeweils eigenen Switches für jede Etage und gegebenenfalls einer weiteren Aufteilung innerhalb der Etage auf weitere Arbeitsgruppen-Switches sehr gut an die baulichen Gegebenheiten angepaßt (s. Abb 3.23). Die strukturierte Verkabelung berücksichtigt hingegen nicht, daß räumlich benachbarte Systeme

völlig unterschiedliche Sicherheitsanforderungen haben können. Einer Verkabelung nach Sicherheitsgesichtspunkten sind hingegen enge Grenzen gesetzt, da das Zusammenfassen räumlich getrennter Systeme in einem eigenen Netz einen sehr hohen Aufwand durch das Verlegen zusätzlicher Kabel erfordert. Der Einsatz von Signalisierungsfiltern ermöglicht hingegen den Aufbau virtueller Netze nach Sicherheitsgesichtspunkten, ohne daß hierfür die vorliegende Verkabelung geändert werden muß.

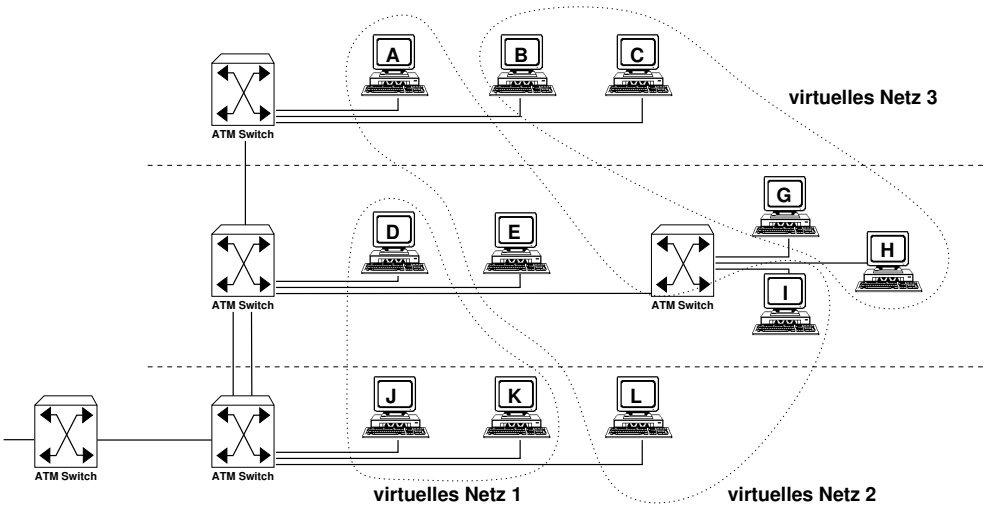


Abbildung 3.23: Aufteilung eines Netzes in virtuelle Netze

Abbildung 3.23 zeigt eine strukturierte Verkabelung mit jeweils eigenen Switches für räumlich getrennte Bereiche (z. B. Etagen).[89] Über Signalisierungsfilter können die Systeme A, E, I und L in einem virtuellen Netz verbunden und gegenüber allen anderen Systemen abgeschottet werden. Die anderen Systeme werden ebenfalls in virtuellen Netzen nach Sicherheitsgesichtspunkten zusammengeschlossen.

Über einen Signalisierungsfilter kann genau festgelegt werden, zwischen welchen ATM-Adressen virtuelle Verbindungen aufgebaut werden dürfen. Wenn Filterregeln auf allen Switches eingesetzt werden, können auf diese Weise die Kommunikationsmöglichkeiten aller Systeme eingeschränkt werden. In herkömmlichen Netzen ist über Filterregeln in Packet Screens nur der Verkehr kontrollierbar, der über die Packet Screen geführt wird. Lokaler Verkehr, der innerhalb eines der beiden an die Packet Screen angeschlossenen Netze bleibt, kann nicht kontrolliert werden. Im Gegensatz zu herkömmlichen Netzen kann in ATM-Netzen in diesem Sinne kein "lokaler Verkehr" vorkommen, da in einem ATM-Netz immer

[89] Eine räumliche Trennung wird in der Abbildung 3.23 durch eine horizontale, gestrichelte Linie symbolisiert.

mindestens ein ATM-Switch an der Kommunikation zwischen zwei Endsystemen beteiligt ist.[90] Daher kann über die Kontrolle der ATM-Switches exakt festgelegt werden, welche Systeme miteinander kommunizieren dürfen. Der Switch in einem ATM-Netz gleicht mit der Möglichkeit, jeden Zugriff eines Endsystems kontrollieren zu können, einem Kernel, der jeden Zugriff eines Prozesses auf Objekte innerhalb eines Systems kontrolliert.

Die Nutzung eines Switches zur Kontrolle der Endsysteme ist prinzipiell nicht nur bei ATM möglich. Auch bei Ethernet werden zunehmend "Broadcast"-Domänen durch Switches ersetzt, an deren Schnittstellen wie bei einem ATM-Netz ebenfalls nur jeweils ein Endgerät oder eine weitere Netzkomponente angeschlossen ist. Eine Kontrolle der Kommunikation ist auch hier durch eine Filterung in den Switches möglich. Da bei Ethernet jedoch nicht wie bei ATM eine Trennung von Datenübertragung und Wegfindung[91] erfolgt, müssen die Filterregeln für jede zu übertragende Dateneinheit ausgewertet werden. Die Performanz wäre bei einem Ethernet-basierten virtuellen Netz folglich viel schlechter als bei einem ATM-basierten virtuellen Netz, bei dem die Auswertung der Filterregeln nur einmalig beim Aufbau einer virtuellen Verbindung erfolgen muß, während die Datenübertragung keine weitere Kontrolle erfordert. Wegen der zu erwartenden geringen Performanz ist ein zukünftiger Einsatz von Filtern zum Aufbau virtueller Netze auf der Basis von Ethernet eher unwahrscheinlich. Beim Anschluß herkömmlicher Netze an ein ATM-Netz müssen daher alle durch ein "Edge-Device" oder einen Router vertretenen Systeme des herkömmlichen Netzes demselben virtuellen Netz zugeordnet werden. Sollen Endsysteme herkömmlicher Netze in Ethernet-Technologie unterschiedlichen virtuellen Netzen zugeordnet werden, muß beim Ethernet eine getrennte Verkabelung durchgeführt werden, und jedes Ethernet-Segment sollte über ein eigenes "Edge-Device" mit dem jeweiligen virtuellen Netz verbunden werden.[92]

Der Signalisierungsfilter kontrolliert die Kommunikation auf der Basis der ATM-Adressen, die beim Aufbau der virtuellen Verbindung angegeben werden. Prinzipiell besteht bei diesen Filtern eine Gefahr durch "Spoofing"-Angriffe. Wenn die Kontrolle des Signalisierungsfilters jedoch innerhalb jedes Switches erfolgt, kann der Versuch eines Endgerätes, die ATM-Absender-Adresse einer Signalisierungsnachricht zu fälschen, bereits von dem Switch erkannt werden, an den der Angreifer angeschlossen ist. Das Fälschen von Absender-Adressen kann bei dieser Netzstruktur daher leicht erkannt und abgewehrt werden. Es spielt da-

[90] Prinzipiell lassen sich auch zwei Endsysteme direkt miteinander über ATM verknüpfen. Jedoch ist dies nur in Ausnahmefällen sinnvoll, da dieses minimale ATM-Netz nicht mehr um weitere Stationen ergänzt werden kann.

[91] Die Wegfindung erfolgt bei ATM bereits durch die Signalisierung.

[92] Es ist denkbar, ein "Edge-Device" für unterschiedliche virtuelle Netze einzusetzen, wenn das "Edge-Device" eine *sichere* Zuordnung verschiedener angeschlossener Ethernet-Segmente zu unterschiedlichen virtuellen Netzen erlaubt.

her keine Rolle, daß die Zugriffskontrolle der Signalisierungsfilter auf fälschbaren Absender-Adressen basieren.[93]

Da die Kontrolle der Signalisierungsfilter auf ATM-Adressen basiert, ist das Konzept der virtuellen Netze gleichermaßen für alle ATM-basierten Protokolle einsetzbar. Natürlich können über "Edge-Devices" oder Router angeschlossene herkömmliche Netze nur einem virtuellen Netz zugeordnet werden, da eine Zuordnung von verschiedenen Endsystemen des herkömmlichen Netzes zu unterschiedlichen virtuellen Netzen eine Umgehung der Trennung ermöglichen würde.

Jedes virtuelle Netz ergibt ein "Cluster", bei dem jedes Endsystem eines "Clusters" Verbindungen zu jedem anderen Endsystem desselben "Clusters" aufbauen kann. In Einzelfällen kann es auch sinnvoll sein, statt eines "Clusters" nur sternförmige Verbindungen von n Clients zu einem Server zuzulassen. Direkte Verbindungen zwischen den Clients werden in diesem Fall unterbunden. Ein Bedarf für solche Konfigurationen besteht bei Netzen, bei denen Endsysteme anstelle von Terminals für den Zugriff auf einen zentralen Server verwendet werden.[94]

Die Konfiguration der Filterregeln für die Trennung der virtuellen Netze ist sehr aufwendig. Bereits die Konfiguration der in einem einzelnen Switch erforderlichen Filterregeln ist eine aufwendige und fehlerträchtige Aufgabe. Der Aufbau von virtuellen Netzen erfordert eine konsistente Konfiguration der Filterregeln in allen Switches des internen Netzes. Diese Aufgabe ist ohne Unterstützung durch ein automatisiertes Verfahren nicht zu bewältigen. Der Verwalter des Netzes sollte bei einem automatisierten Verfahren nur noch definieren müssen, welche Systeme in einem virtuellen Netz zusammengefaßt werden sollen. Die Erstellung der Filterregeln für jeden Switch bzw. für jede Schnittstelle jedes Switches muß dann zusammen mit der Konfiguration der Filterregeln auf den Switches von dem Netzwerkmanagementsystem automatisch durchgeführt werden.[95] Zu den Aufgaben des Netzwerkmanagementsystems sollte ferner auch die initiale Er-

[93] Auf jeder Schnittstelle des Switches, an die ein Endsystem angeschlossen ist, darf nur die Adresse des jeweiligen Endsystems verwendet werden. Bei der Verwendung einer anderen Absender-Adresse muß ein "Spoofing"-Angriff vorliegen, und die jeweilige Schnittstelle wird sofort vom Switch blockiert. Da die Kontrolle des Switches nur eine Verknüpfung von der ATM-Adresse mit der Schnittstelle des Endsystems durchführt, besteht natürlich weiterhin die Gefahr, daß ein Angreifer ein von ihm kontrolliertes Endgerät an die entsprechende Schnittstelle anschließt und mit der jeweils dort vom Switch erwarteten ATM-Adresse signalisiert. Dieses Emulieren des Endsystems durch den Angreifer kann vom Switch nicht erkannt werden. Die Sicherheit des virtuellen Netzes basiert somit auf der physikalischen Zugangssicherung zur Anschlußdose bzw. zum Kabel. Dieses Risiko besteht jedoch genauso bei getrennter Verkabelung.

[94] Für Netze, die Endsysteme wie "Network Computer" (NC) oder "Java Stations" einsetzen, wären diese sternförmigen Konfigurationen besonders interessant.

[95] Einen Ansatz, um auf der Basis einer einheitlichen Sicherheitsrichtlinie und mit Kenntnis der Netztopologie unterschiedliche Filterregeln für dezentrale Packet Screens und Proxies zu erstellen, wird in [Nessett et al. 98] beschrieben.

mittlung gehören, welche ATM-Adressen an welchen Schnittstellen angeschlossen sind. Eine Integration dieses Systems zur Konfiguration von virtuellen Netzen mit anderen Netzwerkmanagementaufgaben erscheint sinnvoll.

Verbinden virtueller Netze über Firewalls: Durch die Strukturierung nach Sicherheitsgesichtspunkten bleiben nur noch die Systeme desselben virtuellen Netzes untereinander erreichbar. Übergänge zwischen unterschiedlichen virtuellen Netzen können Systeme bilden, die gleichzeitig an verschiedene virtuelle Netze angeschlossen sind. Diese Systeme dürfen aufgrund der unterschiedlichen Sicherheitsanforderungen der angeschlossenen virtuellen Netze nur eine gesicherte Kommunikation zwischen Systemen unterschiedlicher virtueller Netze ermöglichen. Alle Systeme, die gleichzeitig an mehrere virtuelle Netze angeschlossen sind, müssen folglich Firewalls sein.

Durch diese Netzstruktur wird eine wichtige Schwäche herkömmlicher Firewall-Konzepte beseitigt. Herkömmliche Firewalls wurden üblicherweise am Übergang zwischen öffentlichen Netzen und einem internen Netz eingesetzt. Falls ein Angreifer diesen Firewall überwinden kann, steht dem Angreifer das gesamte interne Netz offen. Auch gegen sogenannte "Insider-Angriffe" von Benutzern anderer interner Systeme bietet ein Firewall am Übergang zum öffentlichen Netz keinen Schutz. Die implizite Annahme beim ausschließlichen Einsatz eines Firewalls am Übergang zwischen öffentlichen und internen Netz ist, daß alle Systeme des internen Netzes dieselben Sicherheitsanforderungen haben, was in der Praxis aber nicht immer zutrifft. Der Einsatz weiterer Firewalls zur Aufteilung des internen Netzes nach Sicherheitsgesichtspunkten war bisher nur zur Absicherung weniger, besonders sensitiver Abteilungen möglich, da der Aufwand für eine Verkabelung nach Sicherheitsgesichtspunkten in den meisten Fällen zu hoch ist. Durch virtuelle Netze entfällt dieser Aufwand, da die bestehende Verkabelung nicht geändert werden muß.

Dezentrale Firewalls: Am Übergang zwischen virtuellen Netzen kann entweder jeweils ein Firewall zwei virtuelle Netze verknüpfen oder ein zentraler Firewall sorgt für die Verknüpfung aller virtuellen Netze. Der Einsatz jeweils eines Firewalls zwischen zwei virtuellen Netzen (s. Abb. 3.24) unterscheidet sich konzeptionell zunächst nicht von den im Abschnitt 3.4.2 vorgestellten Firewall-Architekturen. Der hauptsächliche Unterschied entsteht dadurch, daß zum Erreichen eines Empfängers mitunter mehrere Firewalls zu überqueren sind, deren Kontrolle jeweils unterschiedliche Sicherheitsrichtlinien zugrunde liegen. Das Überqueren mehrerer Firewalls ist jedoch nur dann akzeptabel, wenn die Kontrolle des Firewalls transparent erfolgt, ohne daß jedesmal eine Interaktion mit dem Benutzer erforderlich wird. Durch die mehrfache Kontrolle der übertragenen Daten wird die Sicherheit nicht nennenswert erhöht, da ein Angriff auch allein von

der Komponente mit den stärksten Kontrollmechanismen erkannt und abgewehrt werden kann. Lediglich bei Schwachstellen eines Firewalls besteht die Chance, daß die anderen zu überquerenden Firewalls diese Lücke nicht enthalten und der Angreifer daher nicht in das gesamte Netz vordringen kann. Die Performanz leidet natürlich unter der wiederholten Kontrolle übertragener Daten bei jedem Firewall. Die Verwaltung dieses Netzes ist zudem sehr aufwendig, da zusätzlich zu den Signalisierungsfiltern auch noch eine Vielzahl von Firewalls konfiguriert und überwacht werden muß.

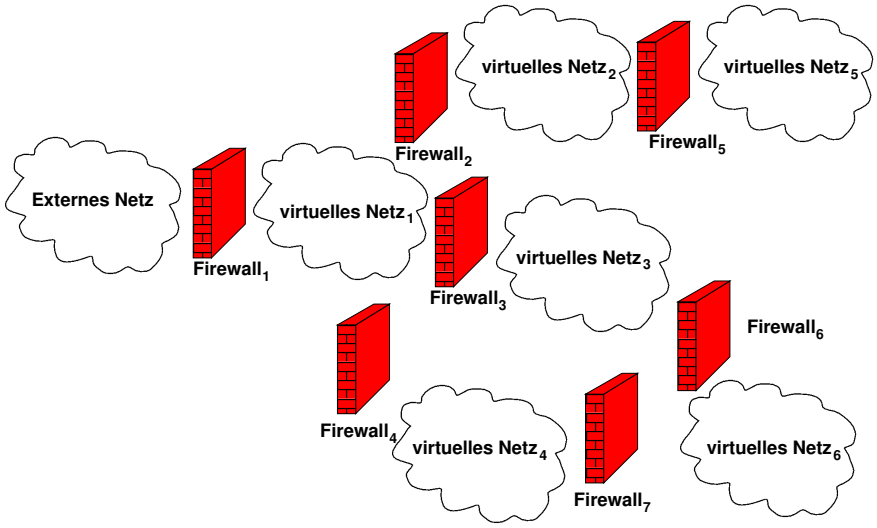


Abbildung 3.24: Kontrolle virtueller Netze durch dezentrale Firewalls

Zentraler Firewall: Ein zentraler Firewall, der die Kontrolle aller virtuellen Netze übernimmt (s. Abb. 3.25), erfordert hingegen nur die Konfiguration an zentraler Stelle. Jeder Empfänger kann entweder im selben virtuellen Netz ohne Kontrolle des Firewalls erreicht werden, oder es muß maximal ein Firewall — der zentrale Firewall — überquert werden. Das Überqueren mehrerer Firewalls ist bei dieser Lösung nicht erforderlich, so daß auch der Einsatz von Firewall-Komponenten akzeptabel ist, die nicht transparent sind. Trotz des Einsatzes eines einzigen zentralen Firewalls muß die vom Firewall durchgeführte Zugriffskontrolle nicht zwischen allen virtuellen Netzen gleich sein. Auch der zentrale Firewall kann die Kommunikation mit einem besonders sensitiven virtuellen Netz genauer kontrollieren als die Kommunikation zwischen anderen virtuellen Netzen. Während zwischen den virtuellen Netzen 1 und 2 (s. Abb. 3.25) zum Beispiel eine Absicherung auf Anwendungsebene über Proxies erforderlich sein kann, kann für die Kontrolle zwischen allen anderen virtuellen Netzen eine Packet Screen ausreichend sein.

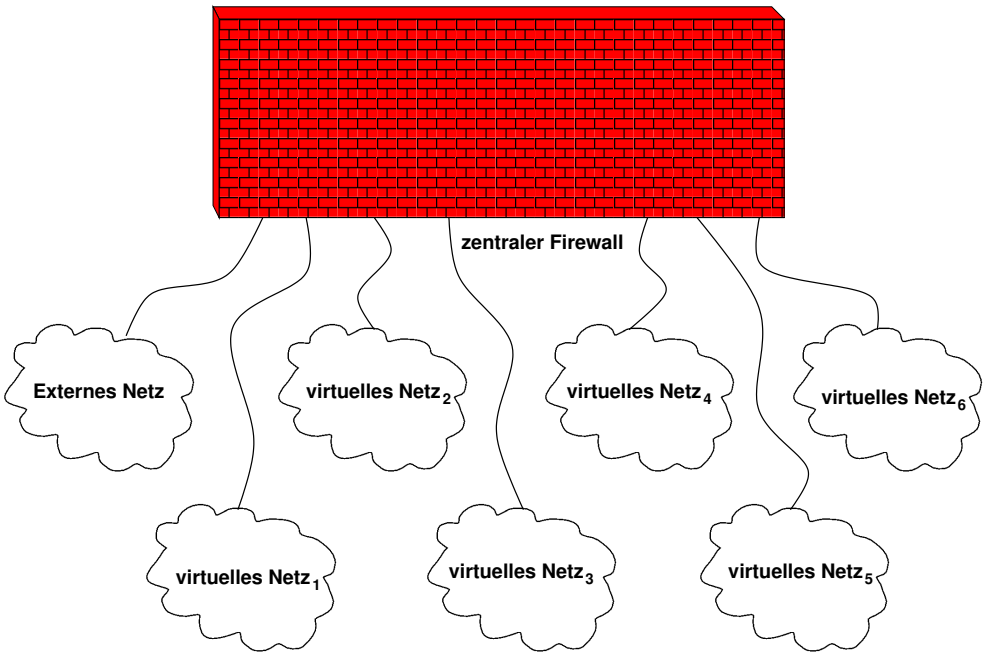


Abbildung 3.25: Kontrolle virtueller Netze durch einen zentralen Firewall

Nachteil des zentralen Firewalls ist, daß der zentrale Firewall eine zentrale Fehlerquelle darstellt. Durch einen Ausfall des zentralen Firewalls wird die Kommunikation zwischen allen virtuellen Netzen gestört. Noch entscheidender ist, daß ein erfolgreicher Angriff auf den zentralen Firewall alle virtuellen Netze und somit das gesamte interne Netz angreifbar macht. Dadurch sind die Systeme trotz der Aufteilung in virtuelle Netze genauso angreifbar wie in einem großen internen Netz, das durch einen einzelnen Firewall geschützt wird. Die Segmentierung des internen Netzes in einzelne virtuelle Netze bringt bei einem erfolgreichen Angriff keinen Vorteil mehr. Das Konzept mit multiplen Firewalls zwischen den virtuellen Netzen ist demgegenüber resistenter gegen Angriffe, da selbst nach dem erfolgreichen Überwinden eines Firewalls nur ein virtuelles Netz erreichbar ist, während die anderen virtuellen Netze durch weitere Firewalls geschützt bleiben.

Ein zentraler Firewall muß nicht zwangsläufig zu einem Performanzengpaß führen, da der zentrale Firewall seinerseits aus mehreren replizierten Komponenten bestehen kann, zwischen denen eine Lastverteilung erfolgt. Für die Einbindung dieser parallelen Firewall-Komponenten werden idealerweise Signalisierungskontroller (s. Abschnitt 3.3.1.2) eingesetzt. Beim Aufbau einer virtuellen Verbindung kann der Signalisierungskontroller dann eine Firewall-Komponente auswählen, die ausreichende Betriebsmittel zur Verfügung hat, um eine Kontrolle der virtuellen Verbindung durchzuführen.

Virtuell zentraler Firewall: Die Leistungsfähigkeit des Konzeptes virtueller Netze kann erst dann ausgenutzt werden, wenn auch der Firewall nur noch virtuell als zentraler Firewall existiert. Bei der Diskussion des zentralen Firewalls wurde bereits dargestellt, daß ein zentraler Firewall durchaus über mehrere replizierte Komponenten verfügen kann, zwischen denen eine Lastverteilung erfolgt. Diese Firewall-Komponenten (Proxies, Packet Screens, Netzwerkmonitore) brauchen nicht an zentraler Stelle plaziert zu werden, da die Signalisierungskontroller die Firewall-Komponenten an jeder beliebigen Stelle des Netzes einbinden können. Trotz der räumlichen Verteilung der Firewall-Komponenten wird wie bei einem zentralen Firewall eine einzige netzwerkweite Sicherheitsrichtlinie durchgesetzt. Dies steht im Gegensatz zu den bereits diskutierten dezentralen Firewalls, von denen jeder Firewall eine unterschiedliche Sicherheitsrichtlinie durchsetzt.[96]

Für die Endgeräte, deren Kommunikation kontrolliert wird, verhält sich der "virtuell zentrale Firewall" wie ein zentraler Firewall, da zum Erreichen eines anderen virtuellen Netzes nur einmal ein Firewall überquert werden muß und nicht wie bei dezentralen Firewalls mehrere Firewalls überquert werden müssen. Da zudem wie bei einem zentralen Firewall nur eine einheitliche Sicherheitsrichtlinie durchgesetzt wird, wurde die Bezeichnung "virtuell zentraler Firewall" für dieses Konzept gewählt.

Damit jede Komponente nach einer einheitlichen Sicherheitsrichtlinie eine Überprüfung durchführen kann, wird eine Möglichkeit benötigt, mit der zentral erstellte und verwaltete Sicherheitsrichtlinien auf die einzelnen Firewall-Komponenten verteilt werden können. Hierfür könnte eine Firewall-Komponente für jede neue virtuelle Verbindung, die kontrolliert werden soll, die für diese Verbindung zutreffenden Regeln bei einem zentralen Richtlinien-Server erfragen. Da beim Verbindungsaufbau nur die ATM-Adressen zur Auswahl einer Regel vorliegen, können später, wenn IP-Adressen oder Benutzernamen vorliegen, weitere Anfragen erforderlich werden. Da die Anfrage beantwortet werden muß, bevor die Datenübertragung über die Firewall-Komponente beginnen kann, entsteht neben den Auswirkungen auf die Performanz auch eine zentrale Fehlerquelle. Die lokale Speicherung der Richtlinien in allen Firewall-Komponenten ist daher vorzuziehen. Die Rolle des zentralen Richtlinien-Servers beschränkt sich dann auf die Verteilung der Richtlinien nach einer Änderung. Während des Betriebs müssen Firewall-Komponenten dann keine Rückfragen mehr beim Richtlinien-Server durchführen. Die Integrität und Authentizität der Sicherheitsrichtlinien muß gleichermaßen bei lokaler und zentraler Speicherung sowie bei der Übertragung gewährleistet werden.

Da die Firewall-Komponenten bereits beim Aufbau der virtuellen Verbindung den für diese virtuelle Verbindung relevanten Teil der Regeln auswählen können, sind

[96] Natürlich müssen die unterschiedlichen Sicherheitsrichtlinien des dezentralen Firewalls wiederum einer unternehmensweiten Sicherheitsrichtlinie genügen.

bei der anschließenden Kontrolle der übertragenen Daten jeweils viel weniger Regeln auszuwerten als bei einem herkömmlichen Firewall. Diese Vorauswahl der Regeln beim Verbindungsaufbau hat zwei wichtige Vorteile. Einerseits kann durch die geringere Anzahl von Regeln natürlich eine bessere Performanz erreicht werden. Auf der anderen Seite können unerwünschte Wechselwirkungen zwischen unterschiedlichen Regeln leichter vermieden werden.

Ein weiterer wichtiger Vorteil des "virtuellen zentralen Firewalls" ist, daß unnötige Umwege vermieden werden. Die Kommunikation zwischen zwei räumlich benachbarten Endsystemen muß nicht zum zentralen Firewall hin und zurück übertragen werden. Stattdessen kann der Signalisierungskontroller die virtuelle Verbindung über die nächstgelegene Firewall-Komponente führen, die eine nach den Sicherheitsrichtlinien erforderliche Kontrolle durchführt. Dadurch kann lokaler Netzwerkverkehr trotz der Kontrolle durch den Firewall weiterhin lokal bleiben.

Fazit: Durch den kombinierten Einsatz von virtuellen Netzen und Firewalls ergeben sich neuartige Möglichkeiten, durch die ein sehr viel höherer Schutz als beim Einsatz herkömmlicher Firewalls erreicht wird. Dezentrale Firewalls ermöglichen nicht nur die Aufteilung des internen Netzes nach Sicherheitsgesichtspunkten, sondern erreichen durch den Einsatz unabhängiger Firewalls auch einen Schutz in der "Tiefe", da das interne Netz nicht mehr ein homogenes Netz gleicher Sicherheitsstufe ist, sondern Angreifern das Vordringen durch weitere Firewalls erschwert wird. Zentrale Firewalls und "virtuell zentrale Firewalls" ermöglichen ebenfalls eine Aufteilung des internen Netzes nach Sicherheitsgesichtspunkten, bilden aber nur eine einzige Schwelle auf dem Weg in andere virtuelle Netze. Hierdurch wird einerseits eine bessere Performanz als mit dezentralen Firewalls erreicht, andererseits braucht ein Angreifer aber nur eine Schwachstelle in *einem* Firewall zu finden, um jedes System hinter dem Firewall erreichen zu können. Mit einem "virtuell zentralen Firewall" können Firewall-Komponenten sehr flexibel zur Kontrolle von virtuellen Verbindungen eingesetzt werden. Signalisierungskontroller ermöglichen zudem eine Lastverteilung zwischen Firewall-Komponenten. Alle in diesem Abschnitt vorgestellten Konzepte zur Absicherung von virtuellen Netzen durch Firewalls eignen sich sowohl zur Kontrolle von IP-basiertem Datenverkehr (CLIP, NHRP, LANE oder MPOA) als auch für "Native"-ATM-Anwendungen.

3.4.6 Einsatz virtueller Schnittstellen

Eine weitere Möglichkeit von ATM-Netzen ist der Einsatz virtueller Schnittstellen, durch die zwei IP-Schnittstellen auf derselben ATM-Schnittstelle verwendet werden können. Das ATM-Konzept der virtuellen Netze, die auf derselben physi-

kalischen Struktur realisiert werden, kann neben der schon beschriebenen Anwendung auf ATM-Switches auch auf Endgeräte ausgedehnt werden. Das zweite Netz wird dann als virtuelle IP-Schnittstelle auf derselben ATM-Schnittstelle konfiguriert. Eine Firewall-Architektur, die eine virtuelle IP-Schnittstelle einsetzt, wird in Abbildung 3.26 vorgestellt.

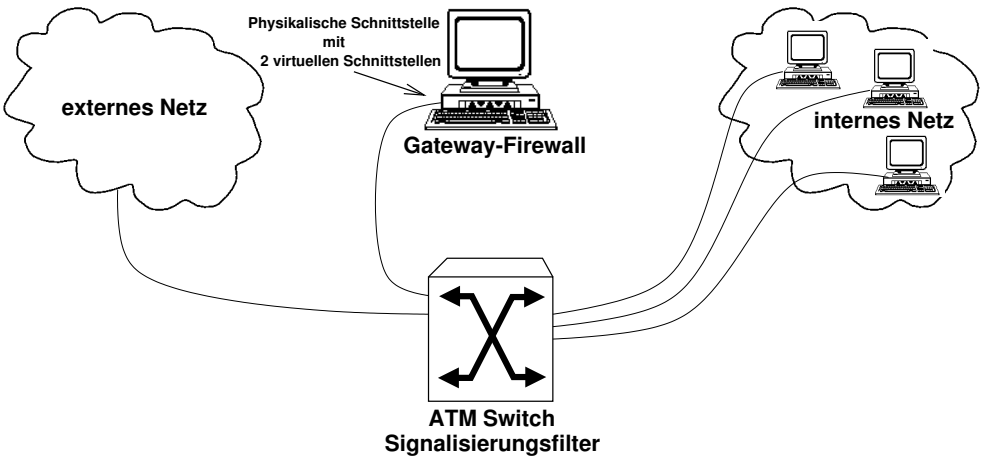


Abbildung 3.26: Gateway-Firewall mit einer virtuellen Schnittstelle

Durch diese weitere Vereinfachung entfallen die besonderen Hardware-Anforderungen (zweite ATM-Schnittstelle) an Packet Screens bzw. Gateway-Firewalls. Daher wird es möglich, eine einfache Workstation als Backup-System einzusetzen. Sofort einsichtig ist, daß sich in diesem Fall der maximale Durchsatz verringert, da sich beide Netze die Übertragungsrate derselben Schnittstelle teilen müssen. Diese Variante mit nur einer ATM-Schnittstelle erscheint daher besonders interessant für ATM-Netze, bei denen der Hauptanteil des Verkehrs lokal in den virtuellen Netzen bleibt und nur ein geringer Anteil zwischen den beiden Netzen über den Firewall geführt zu werden braucht. Ebenfalls sinnvoll ist der Einsatz nur einer Schnittstelle, wenn die Zugriffskontrolle einen Performanzengpaß bildet und aus diesem Grund der Durchsatz der Schnittstelle ohnehin nicht ausgenutzt werden kann. Da über eine ATM-Schnittstelle gleichzeitig gesendet und empfangen werden kann ("voll-duplex"), bleibt es trotz des Einsatzes nur einer ATM-Schnittstelle weiterhin möglich, unidirektionalen Datenverkehr mit der gesamten Sende- bzw. Empfangsleistung der Schnittstelle zu übertragen. Erst bei bidirektionalem Datenverkehr kommt es zu einer Halbierung des Gesamtdurchsatzes im Vergleich zu einer Packet Screen, die über zwei Schnittstellen verfügt. Nicht zuletzt wird durch dieses Konzept der virtuellen Schnittstelle die Skalierbarkeit verbessert, da zum Beispiel durch den Einsatz einer 622 MBit/s

ATM-Schnittstelle[97] eine Packet Screen mit einem maximalen Durchsatz von 300 MBit/s realisiert werden kann.

3.4.7 "Native"-ATM-Firewalls

Bei einer breiteren Verfügbarkeit von ATM ist zu erwarten, daß zusätzlich zum IP-Verkehr auch "Native"-ATM-Anwendungen genutzt werden. Unter "Native"-ATM-Anwendungen werden hier Anwendungen verstanden, die direkt auf ein ATM-Netz aufsetzen, ohne zusätzlich CLIP, NHRP, LANE oder MPOA zu verwenden, um Daten vor der Übertragung in Dateneinheiten herkömmlicher Netze wie z. B. IP oder Ethernet einzubetten. Bei "Native"-ATM können Firewall-Komponenten eingesetzt werden, um die Signalisierung zu überwachen (s. Abschnitt 3.3.1) oder durch Proxy-Server die übertragenen Daten zu kontrollieren (s. Abschnitt 3.3.3.2.2). Da bisher noch keine Proxies für "Native"-ATM-Anwendungen verfügbar sind, wäre es ein möglicher Ausweg, vorerst virtuelle Verbindungen für "Native"-ATM-Anwendungen am Firewall vorbei unkontrolliert in das interne Netz durchzuschalten. Der IP-Verkehr wird dann weiterhin vom Firewall kontrolliert, während die Daten für "Native"-ATM-Anwendungen unkontrolliert zwischen dem internen und externen Netz fließen können.

Dieser Lösungsansatz kann jedoch keine ausreichende Sicherheit bieten, da folgende Risiken bestehen:

- "Native"-ATM-Anwendungen sind noch neu und wenig ausgereift. Daher liegen über die Sicherheitsrisiken und Angriffsmöglichkeiten bei diesen Anwendungen noch keine Erfahrungen vor. Aufgrund bisheriger Erfahrungen mit TCP/IP-basierten Anwendungen muß jedoch bei neuen oder noch in der Erprobung befindlichen Anwendungen von vielen unentdeckten Lücken und Schwachstellen ausgegangen werden. Nach einem erfolgreichen Einbruch über eine "Native"-ATM-Anwendung hat ein Angreifer die Möglichkeit, von diesem internen System aus andere interne Systeme auch über IP anzugreifen.

- Es kann nicht sichergestellt werden, daß auf virtuellen Verbindungen, die für "Native"-ATM-Anwendungen eröffnet wurden, keine IP-Dateneinheiten übertragen werden. Da keine Kontrolle der virtuellen Verbindungen erfolgt, können Kommunikationspartner, nachdem die Verbindung eröffnet ist, beliebige Daten — somit auch IP-Dateneinheiten — übertragen. Diese Tunnel-Problematik besteht allerdings grundsätzlich bei jedem Protokoll,

[97] Der gemischte Einsatz von ATM-Schnittstellen mit unterschiedlichen Geschwindigkeiten stellt bei ATM kein Problem dar.

das durch einen Firewall durchgelassen wird. So ist zum Beispiel auch das Tunneln von IP-Dateneinheiten über eine TCP-Verbindung möglich.

Zur Lösung dieser Probleme gibt es grundsätzlich zwei Möglichkeiten: Entweder wird auch für "Native"-ATM ein vergleichbar sicherer Firewall eingesetzt oder die für "Native"-ATM-Anwendungen eingesetzten Systeme werden in einem virtuellen Netz zusammengefaßt, das vom externen Netz aus direkt erreichbar ist und dessen Systeme für Zugriffe auf Systeme im internen Netz – wie alle anderen externen Systeme – zuerst den Firewall überqueren müssen.

Abtrennen der "Native"-ATM-Anwendungen in einem virtuellen Netz: Die Konfiguration eines virtuellen Netzes ist für "Native"-ATM gleichermaßen möglich wie für IP-basierte Subnetze (s. Abschnitt 3.4.5). Die in Abbildung 3.27 dargestellte logische Struktur ergibt sich durch das Zusammenfassen aller direkt über "Native"-ATM erreichbaren Systeme in einem virtuellen Netz. Wie bei den im Abschnitt 3.4.5 beschriebenen virtuellen Netzen wird über Signalisierungsfilter in den ATM-Switches eine logische Trennung der virtuellen Netze durchgesetzt. Diese logische Netzstruktur spiegelt sich nicht in der physikalischen Netzstruktur wider, so daß räumlich benachbarte Systeme auch an unterschiedliche virtuelle Netze angeschlossen werden können.

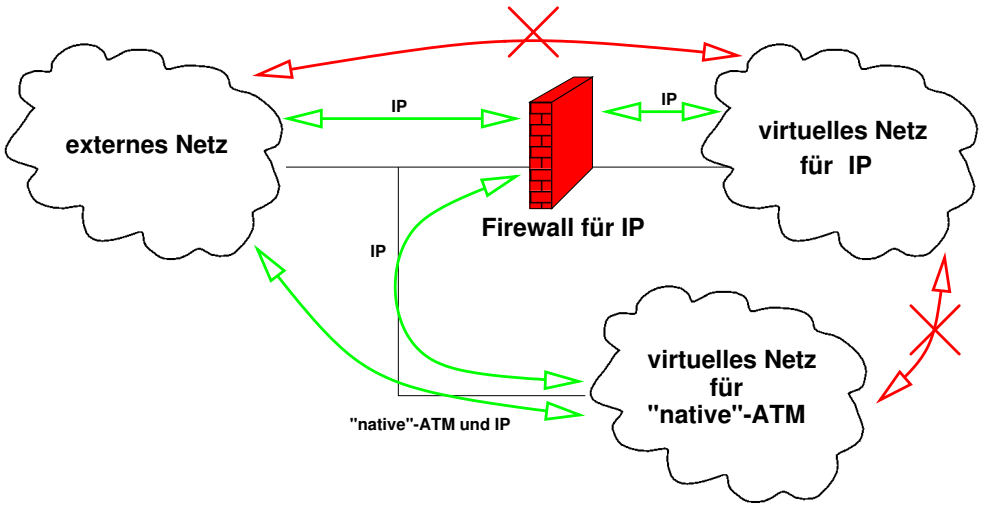


Abbildung 3.27: Trennung von "Native"-ATM und IP durch virtuelle Netze

Da die über "Native"-ATM erreichbaren Systeme nicht durch einen Firewall abgesichert werden, kann für diese Systeme nicht dasselbe Sicherheitsniveau erreicht werden wie für die Systeme in dem hinter dem Firewall gelegenen IP-Netz. Trotz der niedrigeren Sicherheit ist dieser Ansatz insbesondere in den ersten Phasen des

Einsatzes von "Native"-ATM-Anwendungen sinnvoll, in denen "Native"-ATM-Anwendungen in erster Linie noch experimentell eingesetzt werden und auf diesen Systemen auch noch keine Verarbeitung von kritischen Daten erfolgt. Der Zugriff von den "Native"-ATM-Systemen auf Daten im IP-Netz[98] ist stark eingeschränkt, da das "Native"-ATM-Netz vom Firewall wie jedes andere externe Netz behandelt wird. Aus diesem Grund ist die Integration von "Native"-ATM-Anwendungen in den normalen Betrieb mit diesem Ansatz kaum möglich. Spätestens sobald die "Native"-ATM-Anwendungen in den normalen Betrieb integriert werden sollen, muß daher auch für "Native"-ATM ein ebenso sicherer Firewall wie für herkömmliche IP-Dienste zur Verfügung stehen.

Einsatz eines "Native"-ATM-Firewalls: Die in Abschnitt 3.4.3 vorgestellte Kombination von Signalisierungsfilter/-kontroller und Bastion eignet sich sowohl für die Überwachung von IP-basierten Diensten als auch für "Native"-ATM, wenn auf der Bastion geeignete Proxies für die "Native"-ATM-Anwendung eingesetzt werden. Ein Proxy für die "Native"-ATM-Anwendung "Video on Demand" wurde im Abschnitt 3.3.3.2.2 skizziert.

Die Firewall-Architektur für einen integrierten "Native"-ATM- und IP-Firewall ist in Abbildung 3.28 wiedergegeben. Im Vergleich zur Abbildung 3.22 (S. 188) zeigt sich der konzeptionell geringe Mehraufwand gegenüber einem reinen IP-Firewall. Funktional spricht nichts dagegen, für "Native"-ATM und IP eine gemeinsame Bastion einzusetzen. Da die hohen Performanzanforderungen in Hochgeschwindigkeitsnetzen jedoch ohnehin eine Lastverteilung auf mehrere Bastionen erfordern, ergibt sich durch die Integration von "Native"-ATM- und IP-Proxies auf derselben Bastion kein Vorteil, so daß eine Trennung von IP und "Native"-ATM auf zwei Bastionen vorgezogen wird.

3.5 Zukünftige Firewall-Lösungen für ATM-Netze

In diesem Kapitel wurden Firewall-Konzepte für ATM-Netze vorgestellt. Der Einsatz von ATM zur Übertragung von IP-Verkehr ändert nichts an den Schwachstellen von IP und aller auf IP basierenden Protokolle. Vielmehr entstehen neue Schwachstellen durch die Dienste, die eingesetzt werden müssen, um die IP-Kommunikation über ATM-Netze zu ermöglichen. Die Schwachstellen von ATM-Netzen und Angriffe auf die zur Übertragung von IP benötigten Dienste wie ATMARP und NHRP wurden im Abschnitt 3.2 eingehend analysiert. Insgesamt sind ATM-Netze nicht sicherer als herkömmliche Netze, so daß auch für ATM-Netze Sicherheitsmechanismen wie Firewalls erforderlich sind.

[98] Endsysteme können prinzipiell gleichzeitig über "Native"-ATM und über IP kommunizieren.

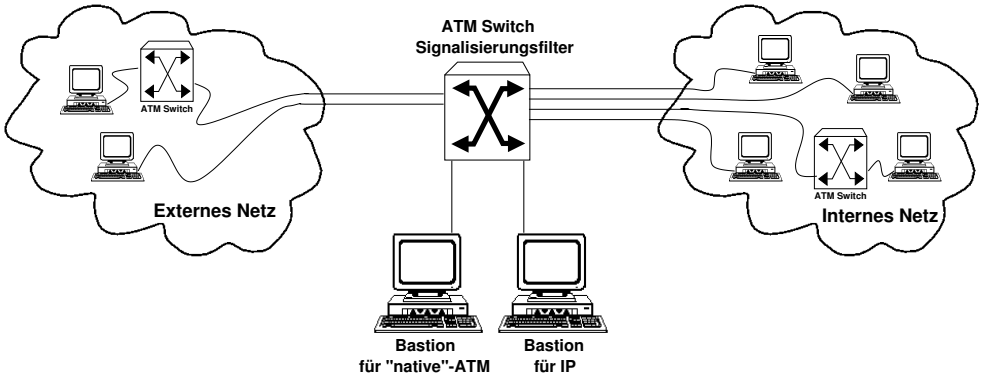


Abbildung 3.28: Ein Firewall für "Native"-ATM und IP

Die heute eingesetzten Firewalls wurden für herkömmliche Netze entwickelt und basieren daher teilweise auf Eigenschaften dieser Netze, die in ATM-Netzen nicht mehr vorhanden sind. So sind zum Beispiel in den übertragenen Dateneinheiten nicht mehr die gesamten Adreßinformationen enthalten und am Übergang zwischen zwei Netzen ist nicht mehr zwangsläufig auch ein Router vorhanden. Für den Einsatz von Firewalls in ATM-Netzen ist daher eine Anpassung der Firewall-Konzepte notwendig. Die in diesem Kapitel vorgestellten Lösungen reichen von einer einfachen Migration bekannter Firewall-Konzepte bis zu völlig neuen Firewall-Konzepten, durch die erst die Möglichkeiten von ATM-Netzen ausgenutzt werden. Zusammenfassend wird basierend auf den bisher vorgestellten Firewall-Komponenten und Firewall-Architekturen skizziert, wie Firewalls bereits heute in ATM-Netzen eingesetzt werden können und welche Entwicklungen in Zukunft zu erwarten sind. Dabei wird in drei Schritten vorgegangen:

1. Direkte Migration herkömmlicher Firewall-Konzepte.
2. In naher Zukunft realisierbare, neue Firewall-Konzepte für ATM-Netze.
3. In der ferneren Zukunft einsetzbare Firewall-Konzepte, die die Eigenschaften von ATM-Netzen optimal ausnutzen.

Direkte Migration herkömmlicher Konzepte: Im Abschnitt 3.4.1 wurde gezeigt, wie herkömmliche Firewall-Konzepte auch in ATM-Netzen weiterhin eingesetzt werden können. Bei dieser Migration sind primär die neuen Sicherheitsrisiken zu beachten, die durch ATM-spezifische Schwachstellen (s. Abschnitt 3.2.1) und durch den für IP benötigten Dienst (ATMARP, Abschnitt 3.2.2) entstehen. Wenn der Firewall wie gefordert nur eine IP-basierte Kommunikation ermöglichen soll, kann der Firewall die beiden ATM-Netze strikt voneinander trennen, so daß direkte Angriffe auf das durch den Firewall geschützte ATM-Netz und die

dort vorhandenen ATMARP-Server verhindert werden. Für die Kontrolle der IP-basierten Kommunikation können alle bisher bekannten Firewall-Architekturen mit den bisher verwendeten Firewall-Komponenten "Packet Screen" (Abschnitt 3.3.2) und "Proxy" (Abschnitt 3.3.3) eingesetzt werden. Auch bereits verfügbare Firewall-Produkte sind auf diese Weise in ATM-Netzen einsetzbar, sofern bei der Hardware die notwendigen ATM-Schnittstellen verfügbar sind bzw. diese von der Software unterstützt werden.

Ein Nachteil dieser Lösung, die ineffektive Nutzung teurer ATM-Switches, kann durch die Verwendung von Signalisierungsfiltern im ATM-Switch umgangen werden (s. Abschnitt 3.4.2). Der Signalisierungsfilter stellt eine logische Trennung des internen vom externen Netz sicher. Durch den Einsatz dieser virtuellen Netze entsteht eine sehr flexible Basis für den Aufbau herkömmlicher Firewall-Architekturen in ATM-Netzen.

Gemeinsamer Nachteil beider Konzepte bleibt jedoch, daß der Firewall nur zur Überwachung IP-basierter Dienste genutzt werden kann. Das ATM-Netz und die spezifischen Eigenschaften des ATM-Netzes werden durch diese direkte Migration herkömmlicher Firewall-Konzepte nicht ausgenutzt.

Neue Firewall-Konzepte für ATM-Netze: In herkömmlichen Netzen ist immer ein Router am Übergang zwischen zwei Netzen erforderlich. Herkömmliche Firewall-Architekturen nutzen diesen Router, um durch eine Packet Screen den Zugriff so einzuschränken, daß nur noch eine oder mehrere Bastionen von außen erreichbar sind. In ATM-Netzen ist am Übergang zwischen zwei Netzen lediglich ein ATM-Switch vorhanden. Router werden nur noch für die IP-basierte Kommunikation in einem ATM-Netz benötigt und müssen auch nicht mehr zwangsläufig direkt am Übergang zwischen zwei ATM-Netzen vorhanden sein. Für "Native"-ATM werden keine Router benötigt.

Der Einsatz von Packet Screens ist für Firewall-Architekturen in ATM-Netzen nicht erforderlich, da hier der Zugriff über einen Signalisierungsfilter eingeschränkt werden kann. Die Kombination von Signalisierungsfilter und Bastion (s. Abschnitt 3.4.3) ist ein neues Firewall-Konzept, daß sowohl für die Kontrolle von IP-basierter Kommunikation als auch für "Native"-ATM eingesetzt werden kann.

Signalisierungsfilter und Proxies für diverse IP-Dienste sind heute bereits verfügbar. Für "Native"-ATM-Anwendungen gibt es bisher noch keine Proxies. Jedoch bestehen konzeptionell keine Unterschiede zwischen Proxies für IP oder für "Native"-ATM, so daß bei einer größeren Verbreitung von "Native"-ATM-Anwendungen auch entsprechende Proxies verfügbar sein werden (s. Abschnitt 3.3.3.2.2).

Der Vorteil eines Signalisierungsfilters gegenüber einer Packet Screen ist, daß die Zugriffskontrolle nur einmalig beim Aufbau einer virtuellen Verbindung erfolgen muß und für die anschließende Datenübertragung kein "Overhead" durch Filterung einzelner Dateneinheiten entsteht. Die Performanzeigenschaften von Proxies unterscheiden sich hingegen nicht von Proxies in herkömmlichen Netzen. Durch die Kombination von einem Signalisierungsfilter mit Bastionen entsteht eine neue Firewall-Architektur, die sehr viel besser an ATM-spezifische Eigenschaften angepaßt ist, als es durch einfache Migration herkömmlicher Firewall-Konzepte möglich ist.

Zukünftige Firewall-Konzepte: Noch etwas weiter in der Zukunft liegt eine Ausdehnung des Konzeptes der virtuellen Netze auf ein gesamtes internes Netz (s. Abschnitt 3.4.5). Zwar ist der Aufbau virtueller Netze über die Konfiguration von Signalisierungsfiltern in ATM-Switches bereits heute schon möglich, aber die Komplexität dieser Konfiguration macht das manuelle Erstellen dieser Regeln praktisch unmöglich, da in jedem Switch für jede Schnittstelle unterschiedliche Signalisierungsfilter konfiguriert werden müssen. Erst mit Werkzeugen, die eine einfache Definition von virtuellen Netzen und eine automatisierte Konfiguration ermöglichen, können virtuelle Netze im gesamten internen Netz eingesetzt werden.

Erst durch den Einsatz von virtuellen Netzen wird es möglich, das interne Netz nach Sicherheitsgesichtspunkten zu strukturieren. In einem so strukturierten Netz brauchen Sicherheitskomponenten nicht mehr am physikalischen Übergang zwischen zwei Netzen eingesetzt werden, sondern können an fast beliebiger Stelle im Netz für eine Absicherung des Übergangs zwischen zwei virtuellen Netzen dienen. Der "virtuell zentrale Firewall" (s. Abschnitt 3.4.5) bietet hier die größten Freiheiten, um beliebige Sicherheitsrichtlinien am Übergang zwischen virtuellen Netzen durchzusetzen.

Damit die Sicherheitskomponenten nach den Anforderungen der Sicherheitsrichtlinie eingebunden werden können, ohne daß hierfür eine Unterstützung der Kommunikationspartner erforderlich ist, waren zwei entscheidende Innovationen erforderlich: Signalisierungskontroller und transparente Proxies für NHRP-Netze.

Für das Einbinden von Sicherheitskomponenten muß es möglich sein, beim Aufbau einer virtuellen Verbindung in Abhängigkeit von den Sicherheitsrichtlinien Verbindungen umzuleiten. Hierfür wurde im Abschnitt 3.3.1.2.2 das neue Konzept des Signalisierungskontrollers vorgestellt, der Signalisierungsnachrichten innerhalb der bestehenden PNNI-Spezifikation verändert, um Sicherheitskomponenten in eine virtuelle Verbindung einzubinden.

Das Umleiten von virtuellen Verbindungen kann nur dann angewendet werden, wenn die eingebundene Sicherheitskomponente transparent ist. Bisher ermögli-

chen nur Packet Screens eine transparente Kontrolle von übertragenen IP-Dateneinheiten. Proxies erfordern bisher hingegen immer die direkte Kommunikation mit dem Client, wofür der Client die Adresse des Proxies (IP-Adresse der Bastion und Portnummer des Proxies) benötigt. Damit Proxies durch Signalisierungskontroller in eine virtuelle Verbindung eingebunden werden können, wurde ein transparenter Proxy für NHRP-Netze (s. Abschnitt 3.3.3.3) entworfen, dessen Konzept auf der Integration von "Network Address Translation" mit Proxy-Servern basiert. Erst durch transparente Proxies wird es möglich, die Auswahl erforderlicher Sicherheitskomponenten allein durch den Signalisierungskontroller bestimmen zu lassen. Zudem ist es auch erst durch die Transparenz einfach möglich, mehrere Firewalls kaskadiert hintereinander zu betreiben. Das Überqueren mehrerer kaskadierter Firewalls ist erforderlich, wenn die Netze der Kommunikationspartner nicht direkt miteinander verbunden sind und die dazwischenliegenden Netze jeweils wieder über Firewalls miteinander verbunden sind.

Durch Signalisierungskontroller können Sicherheitskomponenten fast beliebig in Verbindungen eingebunden werden. Wenn zur Lastverteilung zwischen replizierten Sicherheitskomponenten ausgewählt werden kann, ermöglicht dies eine sehr gute Skalierbarkeit dieses Konzeptes. Außerdem wird eine höhere Verfügbarkeit erreicht, da anstelle einzelner, ausgefallener Sicherheitskomponenten problemlos andere, funktionstüchtige Sicherheitskomponenten durch den Signalisierungskontroller eingesetzt werden können. Zusätzlich zu den Sicherheitskomponenten können auch Switches und Leitungen redundant ausgelegt werden, so daß auch bei diesen Komponenten trotz einzelner Ausfälle der Betrieb des Netzes weiterhin möglich bleibt. Insbesondere können auch die Übergänge ins öffentliche Netz redundant ausgelegt werden.

Durch den Einsatz von transparenten Firewall-Komponenten in Verbindung mit Signalisierungskontrollern kann langfristig eine Strukturierung nach Sicherheitsgesichtspunkten erreicht werden. Das Firewall-Konzept des "virtuell zentralen Firewalls" ermöglicht eine sehr flexible Kontrolle nicht nur gegen Zugriffe von außen, sondern auch zwischen verschiedenen virtuellen Netzen innerhalb des internen Netzes. Gleichzeitig wird durch die lastabhängige Auswahl von Sicherheitskomponenten eine sehr gute Skalierbarkeit erreicht. Bis zur Einsatzfähigkeit dieser Lösung ist noch Entwicklungszeit notwendig, da die benötigten Komponenten noch nicht verfügbar sind.

4 Performanz von Firewalls

Inhalt

4.1 Performanzkriterien für Firewalls 212

4.2 Das Testnetz 215

4.2.1 Meßwerkzeug “Netperf” 217

4.2.2 Datendurchsatz 218

4.2.2.1 Auswahl der Meßpunkte: Nachrichtenlänge 219

4.2.2.2 Unterschied zwischen Nutzdatenlänge und Schreibzugriffslänge 220

4.2.2.3 Auswahl der Meßpunkte: Socket-Größe 223

4.2.2.4 Auswirkungen asymmetrischer Sende- und Empfangspuffergrößen 225

4.2.2.5 Berechnung des theoretisch erreichbaren Datendurchsatzes 226

4.2.2.6 Vergleich gemessener und theoretisch erreichbarer Datendurchsätze 236

4.2.2.7 Datagrammdurchsatz 237

4.2.2.8 Einsatz leistungsfähigerer Workstations 241

4.2.2.9 Schreibzugriffslängen oberhalb der MSS 242

4.2.2.10 “Localhost”-Messung des Datendurchsatzes 243

4.2.2.11 Datendurchsatz bei UDP 246

4.2.3 Nachrichtenlaufzeit 247

4.2.4 Verbindungsaufbauzeit 251

4.2.4.1 Verhältnis von Verbindungsauf- und -abbauzeit zur Transaktionsdauer 256

4.2.5 HTTP-Meßwerkzeug 258

4.2.6 Maximale Anzahl paralleler Verbindungen 263

4.2.7 Transaktionsdurchsatz bei parallelen Transaktionen 269

4.2.8 Transaktionsmessung mit längeren Transaktionen 277

4.2.9 Tuning von TCP-Parametern 281
4.3 Performanz von Packet Screens 287
4.3.1 Datendurchsatz über eine Packet Screen 290
4.3.2 Datagrammdurchsatz über eine Packet Screen 294
4.3.3 Messungen ohne Reduzierung der MTU 301
4.3.4 Nachrichtenlaufzeit über eine Packet Screen 303
4.3.5 Verbindungsaufbauzeit über eine Packet Screen 305
4.3.6 Abschließende Bewertung der Performanz von Packet Screens 307
4.4 Performanz von Proxy-Servern 312
4.4.1 Messungen mit "Socks" 314
4.4.1.1 Datendurchsatz über "Socks" 316
4.4.1.2 Nachrichtenlaufzeit über "Socks" 316
4.4.1.3 Verbindungsaufbauzeit über "Socks" 317
4.4.2 Messungen mit "MGproxy" 321
4.4.2.1 Datendurchsatz über "MGproxy" 322
4.4.2.2 Nachrichtenlaufzeit über "MGproxy" 329
4.4.2.3 Verbindungsaufbauzeit über "MGproxy" 330
4.4.2.4 Anwendungsperformanzmessungen mit "MGproxy" . 332
4.4.2.5 Maximale Anzahl paralleler Verbindungen 333
4.4.2.6 Transaktionsdurchsatz bei parallelen Transaktionen . 337
4.4.2.7 Transaktionsmessung mit längeren Transaktionen . . 343
4.4.2.8 Performanzauswirkungen von Zugriffskontrolle und Verschlüsselung 346
4.4.2.9 Performanzauswirkungen von kryptographischer Authentisierung 353
4.4.3 Abschließende Bewertung der Performanz von Proxy-Servern . 356
4.5 Performanz von Kombinationen aus Packet Screen und Bastion 362
4.5.1 Datendurchsatz über Packet Screen und Bastion 363
4.5.2 Nachrichtenlaufzeit über Packet Screen und Bastion 365
4.5.3 Verbindungsaufbauzeit über Packet Screen und Bastion 366
4.5.4 Abschließende Bewertung der Performanz von Kombinationen aus Packet Screen und Bastion 367
4.6 Abschließende Bewertung der Firewall-Performanz 368

Firewalls, die heute entweder mit einem Durchsatz von 2 MBit/s bei WAN-Anschlüssen oder in klassischen Ethernet-basierten LANs mit Übertragungsgeschwindigkeiten von 10 MBit/s arbeiten, erfüllen zur vollen Zufriedenheit die in sie gesteckten Ziele. Zum jetzigen Zeitpunkt verfügen Universitäten schon über Internet-Anschlüsse in 155 MBit/s-ATM-Technologie und für Anfang 2000 ist das Gigabit-WiN des DFN geplant. Innerhalb von Unternehmensnetzen in ATM-Technik oder mit Gigabit-Ethernet sollen Firewalls besonders sensitive Bereiche gegenüber anderen Abteilungen absichern, ohne dabei als Engpaß zu wirken. Für diese hohen Geschwindigkeiten sind die derzeit im Einsatz befindlichen Firewall-Architekturen nicht entwickelt worden.

In diesem Kapitel wird untersucht, welche Performanz[1] mit der heutigen Firewall-Technologie erreichbar ist und wo die Grenzen dieses Ansatzes liegen. Die hier angenommene Beschränkung auf herkömmliche IP-basierte Firewall-Architekturen in CLIP-Netzen, orientiert sich am allgemein üblichen Vorgehen bei der Migration von herkömmlichen "Legacy"-Netzen zu ATM-Netzen. Auch bei der Migration zu ATM-Netzen werden die Router-basierten Netzstrukturen zunächst beibehalten und erst in späteren Migrations-Phasen durch innovativere Topologien ersetzt. Um Firewalls für diese späteren Migrations-Phasen entwickeln zu können, ist zunächst die Identifizierung der Engpässe beim Einsatz heutiger Firewalls in CLIP-Netzen notwendig. Hieraus werden Vorschläge erarbeitet, welche Veränderungen notwendig sind, um künftig möglichst leicht skalierbare Firewalls in Hochgeschwindigkeitsnetzen einsetzen zu können. Ein weiterer Vorteil des Einsatzes von CLIP und der damit einhergehenden Router-basierten Netzstruktur ist die Möglichkeit, viele Meßergebnisse auch auf andere Hochgeschwindigkeitsnetze wie Fast- und Gigabit-Ethernet übertragen zu können. Trotz des dem CLIP-Netz zugrundeliegenden ATM-Netzes sind die durchgeführten Messungen in erster Linie IP-spezifisch und nicht ATM-spezifisch. Engpässe, die beim Einsatz eines Firewall-Konzeptes in einem ATM-Netz auftreten, sind auch beim Einsatz in anderen Hochgeschwindigkeitsnetzen zu erwarten.

Bisher gibt es nur wenige veröffentlichte Ansätze zur Performanzmessung von Firewalls. Im RFC 1944 [Bradner et al. 96], der das Vorgehen bei Performanzmessungen von Netzwerkkomponenten spezifiziert, wird im Abschnitt 11.4 ein Setup zur Messung bei Packet Screens definiert. Da der Schwerpunkt dieses RFCs auf der Messung von Netzkomponenten liegt, werden nur zwei Konfigurationen (mit einer und mit 25 Filterregeln) definiert. Ein Werkzeug zur Lastgenerierung auf Anwendungsebene wird in [Molitor] vorgestellt. Das Programm "Workload" erlaubt Konstrukte, mit denen parallele und sequentielle Aufträge

[1] Die Bezeichnung "Performanz" wird hier als Oberbegriff für die verschiedenen Leistungsaspekte, wie Datendurchsatz, Paketdurchsatz, Verzögerung und Transaktionsdurchsatz, verwendet.

erteilt werden können. Über Zufallsfunktionen können die Zwischenankunftszeiten der Aufträge variiert und dadurch eine möglichst realistische Last am Firewall erzeugt werden. Eine exakte Bestimmung der Leistungsfähigkeit eines Firewalls ist mit diesem Werkzeug schwierig, da aufgrund des "Overheads" des Werkzeuges keine ausreichende Anzahl von Aufträgen erzeugt werden kann, um an die Leistungsgrenzen hochperformanter Firewalls zu stoßen. Für die Lastgeneration von unterschiedlichen Quellen aus fehlt die notwendige Synchronisation. In [Kostick et al. 95] wird ein Szenario vorgestellt, bei dem Telnet-, FTP-, HTTP- und Email-Aufträge parallel von mehreren Workstations generiert werden, um am Firewall den Durchsatz zu messen. Neben diesen Ansätzen wurden von verschiedenen Computerzeitschriften Vergleichstests von Firewalls durchgeführt, die auch Aussagen zur Performanz enthalten. Die Genauigkeit dieser Messungen ist in der Regel nicht besonders hoch. Daher werden über die Qualität der Meßergebnisse (Standardabweichungen, Vertrauensintervalle) in der Regel keine Aussagen gemacht.

Die Untersuchungen in diesem Abschnitt sollen die Leistungsgrenzen heutiger Firewall-Techniken aufzeigen. Hierzu werden Leistungsmessungen an Firewalls in einem 155 MBit/s ATM-Netzes durchgeführt. Ziel ist es nicht, die Performanz eines Firewalls bei einer simulierten, typischen Benutzerlast zu messen, vielmehr soll die Leistungsfähigkeit durch Lastmessungen von vorab identifizierten Performanzkriterien erfolgen. Zunächst werden im Abschnitt 4.1 die unterschiedlichen Kriterien der Performanz von Firewalls kurz vorgestellt. Anschließend wird im Abschnitt 4.2 die Performanz der Systeme des Testnetzes gemessen, um für Ergebnisse späterer Messungen mit Firewalls die notwendigen Eckdaten als Vergleichswerte vorliegen zu haben. Dabei werden gleichzeitig die für die Untersuchung der verschiedenen Performanzkriterien notwendigen Messungen vorgestellt. In den folgenden beiden Abschnitten wird die Performanz der beiden Firewall-Komponenten Packet Screen (Abschnitt 4.3) und Proxy-Server (Abschnitt 4.4) analysiert. Die Performanzeigenschaften bei der Kombination verschiedener Firewall-Komponenten zu komplexeren Firewall-Architekturen werden im Abschnitt 4.5 analysiert. Das Kapitel endet mit einer zusammenfassenden Bewertung der Performanz von Firewalls. Im darauffolgenden Kapitel 5 werden Ansätze für leicht skalierbare Firewalls vorgestellt, die auch in Netzen mit 155 MBit/s und noch höheren Übertragungsraten nicht zum Engpaß werden.

4.1 Performanzkriterien für Firewalls

Performanz ist ein Oberbegriff für eine Reihe von Leistungskriterien:

Datendurchsatz: Das meist an erster Stelle genannte Performanzkriterium von Firewalls ist der Datendurchsatz (MBit/s).[2] Der (Nutz-)Datendurchsatz kann weiter unterschieden werden in Einzel- und Gesamtdurchsatz. Werden mehrere parallele Verbindungen (oder allgemeiner Kommunikationsbeziehungen) durch einen Firewall geführt, ergibt sich der Gesamtdurchsatz aus der Summe der Einzeldurchsätze. Handelt es sich lediglich um eine einzelne Verbindung, so ist Einzel- gleich Gesamtdurchsatz.

Da bei statischen Packet Screens Dateneinheiten in der Regel unabhängig von vorher verarbeiteten Dateneinheiten untersucht werden und Dateneinheiten sequentiell an der Packet Screen ankommen, ist bei Packet Screens eine Unterscheidung zwischen Einzel- und Gesamtdurchsätzen nicht notwendig. Daher können sich die Messungen auf wesentlich einfachere Einzeldurchsatzmessungen beschränken. Sonderfälle wie dynamische Packet Screens und das "Caching" von Filterergebnissen, bei denen eine unabhängige Bearbeitung einzelner Dateneinheiten nicht mehr gewährleistet sind, werden hier nicht betrachtet.

Bei Proxy-Servern ist eine Unterscheidung zwischen Einzel- und Gesamtdurchsatz jedoch wichtig. Der Einzeldurchsatz, den ein Client auf einer Verbindung über den Proxy-Server erreichen kann, ist dabei von geringerer Bedeutung. Zwar wäre es schön, wenn ein allein zugreifender Client auch den gesamten zur Verfügung stehenden Durchsatz erzielen könnte, jedoch kann der theoretisch mögliche Durchsatz nur von wenigen Clients voll ausgenutzt werden.[3] In der Praxis ist es daher viel wichtiger, daß die Summe der Einzeldurchsätze von "n" parallelen Verbindungen dem theoretisch erreichbaren Gesamtdurchsatz entsprechen. Dabei sollte die Aufteilung des Durchsatzes zwischen den einzelnen Verbindungen möglichst fair sein, d. h. alle Einzeldurchsätze sollten etwa gleich hoch sein. Die Messung des Gesamtdurchsatzes von mehreren parallelen Verbindungen stellt höhere Anforderungen an das Meßwerkzeug, da eine Synchronisation der parallelen Verbindungen notwendig ist.

---

[2] Eine wichtige Fehlerquelle ist die abweichende Verwendung der Faktoren "Kilo" und "Mega" bei Speichergrößen und beim Datendurchsatz. Bei der Angabe von Speichergrößen bezeichnet "Kilo" den Faktor $2^{10} = 1.024$. Ein "Mega-Byte" (MByte) sind entsprechend $2^{20}\ Byte = 1.024 * 1.024\ Byte = 1.048.576\ Byte$. Im Gegensatz zur Speichergröße liegen dem Datendurchsatz immer Takte zugrunde, die in MHz $= 10^6$ Hz angegeben werden. Ein Datendurchsatz von 10 MBit/s bezeichnet daher auch einen Durchsatz von $10 * 10^6$ Bit / Sekunde.

[3] Die beschränkte Leistungsfähigkeit der Speichermedien (bei FTP), eine aufwendige Darstellung der Ergebnisse beim Client (HTTP) oder unzureichende Verbindungsdauer (HTTP) verhindern das Ausschöpfen des zur Verfügung stehenden Netzdurchsatzes auf einer einzelnen Verbindung.

Datagrammdurchsatz: Eine Packet Screen muß vor der Übertragung eines IP-Datagramms zunächst die Filterregeln überprüfen. Mit steigender Anzahl von Filterregeln wird daher der Datagrammdurchsatz einer Packet Screen sinken.

Nachrichtenlaufzeit: Die Überprüfung von Daten im Firewall verzögert die Übertragung zum Empfänger. Aufgrund des höheren Aufwands ist bei Proxy-Servern im Vergleich zu Packet Screens mit einer größeren Verzögerung zu rechnen.

Verbindungsaufbauzeit: Firewalls verlängern die Verbindungsaufbauzeiten zu einem Server auf der anderen Seite des Firewalls erheblich, da zunächst eine Verbindung zum Proxy-Server aufgebaut werden muß, bevor eine zweite Verbindung vom Proxy-Server zum Server aufgebaut werden kann. Packet Screens wirken hier nur indirekt über die Verlängerung der Nachrichtenlaufzeiten auf die Verbindungsaufbauzeit.

Maximale Anzahl paralleler Verbindungen: Firewalls in Hochgeschwindigkeitsnetzen müssen meistens eine große Anzahl von Benutzern unterstützen, die gleichzeitig Verbindungen über den Firewall führen. Da bei Firewalls auf Anwendungsebene für jede Verbindung Kontextinformationen gespeichert werden müssen, ist die maximale Anzahl paralleler Verbindungen begrenzt. Neben der Begrenzung des Speicherplatzes ist auch die Bedienstrategie des Betriebssystems des Firewall-Rechners für die Beschränkung der Anzahl paralleler Verbindungen verantwortlich. So ist zu erwarten, daß ab einer bestimmten Anzahl paralleler Verbindungen der Gesamtdurchsatz sinkt oder daß nur noch ein Teil der Verbindungen ausreichende Fortschritte macht. Als Ursache hierfür kommen in Frage die Kontextwechsel zwischen den Prozessen, die die einzelnen Verbindungen bearbeiten, und die Last, die durch die Überprüfung der übertragenen Daten bedingt ist.

Die maximale Anzahl paralleler Verbindungen ist bei Packet Screens nicht relevant, da die Packet Screen Dateneinheiten unabhängig von deren Zugehörigkeit zu einzelnen Verbindungen untersucht.

Transaktionsdurchsatz: Protokolle wie HTTP, bei denen schnell hintereinander Verbindungen zur Übertragung von jeweils nur geringen Datenmengen aufgebaut werden, belasten einen Firewall auf Anwendungsebene beträchtlich. Werden zusätzlich aufwendige Authentisierungsverfahren eingesetzt, wird der maximale Durchsatz an Transaktionen pro Sekunde schnell erreicht.

Wieviele Transaktionen parallel von einem Proxy-Server verarbeitet werden können, ist daher ein wichtiger Parameter.[4]

Da Packet Screens nicht aktiv am Verbindungsauf- und -abbau teilnehmen, ist dieser Wert für Packet Screens nicht relevant.

4.2 Das Testnetz

Bevor mit den Messungen der Performanz von Firewall-Komponenten begonnen werden kann, müssen zunächst die Eckwerte ermittelt werden. Erst wenn die Eckwerte der Testumgebung bekannt sind, kann die Auswirkung unterschiedlicher Firewall-Architekturen bewertet werden. Für die Vergleichbarkeit der Testergebnisse ist ferner eine genaue Beschreibung des Testnetzes, der Systeme und der Meßwerkzeuge notwendig.

Das Testnetz besteht in der ersten Ausbaustufe aus sechs "Sun Ultra Sparc 1/140", die über Sun ATM-Schnittstellen (SunATM 2.1) an einen "Cisco Lightstream 1010" ATM-Switch angeschlossen sind. Verwendet wird eine kostengünstige UTP[5]-Verkabelung, die für Verbindungen über kurze Distanzen im Labor durchaus ausreichend ist. Auf den Workstations wird das Betriebssystem Solaris 2.6 eingesetzt. Um eine Beeinflussung der Messungen zu vermeiden, werden unnötige Prozesse und Dämonen von den Workstations entfernt. Die üblicherweise nachts ablaufenden Systemwartungsprozeduren (Verschieben von Log-Dateien, Anlegen von Indizes und ähnliche Aufgaben) wurden entfernt oder auf einen Zeitpunkt verlegt, an dem keine Tests laufen werden. Durch diese Maßnahmen wurde sichergestellt, daß Meßwerte nicht durch andere Prozesse verfälscht werden und somit auch reproduzierbar sind. Eine weitere Quelle von Störeinflüssen ist die Belastung des Netzwerkes durch nicht an den Tests beteiligte Systeme. Diese Einflüsse können weitestgehend ausgeschlossen werden, da neben den beschriebenen Systemen keine weiteren Systeme an das ATM-Netz angeschlossen sind. Ferner besteht die Möglichkeit des Zugriffs auf die Testrechner über zusätzliche zu den ATM-Schnittstellen vorhandene Ethernet-Schnittstellen. Der Zugang zu diesem isolierten Ethernet-Segment ist über einen Firewall abgesichert. Der Firewall achtet darauf, daß während der Messungen keine Dateneinheiten über das Ethernet gesendet werden.

[4] Die Messung der maximalen Anzahl an Transaktionen unterscheidet sich von der Messung der maximalen Anzahl paralleler Verbindungen in der geringeren Verbindungsdauer. Die maximale Anzahl paralleler Verbindungen bezieht sich auf offene Verbindungen, während für die Messung der maximalen Anzahl von Transaktionen kurze Verbindungen kontinuierlich geöffnet und wieder geschlossen werden.

[5] "Unshielded Twisted-Pair" (UTP)

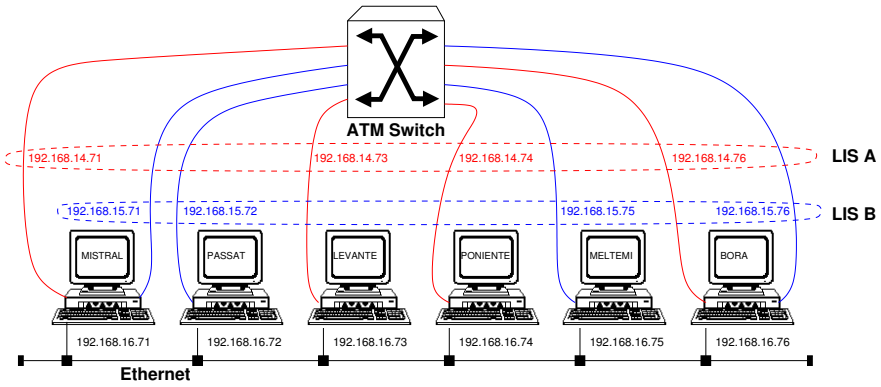


Abbildung 4.1: Das Testnetz: flexible Infrastruktur

Die Komponenten wurden in der in Abbildung 4.1 dargestellten Konfiguration aufgebaut. Auf dem ATM-Netz wurden zwei logische IP-Subnetze (LIS) — LIS A: 192.168.14.0 und LIS B: 192.168.15.0 — unter CLIP eingerichtet. Der ATM-Switch dient als ATMARP-Server für beide LIS. Das Schalten von SVCs,[6] die von CLIP für die Kommunikation zwischen zwei im selben LIS befindlichen IP-Hosts verwendet werden, erfolgt vor jeder Messung, so daß Auswirkungen auf die Messungen durch das Verbindungsmanagement auf ATM-Ebene vermieden werden. Zwei Workstations verfügen über jeweils zwei ATM-Schnittstellen, so daß sie einen Übergang zwischen den beiden Netzen (LIS) bilden können. Je nach angestrebter Konfiguration können diese beiden Rechner als Packet Screen oder Gateway-Firewall dienen, beziehungsweise ohne Filterfunktion als Router eingesetzt werden. Da der Wechsel einer Workstation von einem zum anderen Subnetz ohne Eingriffe in die Hardware über die Änderung einer Konfigurationsdatei möglich ist, entsteht eine flexible Infrastruktur zum Aufbau unterschiedlicher Firewall-Architekturen.

Da sich bei der Ermittlung der Eckwerte zeigte, daß die Performanz der "Ultra-1"-Systeme nicht ausreicht, um die in typischen Einsatzszenarien notwendige Last zu generieren, wurde das Testnetz in einer zweiten Ausbauphase um zwei besonders leistungsfähige "Sun Ultra-2 2/300" ergänzt (s. Abb. 4.2). Diese Systeme verfügen mit jeweils zwei 300 MHz CPUs über die etwa vierfache Leistung im Vergleich zu den "Ultra-1"-Systemen. Beide "Ultra-2"-Systeme erhalten 622 MBit/s ATM-Schnittstellen. Da diese ATM-Schnittstellen die TCP-Prüfsumme bereits in der Hardware berechnen können, ist eine Entlastung der Workstations möglich. Alle Meßergebnisse und Performanzdaten in den folgenden Abschnitten beziehen sich, sofern nicht anderweitig gekennzeichnet, immer auf "Ultra-1"-Systeme mit 155 MBit/s ATM-Schnittstellen. *Sämtliche Messungen mit "Ultra-2"-Systemen*

[6] Für CLIP wird die ATM Verkehrsklasse UBR ("Unspecified Bit Rate") verwendet. Bei UBR werden keine Leistungseigenschaften garantiert ("Best-Effort").

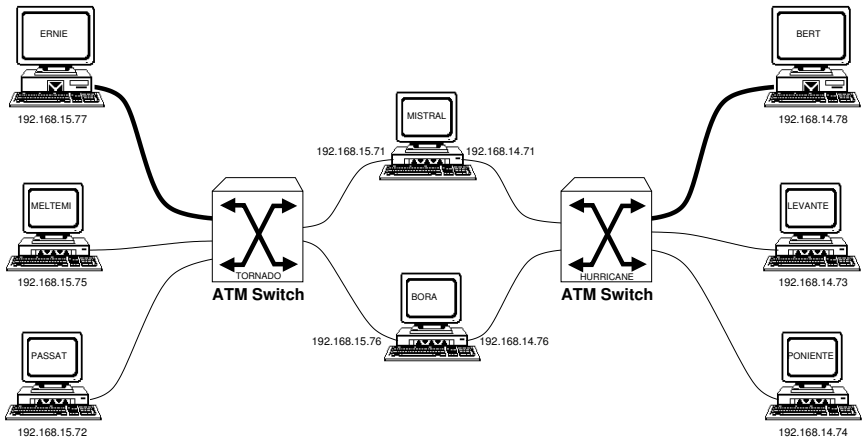


Abbildung 4.2: Das Testnetz: zweite Ausbaustufe

und/oder über 622 MBit/s ATM-Schnittstellen sind ausdrücklich als solche gekennzeichnet.

Die folgenden Unterabschnitte 4.2.2 bis 4.2.9 diskutieren detailliert die wichtigsten Eckwerte der Performanz (s. Abschnitt 4.1) bei direkter Host-zu-Host-Kommunikation im Testnetz. Die ermittelten Eckwerte dienen bei der Bewertung der Performanz von Firewall-Konzepten in den folgenden Abschnitten (4.3 bis 4.5) als Referenzwerte.

4.2.1 Meßwerkzeug "Netperf"

Für die Messungen von Datendurchsatz, Nachrichtenlaufzeit, Verbindungsaufbauzeit und Transaktionsdurchsatz wird das Meßwerkzeug "Netperf" (V 2.1pl1)[7] eingesetzt. Dieses Meßwerkzeug besteht aus einem Server und einem Client. Alle Messungen werden vom Client initiiert. Dieser ermöglicht Messungen unter Variierung von diversen Parametern.[8] Zur statistischen Absicherung der Meßergebnisse wird bei der Datendurchsatzmessung jeder Meßpunkt mehrfach wiederholt und dabei Mittelwert sowie Standardabweichung berechnet. Die Berechnung von Vertrauensintervallen wurde nachträglich ergänzt. Bei den Messungen der Nachrichtenlaufzeiten und des Transaktionsdurchsatzes waren größere Änderungen notwendig, um auch bei diesen Messungen eine statistische Absicherung der Meßergebnisse zu implementieren.

[7] Verfügbar über `ftp://ftp.cup.hp.com/dist/networking/benchmarks/netperf/`

[8] Neben der Nutzdatenlänge, der Testdauer, der Größen der Sende- und Empfangspuffer, die durch Parameter beim Meßwerkzeug selbst eingestellt werden, können bei Solaris zusätzlich über `ndd` die Parameter des Kernels für die TCP-, UDP- und IP-Verarbeitung variiert werden.

4.2.2 Datendurchsatz

Der wichtigste Eckwert ist der von einer Anwendung auf einer TCP-Verbindung erreichbare Durchsatz zwischen zwei Anwendungsprozessen, der als Benutzerdatendurchsatz bezeichnet wird. Auf den für die Messungen verwendeten Rechnern ist die Schnittstelle zwischen Anwendungsprozessen und dem Kommunikationssystem als sogenannte "Socket"-Schnittstelle implementiert. Das Meßwerkzeug öffnet vor der Messung eine TCP-Verbindung und übergibt anschließend über die gewünschte Meßdauer (Δt) Nachrichten der Länge (l Byte) an den "Socket". Auf dem empfangenden System wird gemessen, wieviele Nachrichten vom "Socket" an die Anwendung übergeben werden. Aus der Summe der Bytes aller n empfangenen Nachrichten innerhalb der Meßdauer (Δt) kann der Benutzerdatendurchsatz auf der TCP-Verbindung berechnet werden (Gleichung 4.1).

$$Benutzerdatendurchsatz\,[Byte/s] = \frac{n * l\ Byte}{\Delta t} \tag{4.1}$$

Im weiteren Verlauf wird der Benutzerdatendurchsatz abgekürzt als Datendurchsatz bezeichnet. Die Abbildung 4.3 zeigt den maximalen Datendurchsatz über eine TCP-Verbindung in Abhängigkeit von der Nachrichtenlänge (l) und dem an der "Socket"-Schnittstelle zur Verfügung stehenden Speicher ("Socket"-Größe). Die einzelnen Messungen dauerten jeweils 10 Sekunden ($\Delta t = 10\,s$)[9] und wurden zur statistischen Absicherung der Ergebnisse für jeden Meßpunkt 30 bis 35 Mal wiederholt. Bei einer Abweichung um maximal $\pm 2.5\%$ vom Erwartungswert soll die zu erreichende Vertrauenswahrscheinlichkeit 99% betragen.[10] Die Vertrauensintervalle werden bei den folgenden zweidimensionalen Plots ebenfalls dargestellt.

[9] Die Meßdauer muß ausreichend lang gewählt werden, da zu Beginn einer Messung verschiedene Anlaufeffekte den Durchsatz beeinflussen. Neben dem Auffüllen von leeren Socket-Puffern wird der maximale Durchsatz beim Beginn einer TCP-Verbindung durch das "(TCP-)Slow-Start"-Verfahren gewollt reduziert. Dieses Verfahren wirkt zudem ungünstig mit einem anderen Verfahren dem "Deferred Acknowledge" [Clark 82] zusammen. Dabei wird eine feste Zeit (bei Solaris 50 Millisekunden) gewartet, bevor ein ACK vom Empfänger zurückgesendet wird. Dieses Verfahren soll der Anwendung Zeit geben, damit dann zusammen mit dem ACK gleich Nutzdaten der Applikation mit versendet werden können. Zur Ermittlung des maximalen Durchsatz einer Testkonfiguration müssen die Auswirkungen von Anlaufeffekten soweit wie möglich reduziert werden. Auf der anderen Seite sollte die Meßdauer klein gewählt werden, um die Testergebnisse mit möglichst geringem Zeitaufwand zu ermitteln.
In einer Meßreihe wurde der maximale Durchsatz bei unterschiedlich langer Meßdauer mit Δt = (5, 10, 15, 20, 30, 45 und 60 Sekunden) ermittelt. Dabei zeigte sich, daß bereits ab einer Meßdauer von 10 Sekunden die Auswirkungen der Anlaufeffekte vernachlässigt werden können und eine akzeptable Genauigkeit erreicht wird. [Ellermann et al. 98a]

[10] Eine Vertrauenswahrscheinlichkeit von 99% gibt an, daß der tatsächliche Erwartungswert mit einer Wahrscheinlichkeit von 99% in dem berechneten Intervall liegt. [Sachs 92]

An dieser Stelle sei schon auf die sehr kleinen Vertrauensintervalle hingewiesen, die eine qualitative Auswertung von Messungen mit unterschiedlichen Konfigurationen erst ermöglichen.

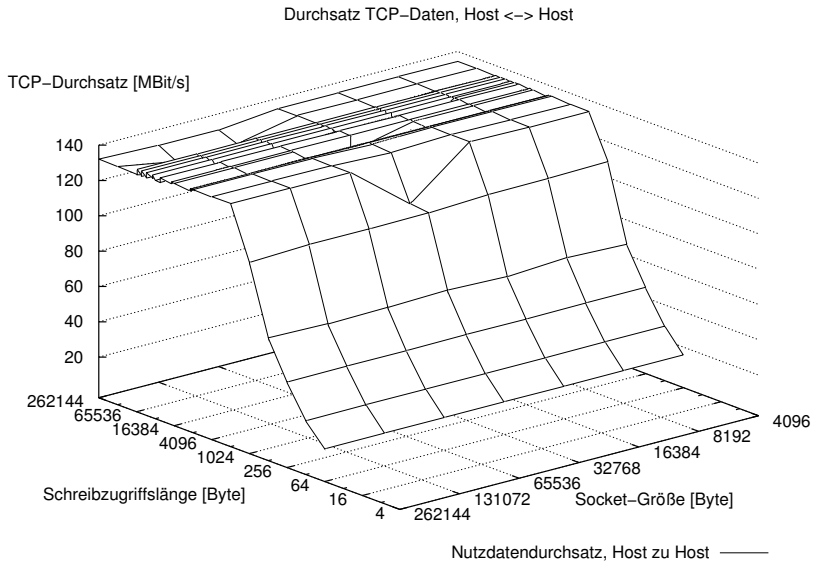


Abbildung 4.3: Maximaler Datendurchsatz über TCP

4.2.2.1 Auswahl der Meßpunkte: Nachrichtenlänge

Der erreichbare Datendurchsatz hängt von der Datagrammlänge ab. Da die Header-Länge für kurze und lange Datagramme konstant ist, muß bei kurzen Datagrammen ein größerer anteiliger "Overhead" des Headers zu den übertragenen Nutzdaten akzeptiert werden. Um einen Überblick zu gewinnen, werden die Meßpunkte zunächst so gewählt, daß der gesamte Bereich der vom Meßwerkzeug unterstützten Nutzdatenlängen[11] abgedeckt ist. Begonnen wird mit einer Nutzdatenlänge von 4 Byte.[12] Die weiteren Meßpunkte liegen jeweils bei den Zweierpotenzen und enden bei dem Wert von $2^{18} = 262.144$ Byte.

[11] Bei "Netperf" wird die Bezeichnung "Messagesize" (Nachrichtenlänge) verwendet. Um Verwechselung mit anderen Nachrichtenlängen — etwa Längen von E-Mails, HTML-Seiten, usw. — zu vermeiden, soll hier der Begriff Nutzdatenlänge verwendet werden. Die Nutzdatenlänge wird im ISO-OSI Modell als "Service Data Unit Size" kurz "SDU-Size" bezeichnet. Die Nutzdatenlänge ergibt zusammen mit der Header-Länge die PDU-Länge ("Protocol Data Unit Size"). Für die unterschiedlichen Protokolle werden die PDU-Längen bezeichnet als TCP-Segmentlänge, UDP-Datagrammlänge oder IP-Datagrammlänge.

[12] Bei späteren Messung wurde mit einer Nutzdatenlänge von 1 Byte begonnen.

Ein wichtiger Bereich, der mit zusätzlichen Meßpunkten genauer analysiert werden soll, liegt beim Erreichen der MTU-Größe. Die "Maximum Transfer Unit" (MTU) ist auf "Classical IP over ATM"-Netzen mit 9.180 Byte voreingestellt [Atkinson 94, Laubach et al. 98]. Mit der MTU wird die Größe einer IP-Dateneinheit begrenzt, die maximal von der unter IP liegenden Infrastruktur übertragen werden kann. Größere Nutzdaten müssen von IP fragmentiert werden oder werden im Falle von TCP bereits in mehreren TCP-Segmenten übertragen. Daraus resultiert ein Gemisch aus kurzen und langen Dateneinheiten. Weitere Meßpunkte werden ebenfalls bei den Vielfachen der MTU-Größe festgelegt.

4.2.2.2 Unterschied zwischen Nutzdatenlänge und Schreibzugriffslänge

Das Meßwerkzeug (Netperf) verwendet bei der Messung jeweils Schreibzugriffe mit der angegebenen Nutzdatenlänge. Jeder Schreibzugriff soll zur Generierung genau einer IP-Dateneinheit führen. Da Kernel und Anwendung voneinander getrennt sind, wird der Anwendung keine direkte Kontrolle der Generierung von IP-Dateneinheiten erlaubt. Schreibzugriffe auf ein Socket[13] sind in UNIX durch das Kopieren der Daten in einen Puffer realisiert. Die Anwendung überläßt es dann dem Kernel, diese Daten weiterzuverarbeiten. Grundsätzlich sollte der Kernel dann die Daten in ein TCP-Segment und anschließend zu einem IP-Datagramm weiterverarbeiten. Da es jedoch effizienter ist, statt vieler kleiner ein großes Segment zu versenden, werden TCP-Segmente nicht immer sofort generiert. Der Algorithmus von Nagle [Nagle 84] verzögert das Generieren von TCP-Segmenten unter gewissen Voraussetzungen[14] zur Optimierung des Durchsatzes.

Folglich kann auch das Meßwerkzeug "Netperf" nicht garantieren, daß die in einem Schreibzugriff übergebenen Daten in genau einem TCP-Segment resultieren.[15] Durch die `TCP_NODELAY`-Option ist es zumindest möglich, die beabsichtigte Optimierung des Kernels von mehreren Schreibzugriffen zu einem TCP-Segment

---

[13] Ein Socket ist ein spezieller Datentyp in BSD-UNIX-Systemen, der Anwendungen den Zugriff auf das Netz ermöglicht.

[14] Sofern noch unbestätigte TCP-Segmente ausstehen, sollte mit dem Senden kleinerer Dateneinheiten gewartet werden.
John Nagle: "Congestion Control in IP/TCP Internetworks" (RFC-896) [Nagle 84]

> "The solution is to inhibit the sending of new TCP segments when new outgoing data arrives from the user if any previously transmitted data on the connection remains unacknowledged."

[15] Dies ist auch der Grund dafür, daß Messungen mit Schreibzugriffslängen von 262.144 Byte durchgeführt werden können, obwohl die maximale Segment-Größe von 65.495 Byte dabei überschritten wird. "Maximum Segment Size" (MSS) bezeichnet die maximale Nutzdatenlänge, die in einem TCP-Segment übertragen werden kann. Sie ergibt sich aus der maximalen Länge einer IP-Dateneinheit (65.535 Byte) abzüglich der IP und TCP Header-Längen (zusammen 40 Byte).

zu unterdrücken. Der Fall, daß wegen eines gefüllten Fensters ("Window", s. u.) oder wegen der Belastung des Systems das sofortige Senden der Daten verhindert wird und deshalb mehrere Schreibzugriffe in einem Segment resultieren, kann durch die `TCP_NODELAY`-Option nicht beeinflußt werden, so daß auch unter Ausnutzung dieser Option immer noch ein gewisser Anteil der Segmente eine größere Länge hat.

Die Aggregation von mehreren kleinen Schreibzugriffen zu einem großen TCP-Segment kann zuverlässig verhindert werden, indem die MTU-Größe reduziert wird. Um die Fragmentierung eines TCP-Segmentes zu verhindern, reduziert TCP die "Maximum Segment Size" (MSS) auf die MTU-Größe abzüglich 40 Byte für IP- und TCP-Header. Für jeden Meßpunkt muß die MTU-Größe auf die jeweilige Schreibzugriffslänge angepaßt werden, so daß die Daten eines Schreibzugriffs zusammen mit den TCP- und IP-Protokoll-Headern in genau einer MTU übertragen werden können. Da keine IP-Datagramme gesendet werden können, deren Länge die MTU-Größe überschreitet, kann so effektiv auch die Aggregation mehrerer Schreibzugriffe zu einem großen TCP-Segment verhindert werden.[16]

Die Auswirkungen durch die Reduzierung der MTU-Größe auf den Durchsatz im Vergleich zu einer Messung mit der Standard-MTU-Größe [17] zeigt die Abbildung 4.4. Deutlich erkennbar ist, wie der Durchsatz bei kürzeren Nutzdatenlängen durch die Aggregation mehrerer Datenblöcke von getrennten Schreibzugriffen zu einem TCP-Segment gesteigert werden kann. Ist der maximale Durchsatz von etwa 134 MBit/s jedoch erreicht, kann durch das Zusammenpacken kein Vorteil mehr erreicht werden. Ferner ist an dem etwas ungleichmäßigerem Verlauf der Kurve mit Grundeinstellung erkennbar, daß der Performanzvorteil durch das Zusammenpacken für manche Nutzdatenlängen größer ist. Für Schreibzugriffslängen unterhalb von 32 Byte können mit diesem Verfahren keine Meßwerte

[16] Mehrere Schreibzugriffe werden von TCP nur bis zum Erreichen der MSS aggregiert. TCP aggregiert keine Schreibzugriffe zu einem großen TCP-Segment, das anschließend von IP wieder fragmentiert werden müßte.
Alternativ wäre es bei den im Testnetz verwendeten Solaris-Systemen auch möglich gewesen, über `ndd` die "Maximum Segment Size" (MSS) zu reduzieren und so direkt Einfluß auf TCP zu nehmen. Diese Möglichkeit des Setzens der MSS besteht bei anderen Betriebssystemen jedoch häufig nicht.

[17] Die Reduzierung der MTU-Größe bewirkt auch ohne die `TCP_NODELAY`-Option die gewünschte Verhinderung der Aggregation mehrerer Schreibzugriffe zu einem großen TCP-Segment. Wie spätere Messungen bestätigten, hat das zusätzliche Setzen der `TCP_NODELAY`-Option in diesem Falle keine Auswirkungen auf das Verhalten der Systeme. Da zu diesem Zeitpunkt bereits Messungen vorlagen, bei denen beide Maßnahmen aktiviert waren, wurde auch bei folgenden Messungen mit Reduzierung der MTU-Größe zusätzlich die `TCP_NODELAY`-Option eingeschaltet, um eine Vergleichbarkeit mit bereits durchgeführten Messungen zu gewährleisten.

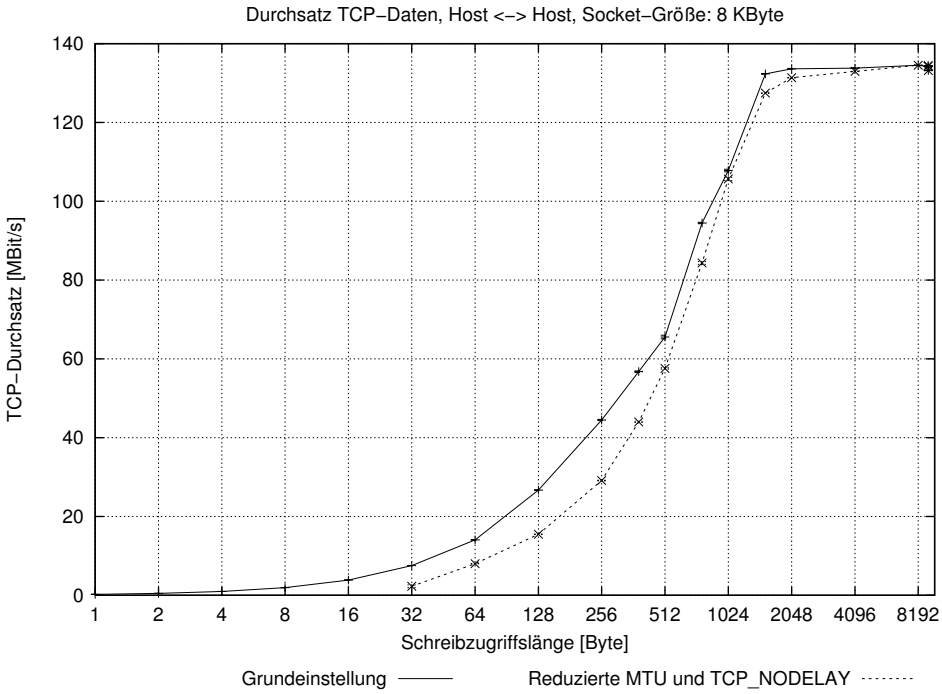


Abbildung 4.4: Durchsatz mit reduzierter MTU-Größe und `TCP_NODELAY`

ermittelt werden, da die verwendeten Solaris 2.6 Systeme die Reduzierung der MTU unter 72 Byte[18] nicht zulassen.

Ob eine Messung in der Grundeinstellung oder mit reduzierter MTU-Größe durchgeführt werden muß, hängt davon ab, ob die Performanz auf Anwendungs- oder auf Netzwerkschicht gemessen werden soll. In den meisten Fällen soll der Durchsatz gemessen werden, den eine Anwendung in der Testumgebung erreichen kann. Diese Messungen der Anwendungsperformanz sind vorzugsweise in der Grundeinstellung durchzuführen. Demgegenüber hängt die Performanz von Komponenten auf der IP-Ebene, wie Packet Screens, von der Anzahl und der Länge der Datagramme ab. Die Performanz von Packet Screens muß daher mit reduzierter MTU-Größe gemessen werden. Bei allen in diesem Dokument enthaltenen Meßergebnissen und Grafiken wird explizit angegeben, wenn die Messung mit gesetzter MTU gemessen wurde. Erfolgt keine Angabe, so wurde die Messung in der Grundeinstellung durchgeführt.

[18] 32 Byte Nutzdaten erreichen zusammen mit 20 Byte IP- und 20 Byte TCP-Header die MTU-Größe von 72 Byte.

4.2.2.3 Auswahl der Meßpunkte: Socket-Größe

Die wichtige Rolle der Socket-Größe in Hochgeschwindigkeitsnetzen war bereits Gegenstand vieler Untersuchungen [Comer et al. 94, Campanella et al. 96a, Jacobson et al. 92, Andrikopoulos et al. 96, Andrikopoulos et al. 97, Bianco 94]. Die Socket-Größe ("Socket Size") gibt den für eine (TCP-)Verbindung reservierten Pufferspeicher an. Der Sender speichert Dateneinheiten im Puffer, bis der korrekte Empfang vom Kommunikationspartner bestätigt wird. Bleibt die Bestätigung aus, wiederholt TCP das Senden der Dateneinheiten, ohne daß die Daten erneut von der Applikation angefordert werden müssen. Die Größe des Pufferspeichers muß an die Datenmenge angepaßt werden, die im Netz übertragen werden kann, bis die Empfangsbestätigung beim Sender ankommt. Das Produkt aus der Übertragungsgeschwindigkeit und der Verzögerungszeit bis zum Eintreffen der Bestätigung[19] (*"Bandwidth x Delay Product"*, BDP) gibt die Datenmenge an, die das Netz quasi "speichert". Eine Applikation kann nur dann die Übertragungsgeschwindigkeit eines Netzes ausnutzen, wenn der zur Verfügung stehende Puffer größer ist als das BDP. Ist der Puffer hingegen mit unbestätigten Dateneinheiten gefüllt, dann werden die Schreibaufrufe der Applikation blockiert, bis erneut Speicher frei wird. In Hochgeschwindigkeitsnetzen verschärft sich dieses Problem, da ein hoher Durchsatz erreicht wird, aber die Verzögerungszeit praktisch gleich bleibt. Die Lichtgeschwindigkeit bildet hier eine nicht verschiebbare Obergrenze. Sende- und Empfangspuffer müssen daher in Hochgeschwindigkeitsnetzen größer sein als in herkömmlichen Netzen und in Weitverkehrsnetzen größer als in LANs.

Auf der Empfängerseite werden die Dateneinheiten ebenfalls in einem Puffer abgelegt. Da die Einhaltung der Reihenfolge der Dateneinheiten in IP Netzen nicht gewährleistet wird, müssen Dateneinheiten gespeichert werden, bis noch fehlende verzögerte Dateneinheiten empfangen werden. Sofern nicht anders angegeben, werden bei den folgenden Messungen immer dieselben Socket-Größen sowohl für den Sende- als auch den Empfangspuffer reserviert.

Die Socket-Größe wirkt direkt auf einen weiteren wichtigen Performanzparameter — die Fenstergröße ("TCP Window Size"). Sie gibt an, wieviele TCP-Segmente unbestätigt gesendet werden dürfen. Da die TCP-Segmente im Socket-Puffer gespeichert werden müssen, wird die Fenstergröße begrenzt auf die Anzahl der maximalen TCP-Segmente ("Maximum Segment Size", MSS), die im Socket-Puffer gespeichert werden können. Im Laufe einer Verbindung kann die Fenstergröße variieren (siehe Abschnitt 4.2.9).

In der Abbildung 4.3 zur Messung des maximalen Durchsatzes wird die Socket-Größe in Zweierpotenzen von 4.096 (2^{12}) bis 262.144 Byte (2^{18}) variiert. In Ab-

[19] Die Zeit bis zum Eintreffen der Bestätigung wird als "Delay" oder auch als "Round-Trip Time" (RTT) bezeichnet.

bildung 4.5[20] sind die Meßwerte derselben Messung für einzelne Socket-Größen in zweidimensionaler Darstellung wiedergegeben.

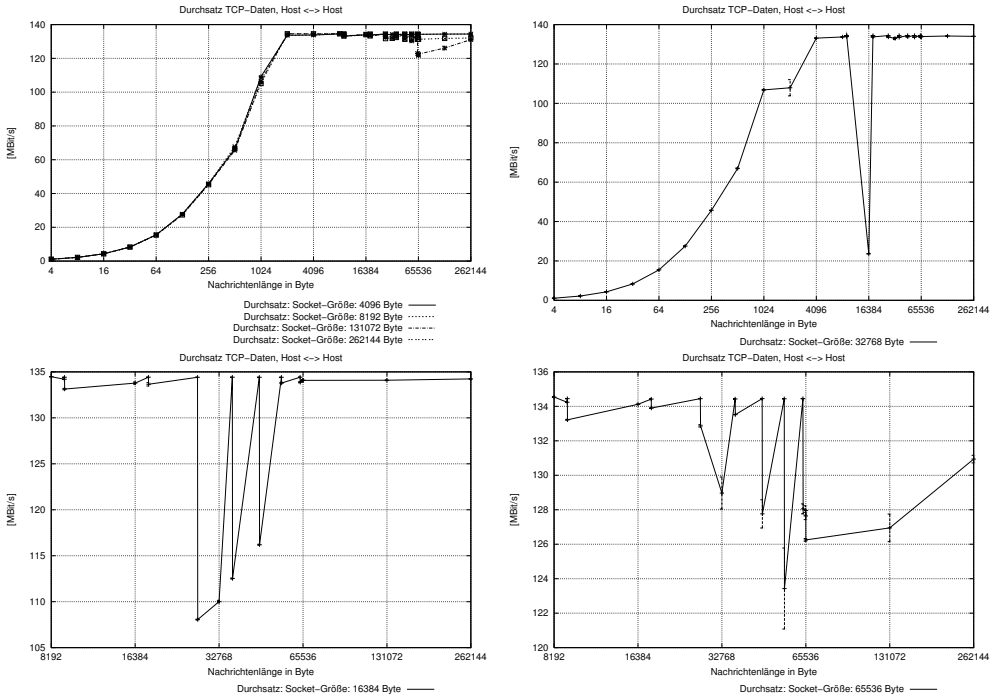


Abbildung 4.5: Maximaler Durchsatz über TCP

page

Die wichtigsten Ergebnisse dieser Messung sind:

- Der maximale Durchsatz über TCP beträgt 134,55 MBit/s und wird erreicht bei einer Socket-Größe von 262.144 Byte und einer Nutzdatenlänge von 4.096 sowie 8.192 Byte. Dieser Wert erreicht fast das theoretische Maximum von 134,58 MBit/s. (s. Abschnitt 4.2.2.5)

- Ein Durchsatz von 134 MBit/s ist nur mit Nachrichtengrößen über 2.048 Byte erreichbar (s. Abb. 4.5.1). Bei kleineren Nachrichtengrößen wird ein zu großer Anteil der zur Verfügung stehenden Übertragungsleistung von dem mit jeder Nachricht verbundenen Protokoll-"Overhead" verbraucht.

- Bei der Socket-Größe von 32 KByte (s. Abb. 4.5.2) gibt es zwei deutliche Einbrüche der Meßwerte bei den Nutzdatenlängen von 2.048 und 16.384 Byte. Bei diesen Nutzdatenlängen werden maximale Durchsätze von lediglich 108 MBit/s (2.048 Byte Nutzdatenlängen) bzw. 24 MBit/s (16.384

[20] Die Meßkurven sind in voller Größe im Anhang B auf den Seiten 456 bis 458 enthalten.

Byte Nutzdatenlängen) erreicht. Insbesondere der extreme Performanzverlust bei 16 KByte langen Nachrichten weist auf Implementationsprobleme hin. Vergleichsmessungen mit Solaris 2.5.1 zeigten bei sonst gleicher Konfiguration diesen Einbruch nicht. Die beiden Einbrüche treten auch unter Solaris 2.6 bei Vergleichsmessungen mit `TCP_NODELAY`-Option (s. o.) nicht mehr auf [Ellermann et al. 98a].

- Bei den Socket-Größen von 16 und 64 KByte sind ebenfalls Einbrüche zu erkennen (Abb. 4.5.3 und 4.5.4). Ursache für die in den Abbildungen zu erkennenden starken Schwankungen zwischen benachbarten Werten sind ungünstige Aufteilungen, die bei langen Nachrichten mitunter zu eine Mischung von langen und kurzen TCP-Segmenten führen können. Dieser Effekt ist bereits beim Überschreiten der MTU-Größe zu beobachten. Bei bestimmten Kombinationen von Schreibzugriffslänge und Socket-Größe verstärkt sich dieser Effekt. Auf eine detaillierte Analyse der Ursachen für diese Verstärkung des Effektes wird an dieser Stelle aus zwei Gründen verzichtet. Zum einen verwenden gebräuchliche Anwendungen keine Schreibzugriffe, die länger als 8 KByte sind. Zum anderen führen Schreibzugriffe des Meßwerkzeuges mit einer Länge oberhalb der MTU in jedem Fall zum Generieren mehrerer TCP-Segmente. Die Ergebnisse von Messungen in diesem Bereich sind daher sehr stark von der Protokoll-Implementation in dem sendenden System abhängig. Da jedoch nicht die Performanz von den zur Messung verwendeten Endsystemen ermittelt werden soll, sondern die Performanz der zwischen den Endsystemen installierten Firewall-Komponenten gemessen werden soll, ist es ausreichend, Schreibzugriffe bis zu einer maximalen Länge der MTU-Größe durchzuführen. Durch diese Schreibzugriffe wird der gesamte Bereich der Nutzdatenlängen abgedeckt, die am Firewall ankommen können.

4.2.2.4 Auswirkungen asymmetrischer Sende- und Empfangspuffergrößen

Für die Untersuchung der Auswirkung der Puffergröße auf den maximalen Durchsatz wurden die Nutzdatenlängen von 16 bis 9.150 Byte variiert. Für jede Nutzdatenlänge wurde die Sende- und Empfangspuffergröße jeweils von 4 bis 256 KByte gegeneinander verändert, so daß zum Beispiel Aussagen über die Folgen von Konfigurationen mit großem Puffer beim Sender aber kleinem Puffer beim Empfänger gemacht werden können.

Da es sich bei dem Testnetz um ein LAN handelt, sind die Nachrichtenlaufzeiten sehr gering und deshalb ist auch das "Bandwidth x Delay Product" (s. o.) klein. Da es zudem kaum zum Verlust von Nachrichten bei der Übertragung kommt, haben Sende- und Empfangspuffergrößen für die gemessenen Werte des maximalen Durchsatzes nur eine geringe Bedeutung. Die Messungen zeigten vielmehr implementationspezifische Vorteile bei bestimmten Kombinationen von Nutzdaten-

länge, Sende- und Empfangspuffergrößen.[21] "Deadlocks" bei bestimmten Kombinationen von Sende- und Empfangspuffergrößen, wie in [Moldeklev et al. 95] beschrieben, werden bei dieser Messung nicht erreicht.[22]

Auf Basis dieser Messungen können abschließend keine optimalen Werte für die Sende- und Empfangspuffergrößen angegeben werden. Da Firewalls verzögernd auf die übertragenen Daten wirken, muß bei späteren Messungen mit Firewalls überprüft werden, ob durch die Wahl größerer Puffer bessere Ergebnisse erzielt werden können. Die Verzögerung durch den Firewall können so groß sein, daß eine günstige Wahl der Sende- und Empfangspuffergrößen bereits zu meßbarer Erhöhung des Durchsatzes führen kann.

4.2.2.5 Berechnung des theoretisch erreichbaren Datendurchsatzes

Der auf Anwendungsebene bei Dateneinheiten mit MTU-Größe zur Verfügung stehende Datendurchsatz wurde bereits von verschiedenen Autoren berechnet [Andrikopoulos et al. 96, Andrikopoulos et al. 97, Benecke et al. 97a]. Die ersten Messungen haben bereits gezeigt, daß die Systeme des Testnetzes sehr nah an diesen Wert herankommen. Bei kleineren Schreibzugriffslängen sinkt der gemessene Datendurchsatz jedoch rasch. Um auch in diesem Bereich Vergleichswerte zu haben, ist es notwendig, eine Funktion für den theoretisch erreichbaren Datendurchsatz in Abhängigkeit von der Nachrichtenlänge zu entwickeln. Die Entwicklung einer Funktion ist zwar zunächst aufwendiger als eine Berechnung für eine konkrete Nachrichtenlänge, jedoch sind anschließende Berechnungen durch einfaches Einsetzen der Parameter in die entwickelte Funktion möglich.

Der im Abschnitt 2.4.1 vorgestellte CLIP-Protokollstapel wird für die in diesem Kapitel folgenden Messungen zugrunde gelegt. Wichtig ist bei der Angabe der üblichen Übertragungsraten von ATM-Schnittstellen wie 155 oder 622 MBit/s, daß sich diese Angaben immer nur auf die Bruttoübertragungsrate beziehen. Da auch die auf verschiedenen Schichten hinzukommenden "Header" und "Trailer" übertragen werden müssen, verbleibt einer Anwendung nur eine geringere Nettoübertragungsrate. Dieser einer Anwendung zur Verfügung stehende Datendurchsatz und der sogenannte Protokoll-"Overhead" sollen in diesem Abschnitt für ein "Classical IP over ATM"-Netz in Abhängigkeit von der Nachrichtenlänge berechnet werden. Zuerst wird dafür "bottom-up" der Datendurchsatz über SONET berechnet, der unabhängig von der jeweiligen Nachrichtenlänge zur Verfügung steht. Der von der Nachrichtenlänge abhängige "Overhead" wird anschließend beginnend mit der Transportschicht "top-down" berechnet.

[21] Siehe [Ellermann et al. 98a]

[22] In [Moldeklev et al. 95] wird empfohlen, daß der Sendepuffer größer als der Empfangspuffer sein soll und daß die Größe des Sendepuffers mindestens drei MSS betragen soll.

Die Übertragungsgeschwindigkeiten von ATM-Netzen werden oft mit Kürzeln wie "STS-3" oder "OC-12" angegeben. "STS" (Synchronous Transport Signal) wird für elektrische Signale verwendet, während "OC" (Optical Carrier) für optische Signale Verwendung findet [Peterson et al. 96, S. 93]. Die angehängten Zahlen geben die Übertragungsgeschwindigkeit an. In der SONET[23]-Hierarchie wird definiert, daß ein STS-1 Rahmen immer 125 μs ($1,25 * 10^{-4}$ Sekunden) "lang" ist. Dies bedeutet, daß in einer Sekunde 8.000 Rahmen übertragen werden können [Hein et al. 98, S. 56-79]. Jeder STS-1 Rahmen hat eine Länge von 810 Byte (s. Abb. 4.6). Mit einem "Section-Overhead" von 9 Byte, einem "Line-Overhead" von 18 Byte und einem "Path-Overhead" von 9 Byte enthält jeder Rahmen einen "Overhead" von 36 Byte, und es bleiben Netto 774 Byte für die Nachricht übrig.

$$\begin{aligned} Bruttoübertragungsrate_{STS-1} &= 8000\ Rahmen/s * 810\ Byte/Rahmen \\ &= 6,480 * 10^6\ Byte/s * 8\ Bit/Byte \\ &= 51,840\ MBit/s \end{aligned}$$

$$\begin{aligned} Nettoübertragungsrate_{STS-1} &= 8000\ Rahmen/s * 774\ Byte/Rahmen \\ &= 6,192 * 10^6\ Byte/s * 8\ Bit/Byte \\ &= 49,536\ MBit/s \end{aligned}$$

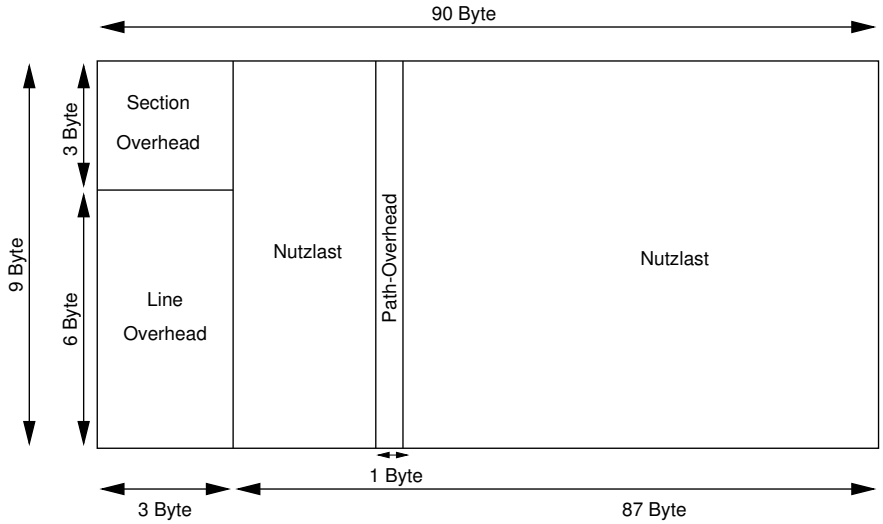


Abbildung 4.6: Struktur eines SONET STS-1 Rahmens

[23] "Synchronous Optical Network" (SONET)

STS-n Rahmen sind aus dem STS-1 Rahmen abgeleitet, haben jedoch eine n-fache Größe. Über STS-3 können dann $3*51,840$ MBit/s $= 155,520$ MBit/s übertragen werden. Der "Overhead" pro Rahmen verdreifacht sich ebenfalls. Bei der für ATM-Netze gebräuchlichen STS-3c Variante — das angehängte "c" steht für "*concatenated*" — werden nur der "Section-" und "Line-Overhead" verdreifacht, aber es erfolgt keine Verdreifachung des "Path-Overheads". Dadurch ergibt sich für die STS-3c Rahmenlänge von 2.430 Byte ein "Overhead" von 90 Byte — 27 Byte ("Section-Overhead"), 54 Byte ("Line-Overhead") und 9 Byte ("Path-Overhead"). Für die Nettoübertragungsrate über STS-3c ergeben sich dann:

$$\begin{aligned} Nettoübertragungsrate_{STS-3c} &= 8000\ Rahmen/s * 2340\ Byte/Rahmen \\ &= 18,720 * 10^6\ Byte/s * 8\ Bit/Byte \\ &= 149,760\ MBit/s \end{aligned}$$

Der darüberliegenden ATM-Schichten stehen somit von den ursprünglichen 155,520 MBit/s nur noch 149.760 MBit/s zur Verfügung; die Differenz von 5,76 MBit/s wird vom "SONET-Overhead" verbraucht. Einen Überblick der Übertragungsraten bei SONET gibt Tabelle 4.1.

Medium		Übertragungsrate [MBit/s]	Nettoübertragungsrate [MBit/s]	[10^6 Byte/s]
OC-1	STS-1	51,840	49,536	6,192
OC-3	STS-3	155,520	148,61	18,576
OC-3c	STS-3c	155,520	149,76	18,720
OC-12	STS-12	622,080	594,43	74,304
OC-12c	STS-12c	622,080	599,04	74,880
OC-24	STS-24	1.244	1.189	148,608
OC-48	STS-48	2.488	2.378	297,216
OC-192	STS-192	9.995	9.511	1188,864

Tabelle 4.1: SONET-Übertragungsraten

Während der "Overhead" auf der SONET-Schicht unabhängig von der übertragenen Nachrichtenlänge anfällt, soll im folgenden der von der Nachrichtenlänge

abhängige "Overhead" berechnet werden.[24] Die Tabelle 4.2[25] gibt zunächst einen Überblick über den auf verschiedenen Schichten anfallende "Protokoll-Overhead". [ATM Forum 94, ATM Forum 96a, Hein et al. 98, Peterson et al. 96]

Header	Header/Trailer-Länge	max. Nutzdatenlänge
TCP	**20 Byte**	keine Begrenzung in TCP
UDP	8 Byte	65.535 − 8 Byte
IP	**20 Byte**	65.535 − 20 Byte
IP-Fragment	20 Byte	9.160 Byte (bei MTU 9.180 Byte)
IPv6	40 Byte	65.535 − 40 Byte
IPv6-Fragment	40 + 8 Byte	9.132 Byte (bei MTU 9.180 Byte)
SNAP	**8 Byte**	
AAL-5	**8 Byte**	65.535 Byte
	+ 0–47 Byte Pad	
AAL-3/4	8 Byte	
	+ 4 Byte pro Zelle	
ATM-Zellen	**5 Byte**	**48 Byte**
STS-3c Rahmen	**90 Byte**	**2.340 Byte**
STS-1 Rahmen	36 Byte	774 Byte
Ethernet Frame	14 + 13 = 27 Byte	46 bis 1.500 Byte

Tabelle 4.2: "Protokoll-Overhead" in "Classical IP"-Netzen

Zu einer Nutzdatenlänge von x Byte addieren sich bei einer TCP-Verbindung in einem "Classical IP"-Netz noch TCP-, IP- und SNAP-Header sowie der AAL-5-Trailer als "Protokoll-Overhead". Die Gesamtlänge der Dateneinheit beträgt bei TCP dann "$x + 20 + 20 + 8 + 8 = x + 56$ Byte".[26]

$$\textit{Nutzdatenlänge mit Headern} = x\ \textit{Byte} + 56\ \textit{Byte} \tag{4.2}$$

[24] Da ATM-Zellen eine festgelegte Länge haben, kann prinzipiell auch der durch den Zell-Kopf anfallende "Overhead" unabhängig von der Länge der transportierten Nachrichten berechnet werden. Weil der "Overhead" auf ATM-Schicht jedoch leicht bei der Funktion des auf Anwendungsebene zur Verfügung stehenden Datendurchsatzes berücksichtigt werden kann, soll nur kurz erwähnt werden, daß in einem OC-3c basierten ATM-Netz ein "Overhead" von 14,13 MBit/s für die Übertragung der 5 Byte langen ATM-Zell-Köpfe verbraucht wird. Zusammen mit dem "Overhead" bei SONET von 5,76 MBit/s sind von den ursprünglichen 155,520 MBit/s bereits 19,89 MBit/s – etwa 13% – verbraucht.

[25] Die in die folgende Berechnung des maximalen Durchsatzes über CLIP einfließenden Parameter sind in der Tabelle fett dargestellt. Zusätzlich sind die für jede Schicht alternativ zur Verfügung stehenden Protokolle aufgeführt.

[26] Für UDP ergibt sich entsprechend: "$x + 8 + 20 + 8 + 8 = x + 44$ Byte"

In jeder Zelle können 48 Byte Nutzdaten übertragen werden. Die benötigte Anzahl der Zellen pro Nachricht ergibt sich durch Aufrunden. Durch dieses Aufrunden wird das "Padding" auf der AAL-5-Schicht [Peterson et al. 96, S. 184f] in der Gleichung simuliert.

$$Zellen\ pro\ Nachricht = \left\lceil \frac{x\ Byte + 56\ Byte}{48 \frac{Byte}{Zelle}} \right\rceil \tag{4.3}$$

Multipliziert mit der Zell-Gesamtlänge von 53 Byte ergibt sich die Anzahl der Byte auf der ATM-Schicht.

$$Byte\ pro\ Nachricht_{ATM} = \left\lceil \frac{x\ Byte + 56\ Byte}{48 \frac{Byte}{Zelle}} \right\rceil * 53 \frac{Byte}{Zelle} \tag{4.4}$$

In einem STS-3c Netz wurde die Nettoübertragungsrate bereits mit $18,720 * 10^6 Byte/s$ berechnet (s. Tab. 4.1). Dividiert durch die Anzahl der Byte pro Nachricht ergibt sich die Anzahl der Nachrichten pro Sekunde.

$$Nachrichten\ pro\ Sekunde = \frac{18,720 * 10^6\ Byte/s}{\left\lceil \frac{x\ Byte + 56\ Byte}{48 \frac{Byte}{Zelle}} \right\rceil * 53 \frac{Byte}{Zelle}} \tag{4.5}$$

Wird (4.5) mit der Nutzdatenlänge von $x\ Byte$ multipliziert, erhält man die Nettoübertragungsrate in Byte pro Sekunde.

$$Netto\ Byte/s = x\ Byte \frac{18,720 * 10^6\ Byte/s}{\left\lceil \frac{x\ Byte + 56\ Byte}{48 \frac{Byte}{Zelle}} \right\rceil * 53 \frac{Byte}{Zelle}} \tag{4.6}$$

Über den Faktor $\frac{8}{1.000.000}$ kann dieser Wert in die gebräuchlichere Angabe MBit/s umgerechnet werden:

$$Netto\ TCP\ MBit/s = \frac{8 \frac{Bit}{Byte}}{1.000.000} * x\ Byte \frac{18,720 * 10^6\ Byte/s}{\left\lceil \frac{x\ Byte + 56\ Byte}{48 \frac{Byte}{Zelle}} \right\rceil * 53 \frac{Byte}{Zelle}} \tag{4.7}$$

Da der UDP-Header nur 8 statt 20 Byte lang ist, summiert sich die Gesamtlänge des "Protokoll-Overheads" nur auf 44 Byte (statt 56 Byte bei TCP).

$$Netto\ UDP\ MBit/s = \frac{8\frac{Bit}{Byte}}{1.000.000} * x\ Byte \frac{18,720 * 10^6\ Byte/s}{\left\lceil \frac{x\ Byte + 44\ Byte}{48\frac{Byte}{Zelle}} \right\rceil * 53\frac{Byte}{Zelle}} \tag{4.8}$$

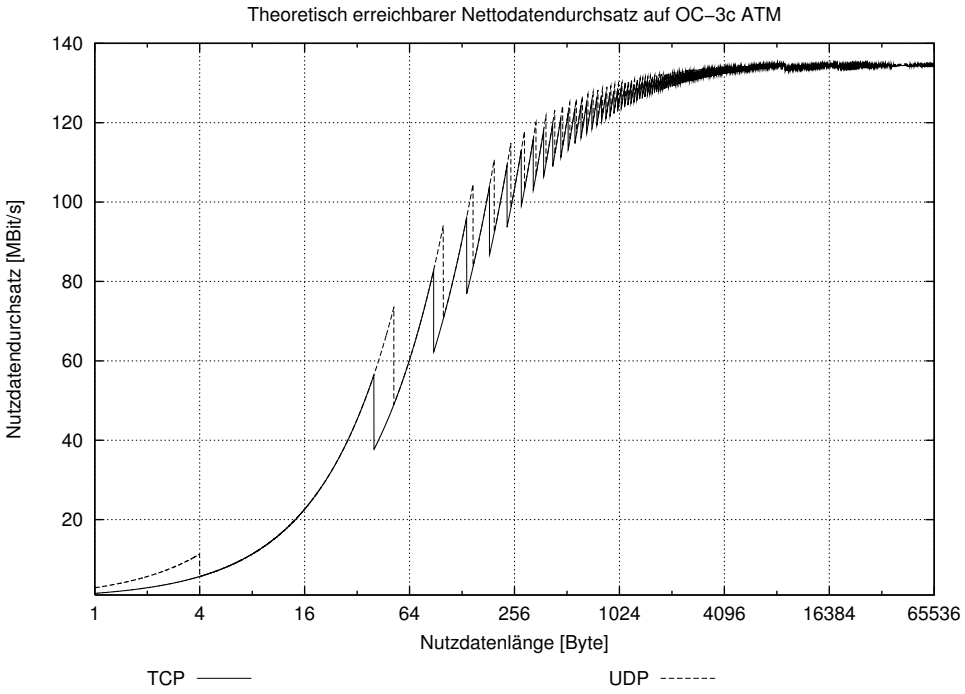


Abbildung 4.7: Theoretisch erreichbarer Durchsatz

Diese Gleichung gilt jedoch nur bis zum Erreichen der "Maximum Transfer Unit" (MTU), die bei "Classical IP" [Laubach 94] 9.180 Byte beträgt.[27] Wenn die Nutzdatenlänge zusammen mit dem IP- und dem TCP- oder UDP-Header die MTU überschreitet, muß die Nachricht — das TCP-Segment oder das UDP-Datagramm — von IP fragmentiert werden. Dabei wird die Nachricht in mehrere Teile aufgebrochen, die jeweils die MTU nicht überschreiten. Jedes Fragment erhält einen eigenen IP-Header. Pro Fragment entsteht ein zusätzlicher "Overhead" von $20\,Byte$ IP $+8\ Byte$ SNAP $+8\ Byte$ AAL-5 $= 36\ Byte$. Der TCP- bzw. UDP-Header ist nur im ersten Fragment enthalten. Alle Fragmente mit Ausnahme des letzten Fragmentes werden bei diesem Verfahren an der MTU-Länge (9.180 Byte) ausgerichtet. Ungünstigerweise ist für Fragmente bei MTU-Länge zusätzlich 20 Byte

[27] Die Vorgabe aus RFC-1577 wird in der Regel eingehalten, der RFC läßt jedoch auch das Aushandeln abweichender MTUs zu.

"Padding" (AAL-5) erforderlich, um den AAL-5-Trailer auf das Ende der letzten Zelle auszurichten. Wenn dieses "Padding" durch die Wahl einer günstigen MTU-Größe, die besser auf die Zell-Länge von 48 Byte angepaßt ist, vermieden worden wäre, hätte durch Fragmentierung bei sehr großen Nutzdatenlängen ein höherer Durchsatz erreicht werden können. Der Vorteil, außer im ersten Fragment keinen TCP-Header (20 Byte) zu benötigen und somit 20 Byte zu sparen, wird daher durch das zusätzliche "Padding" von 20 Byte wieder aufgehoben. Im Bereich oberhalb der MTU-Länge kann durch Fragmentierung somit kein höherer Durchsatz erreicht werden.

Die Fragmentierung von IP-Datagrammen hat sich in der Praxis als ungünstig herausgestellt [Kent et al. 87], da der Verlust eines Fragmentes die übrigen Fragmente wertlos macht. Daher sorgt bereits TCP dafür, daß keine Segmente generiert werden, die die MTU-Größe überschreiten würden.[28] Größere Nachrichten teilt TCP bereits in mehrere TCP-Segmente auf, so daß IP keine Fragmentierung vor dem Senden durchführen muß.

Die Formeln zur Berechnung des maximal erreichbaren Datendurchsatzes (4.7 und 4.8) werden im Fall von UDP um die Auswirkungen der Fragmentierung ergänzt. Bei TCP wird von der Generierung zusätzlicher TCP-Segmente ausgegangen. Im folgenden wird die Funktion erweitert, um den bei TCP durch zusätzliche Segmente entstehenden "Overhead" berücksichtigen zu können.

Bei der MTU-Größe von 9.180 Byte können pro TCP-Segment maximal 9.140 Byte übertragen werden. Wenn die Nachricht in n Segmenten übertragen werden muß, erreichen $n-1$ Segmente die MTU-Größe von 9.180 Byte, die jeweils in 192 Zellen[29] übertragen werden.

$$Zellen_{Seg_{1\ldots(n-1)}} = \left(\left\lceil \frac{x\ Byte}{9140 \frac{Byte}{Segment}} \right\rceil - 1 \right) * 192 \frac{Zellen}{Segment} \tag{4.9}$$

Die Länge des letzten Segmentes beträgt dann x abzüglich des in den Segmenten 1 – (n-1) übertragenen Teils der Nachricht. Nach der Division durch die 48 Byte, die pro Zelle übertragen werden können, erhalten wir die Anzahl der Zellen des letzten Segmentes durch Aufrunden.

$$Zellen_{Seg_n} = \left\lceil \frac{x\ Byte + 56\ Byte - \left(\left\lceil \frac{x\ Byte}{9140 \frac{Byte}{Segment}} \right\rceil - 1 \right) * 9140 \frac{Byte}{Segment}}{48 \frac{Byte}{Zelle}} \right\rceil \tag{4.10}$$

[28] Die MSS wird auf einen Wert gesetzt, der 40 Byte unterhalb der MTU liegt.

[29] Berechnet durch Einsetzen in Gleichung 4.3.

Die Summe beider Gleichungen ergibt die für die Übertragung der Nachricht benötigte Anzahl der Zellen. Eingesetzt in die vorher entwickelte Annäherung (Gleichung 4.6) ergibt sich:

$$Netto\ Byte/s = \frac{18,720 * 10^6\ Byte/s * x\ Byte}{\left(Zellen_{Seg_{1\ldots(n-1)}} + Zellen_{Seg_n}\right) * 53\frac{Byte}{Zelle}} \tag{4.11}$$

$$Netto\ TCP\ MBit/s = \frac{18,720 * 10^6\ Byte/s * \frac{8\frac{Bit}{Byte}}{1.000.000} * x\ Byte}{\left(Zellen_{Seg_{1\ldots(n-1)}} + Zellen_{Seg_n}\right) * 53\frac{Byte}{Zelle}} \tag{4.12}$$

Bei UDP wird von einer Fragmentierung ausgegangen. Daher enthält nur das erste Fragment einen UDP-Header. Es addieren sich daher zur Nachrichtenlänge von x Byte einmalig 8 Byte für den UDP-Header. In jedem Fragment können 9160 Byte dieses UDP-Datagramms übertragen werden. Die Formel für den theoretisch erreichbaren Durchsatz über UDP ergibt sich daher wie folgt:

$$Zellen_{UDP,frag_{1\ldots(n-1)}} = \left(\left\lceil \frac{x\ Byte + 8\ Byte}{9160\frac{Byte}{Fragment}} \right\rceil - 1\right) * 192\frac{Zellen}{Fragment} \tag{4.13}$$

$$Zellen_{UDP,frag_n} = \left\lceil \frac{x\ Byte + 44\ Byte - \left(\left\lceil \frac{x\ Byte + 8\ Byte}{9160\frac{Byte}{Fragment}} \right\rceil - 1\right) * 9160\frac{Byte}{Fragment}}{48\frac{Byte}{Zelle}} \right\rceil \tag{4.14}$$

$$Netto\ UDP\ MBit/s = \frac{18,720 * 10^6\ Byte/s * \frac{8\frac{Bit}{Byte}}{1.000.000} * x\ Byte}{\left(Zellen_{UDP,frag_{1\ldots(n-1)}} + Zellen_{UDP,frag_n}\right) * 53\frac{Byte}{Zelle}} \tag{4.15}$$

Der maximale Durchsatz kann wegen des notwendigen "Paddings" (wie bereits erwähnt) selbst bei Fragmentierung nicht oberhalb der MTU-Größe erreicht werden. Es ist daher plausibel anzunehmen, daß der maximale Durchsatz bei Erreichen der MTU-Größe erzielt wird. Bei Betrachten der Funktion des theoretisch erreichbaren Durchsatzes in Abhängigkeit von der Nutzdatenlänge (Abb. 4.7) fällt

der deutliche Abfall nach Überschreiten jeder 48 Byte Zell-Grenze auf. Dieser Effekt macht sich zwar bei größeren Nachrichten nicht mehr so stark bemerkbar, bleibt aber erkennbar. Daher ist auch die nächstkleinere Nutzdatenlänge unterhalb der MTU, die sich ohne "Padding" auf 48 Byte Zellen übertragen läßt, ein möglicher Wert, bei dem das Maximum erreicht werden könnte. Das Maximum wird durch Einsetzen der beiden Nutzdatenlängen in die Gleichungen 4.12 beziehungsweise 4.15 ermittelt:

- Bei einer Nutzdatenlänge von 9.140 Byte (TCP) bzw. 9.152 Byte (UDP) wird die MTU-Länge erreicht. Zur Übertragung einer Nachricht werden 192 Zellen benötigt (20 Byte "Padding"). Der maximale Durchsatz beträgt hier 134,51 MBit/s (TCP) bzw. 134,69 MBit/s (UDP).
- In 191 Zellen lassen sich ohne zusätzliches "Padding" bei TCP 9.112 Byte, bei UDP 9.124 Byte versenden. Der maximale Durchsatz beträgt 134,80 MBit/s (TCP) bzw. 134,98 MBit/s (UDP).

Bei der nächstkleineren Nutzdatenlänge unterhalb der MTU-Größe, die sich ohne "Padding" auf 48 Byte Zellen verteilen läßt, kann demnach theoretisch ein geringfügig höherer Durchsatz erreicht werden als exakt bei Erreichen der MTU-Größe. Dieses Ergebnis wird auch in dem Ausschnitt eines Plots der Funktion anschaulich (s. Abb. 4.8). Wie groß der Einfluß des "Paddings" selbst bei großen Dateneinheiten noch ist, zeigt sich auch daran, daß selbst bei acht KByte langen Nachrichten mit 134,58 MBit/s noch ein Durchsatz oberhalb des Durchsatzes bei Dateneinheiten mit MTU-Größe erreicht wird.

Die Abbildungen 4.7, 4.8 und 4.9 wurden mit den Funktionen 4.12 und 4.15 berechnet. Die Ausschnittsvergrößerung in Abbildung 4.9 zeigt den Effekt der Fragmentierung bei UDP und der Segmentierung bei TCP auf den maximal erreichbaren Datendurchsatz.

Die Standards für "Classical IP over ATM" [Laubach 94, Perez et al. 95] erlauben es "Classical IP"-Knoten, beim Aufbau eines SVCs eine MTU-Größe auszuhandeln. Die MTU-Größe wird dann nur noch durch die maximal in einem AAL-5-Rahmen übertragbare Nutzdatenlänge (65.535 Byte) begrenzt. Abzüglich des SNAP-Headers von 8 Byte Länge ergibt sich eine maximale MTU-Größe von 65.527 Byte. Unter Berücksichtigung des "TCP-Overheads" (20 Byte) und des "IP-Overheads" (20 Byte) ergibt sich eine maximale Nutzdatenlänge von 65.487 Byte für TCP- und 65.499 Byte für UDP-Dateneinheiten. Eingesetzt in Gleichung 4.7 läßt sich der theoretisch erreichbare Durchsatz auf einer TCP-Verbindung auf 135,46 MBit/s berechnen. Für UDP (8 Byte Header) ergeben sich nach Gleichung 4.8 entsprechend 135.49 MBit/s. Diese Werte haben aus zwei Gründen nur eine beschränkte Bedeutung. Die vergrößerten MTU-Größen werden nicht von

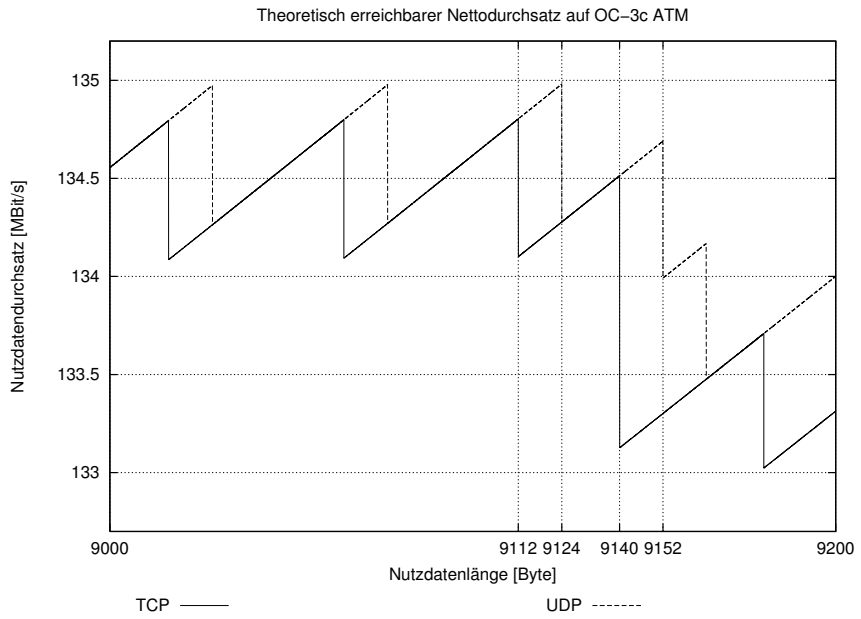


Abbildung 4.8: Theoretisch erreichbarer Durchsatz (Ausschnitt)

allen Implementationen unterstützt. Außerdem wird die Übertragung von ATM-Zellen nicht garantiert. Bei Verlust nur einer Zelle kann das IP-Datagramm nicht mehr vom Empfänger reassembliert werden und höhere Protokollschichten (TCP) müssen eine Wiederholung der Übertragung des gesamten IP-Datagramms sicherstellen.[30] Werden bei der Standard-MTU-Größe von 9.180 Byte nur 192 Zellen benötigt, sind bei der MTU-Größe von 65.527 Byte bereits 1.366 Zellen erfor-

[30] Da über TCP nur dann hohe Durchsätze erzielt werden, wenn mehrere TCP-Segmente gesendet werden können, bevor der Empfang bestätigt werden muß, sind bei großer MTU entsprechend große Sende- und Empfangspuffer ($\gg$ 256 KByte) erforderlich.

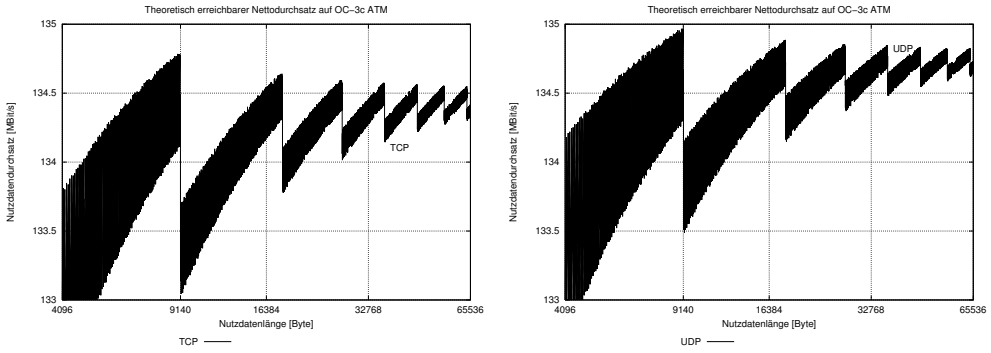


Abbildung 4.9: Effekt von Segmentierung und Fragmentierung auf den Durchsatz

derlich. Da für den Empfang eines IP-Datagramms alle Zellen unversehrt beim Empfänger eintreffen müssen, steigt bei großen MTU-Größen die Wahrscheinlichkeit, daß das IP-Datagramm nur unvollständig beim Empfänger ankommt. Die durch TCP ausgelöste Wiederholung der Übertragung reduziert den maximal erreichbaren Durchsatz. Daher wäre es in der Praxis empfehlenswert, die MTU-Größe abhängig von der aktuell gemessenen Häufigkeit des Zellverlustes zu wählen. Die Performanzauswirkungen einer vergrößerten MTU werden im folgenden nicht weiter verfolgt.[31]

4.2.2.6 Vergleich gemessener und theoretisch erreichbarer Datendurchsätze

Die Qualität der Protokoll-Implementation zeigt der Vergleich der gemessenen Werte mit dem theoretisch berechneten maximalen Durchsatz (s. Abschnitt 4.2.2.5). Die Abbildung 4.10 enthält den für eine Socket-Größe von 262.144 Byte ermittelten Durchsatz[32] und die bereits aus Abbildung 4.7 bekannte Funktion des theoretisch erreichbaren Durchsatzes. Während der gemessene Durchsatz für kurze Nachrichten deutlich unter dem theoretischen Maximum bleibt, wird der berechnete Wert bei den Nutzdatenlängen von 1.536 und 2.048 Byte (134,52 vs. 132,03 MBit/s) deutlich überschritten. Die Ausschnittsvergrößerung in Abbildung 4.11 zeigt sogar, daß auch der Durchsatz für 4.096 Byte lange Schreibzugriffe noch das theoretische Maximum überschreitet. Ursache ist das Aggregieren mehrerer kleiner Schreibzugriffe zu einem großen TCP-Segment. Diese Optimierung ist möglich, da die Anwendung keine direkte Möglichkeit hat, die Generierung von Datagrammen einer bestimmten Länge zu veranlassen (vgl. S. 220). Obwohl das Meßwerkzeug durch einzelne Schreibzugriffe 2.048 Byte lange Nachrichten erzeugt, haben 80% der schließlich gesendeten TCP-Segmente eine Länge von 8.192 Byte. Jeweils etwa 10% der Segmente haben eine Länge von 2.048 und 4.096 Byte.[33] Bei dieser Nutzdatenlängenverteilung wäre nach der Gleichung 4.12 ein theoretischer Durchsatz von über 134 MBit/s erreichbar, was dem gemessenen Wert entspricht. Durch das Reduzieren der MTU-Größe wird die Aggregation mehrerer Schreibzugriffe zu einem größeren TCP-Segment verhindert. Die Meß-

[31] In Netzen, die nicht ausschließlich auf ATM basieren, wird die MTU-Größe in den meisten Fällen deutlich kleiner als die Standard-MTU-Größe für CLIP von 9.180 Byte sein, da die MTU-Größe durch "Path-MTU-Discovery" auf das Netz mit der geringsten MTU angepaßt wird.

[32] Die Socket-Größe ist ein Parameter der TCP-Realisierung, er beeinflußt den *theoretisch* erreichbaren Durchsatz nicht. Die Socket-Größe von 262.144 Byte wurde für den Vergleich gewählt, da diese keine Ausreißer wie zum Beispiel bei der Socket-Größe von 32.768 Byte zeigt.

[33] Diese Verteilung wurde mit Hilfe von `atmsnoop` ermittelt, einem "Sniffer", der die während einer Messung übertragenen Dateneinheiten aufzeichnet.

werte bleiben dann wie erwartet auch für 1.536 und 2.048 Byte Schreibzugriffe unterhalb des berechneten theoretischen Maximums.[34]

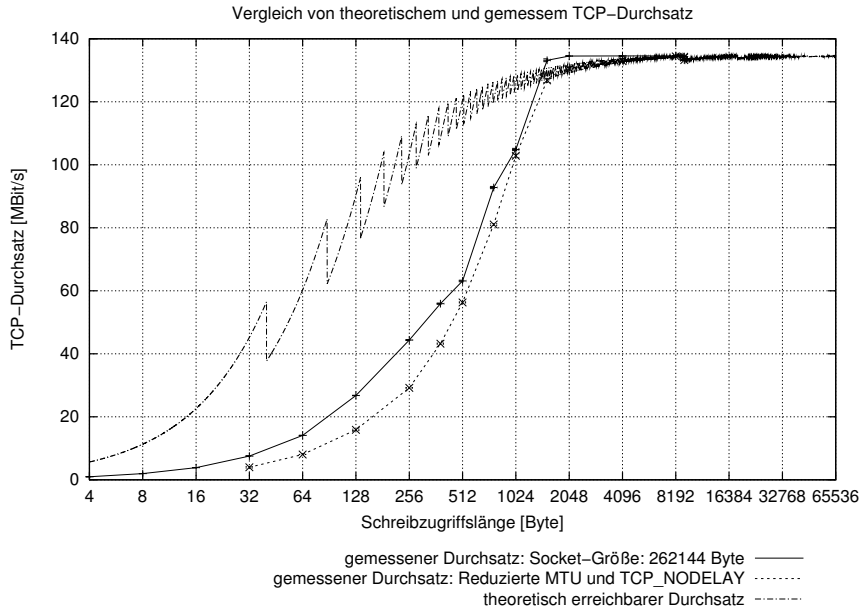


Abbildung 4.10: Vergleich mit theoretisch erreichbarem Durchsatz

4.2.2.7 Datagrammdurchsatz

Das deutliche Unterschreiten des theoretisch erreichbaren Durchsatzes bei kleinen Nutzdatenlängen hängt mit dem höheren Verarbeitungsaufwand für die kleineren Nutzdatenlängen zusammen. Da der "Overhead" für das Erzeugen der Protokoll-Header für kurze und lange Nachrichten etwa gleich ist, wird es einen Grenzwert für die maximale Anzahl der Datagramme geben, die das System pro Sekunde erzeugen kann. Vermutlich wird für die kleineren Nutzdatenlängen die maximale Anzahl der Datagramme erreicht, die die Workstation pro Sekunde verarbeiten kann. Die Anzahl der bei der Messung übertragenen Datagramme kann aus der Länge der Datagramme und dem erreichten Durchsatz berechnet werden:[35]

[34] Wenn nur die `TCP_NODELAY`-Option benutzt, aber auf das Reduzieren der MTU verzichtet wird, überschreiten die Meßwerte für 2.048 und 4.096 Byte Schreibzugriffslänge das berechnete, theoretische Maximum.

[35] Zu beachten ist, daß der Datagrammdurchsatz hier indirekt über den gemessenen Datendurchsatz bestimmt wird. Die Gleichung 4.16 berücksichtigt nur die für die Übertragung der Nutzdaten generierten Datagramme. Nicht enthalten sind die empfangene Bestätigung (ACK) und aufgrund von Übertragungsfehlern mehrfach zu sendende Datagramme.

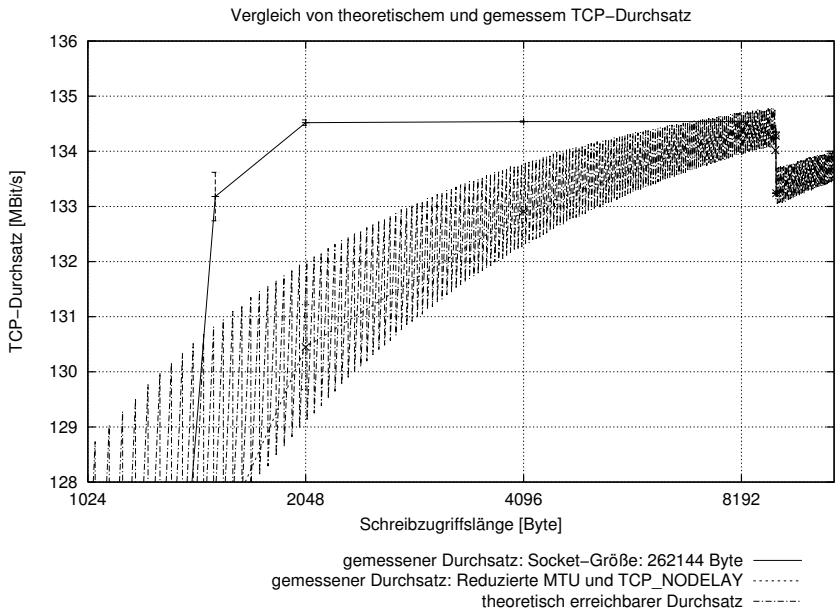


Abbildung 4.11: Vergleich mit theoretisch erreichbarem Durchsatz (Ausschnitt)

$$Datagrammdurchsatz = \frac{Datendurchsatz\ in\ Bit/s}{8\ \frac{Bit}{Byte} * Nutzdatenlänge\ in\ Byte} \tag{4.16}$$

Die Abbildung 4.12 ergänzt die Meßergebnisse aus Abbildung 4.10. Zusätzlich wurde für jeweils beide Kurven die Anzahl der pro Sekunde verarbeiteten Datagramme (Funktion 4.16) berechnet. In der Abbildung 4.12 ist zu erkennen, daß der Grenzdurchsatz der gemessenen TCP-Verbindung knapp unterhalb von 16.000 Datagrammen pro Sekunde liegt. Zum Erreichen des theoretisch berechneten, maximalen Durchsatzes müßten bei kurzen Nachrichten jedoch etwa 180.000 Datagramme pro Sekunde[36] verarbeitet werden. Betrachtet man beginnend bei großen Schreibzugriffslängen den Verlauf der Funktion der Datagramme pro Sekunde, so zeigt sich, daß bei einer Schreibzugriffslänge von 2.000 Byte die maximale Anzahl der Datagramme erreicht ist, die das System verarbeiten kann. Ab diesem Punkt steigt der Datagrammdurchsatz nur noch geringfügig, da mit kürzeren Datagrammen der Verarbeitungsaufwand in den Systemen etwas abnimmt.[37] Bei dieser Schreibzugriffslänge knickt auch die Kurve des gemessenen Durchsatzes ab. Dies bestätigt die ursprüngliche Vermutung, daß für kleine

[36] Nach Funktion 4.16 sind 176.603 TCP-Segmente pro Sekunde notwendig, um auf einem OC-3c ATM-Netz über "Classical IP" mit geringen Nutzdatenlängen ($\leq$ 40 Byte) den theoretisch möglichen Datendurchsatz zu erreichen.

[37] Es müssen geringere Datenmengen innerhalb der Systeme kopiert werden.

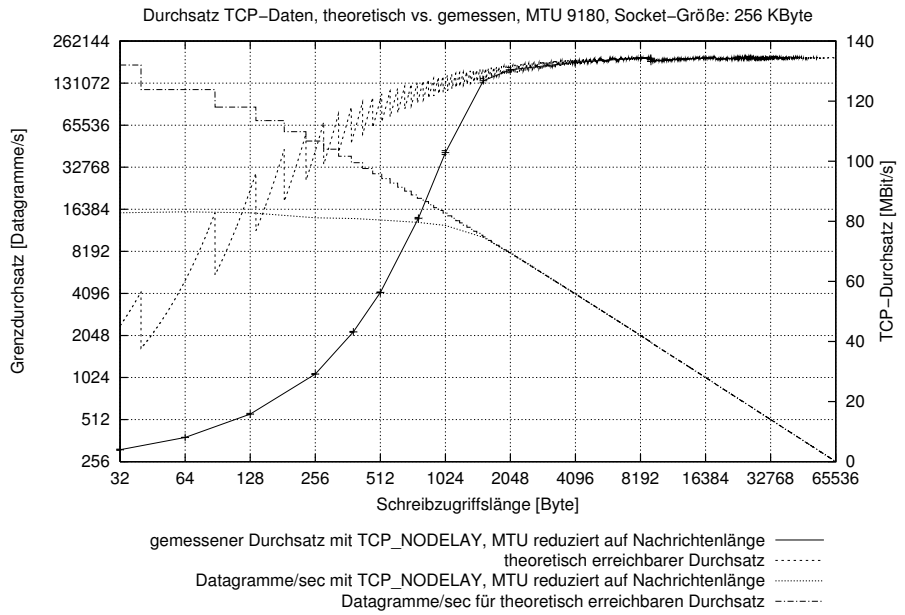


Abbildung 4.12: Datagrammdurchsatz

Schreibzugriffslängen der Grenzdurchsatz an Datagrammen pro Sekunde erreicht wird.

Der errechnete Grenzdurchsatz der Datagramme pro Sekunde wurde durch zwei weitere Messungen ergänzt, bei denen der Durchsatz einer TCP-Verbindung auf Ethernet und einer TCP-Verbindung nach `localhost`[38] ermittelt wurde. Die berechneten Grenzdurchsätze werden in Abbildung 4.13 miteinander verglichen.

Die Messung über das ATM-Netz erreichte den höchsten Durchsatz mit knapp 16.000 Datagrammen pro Sekunde. Bei der "Localhost"-Messung wird demgegenüber nur etwa der halbe Durchsatz erreicht, da sowohl der Sende- als auch der Empfangsprozeß auf demselben System um die Ressourcen konkurrieren. Der nur zu erahnende Abfall der Datagramme pro Sekunde im Bereich oberhalb von 4.096 Byte langen Schreibzugriffen setzt sich auch oberhalb von 8.192 Byte weiter fort (ohne Abbildung). Der Sprung bei Nutzdatenlängen, die nur geringfügig größer als 8.192 Byte sind, beruht jedoch auf der Überschreitung der MTU-Größe, die bei "Localhost" mit 8.232 Byte festgelegt ist.

Die Messung für Ethernet unterscheidet sich von den beiden anderen Messungen, da hier im gesamten Meßbereich das Netz der begrenzende Faktor ist und nicht der Durchsatz der Workstation. Daher wird auch bei kleineren Nachrichten nicht

[38] "Localhost" (IP-Adresse: `127.0.0.1`) ist eine Pseudo-IP-Adresse, über die jeder IP-Knoten auf seine eigenen Netzwerkdienste zugreifen kann, ohne daß Datagramme über eine physikalische Netzwerkschnittstelle gesendet werden müssen.

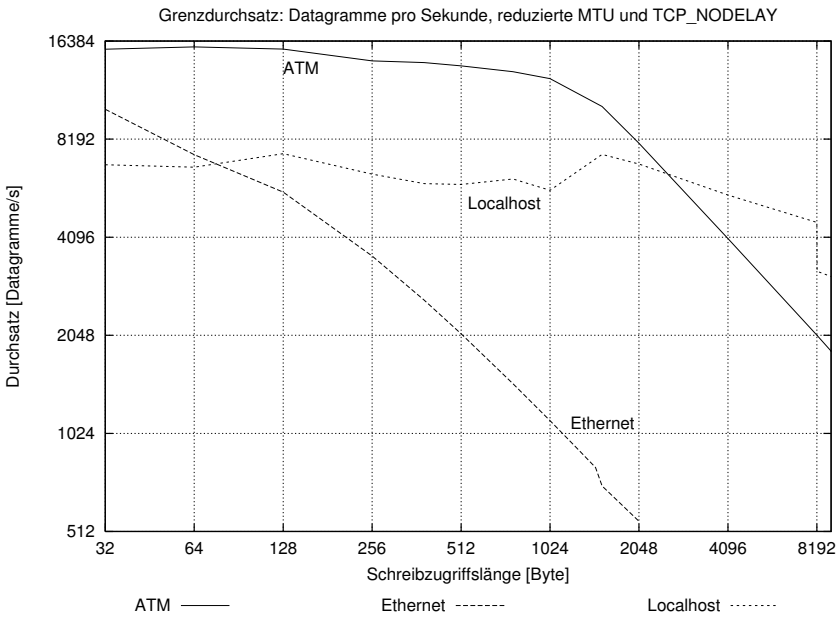


Abbildung 4.13: Datagrammdurchsatz bei unterschiedlichen Netzen

der maximale Durchsatz an Datagrammen erreicht. Ob auch über Ethernet bei Schreibzugriffen, die kürzer als 32 Byte sind, der maximale Durchsatz an Datagrammen pro Sekunde erreicht wird, kann nicht ermittelt werden, da sich die MTU-Größe der Test-Systeme nicht unter 72 Byte verringern läßt.[39] Meßpunkte unterhalb von 32 Byte Schreibzugriffslänge enthalten daher immer Aggregationseffekte und können somit nicht für die Berechnung der Datagramme pro Sekunde herangezogen werden.

Diese Werte lassen schon erwarten, daß Firewalls besonders bei kürzeren Datagrammlängen einen Engpaß darstellen werden. Den maximalen Durchsatz bei großen Datagrammlängen zu erreichen, wird für Firewalls weniger ein Problem sein. Vorsicht ist also geboten bei Aussagen, daß ein Firewall "Wire-Speed" erreicht — bei 8 KByte Schreibzugriffen wird für einen Durchsatz von 134 MBit/s nur ein geringer Datagrammdurchsatz von etwa 2.000 Datagrammen pro Sekunde benötigt.

[39] Die minimale SDU-Größe beträgt bei Ethernet 46 Byte, so daß theoretisch minimal 6 Byte lange Schreibzugriffe — mit 20 Byte IP- und 20 Byte TCP-Header ergeben sich 46 Byte — möglich wären.

4.2.2.8 Einsatz leistungsfähigerer Workstations

Wie bereits gezeigt wurde, ist das Nichterreichen des theoretisch möglichen Durchsatzes bei kleineren Nachrichtenlängen auf die begrenzte Leistungsfähigkeit der eingesetzten Workstations zurückzuführen. Für spätere Messungen der Performanz von verschiedenen Firewall-Konzepten ist es wichtig, eine entsprechend hohe Datagrammlast erzeugen zu können, um auch im Bereich der geringeren Nachrichtenlängen den theoretisch erreichbaren Durchsatz zu erzielen. Mit zwei "Sun Ultra-2/300"-Workstations wurden zwei Systeme beschafft, die mit jeweils zwei 300 MHz CPUs etwa eine vierfache Leistungsfähigkeit[40] im Vergleich zu den "Ultra-1"-Systemen mit einem 143 MHz Prozessor erzielen. Die "Ultra-2"-Systeme verfügen über 622 MBit/s ATM-Schnittstellen, die zudem auch die TCP-Prüfsumme in der Hardware berechnen können. Um eine Vergleichbarkeit der Meßergebnisse zu gewährleisten, kann der Durchsatz der Schnittstelle auch auf niedrigere Werte konfiguriert werden.

Der Vergleich der Meßkurven einer "Ultra-1" mit einer "Ultra-2" (s. Abb. 4.14) zeigt im Bereich der kleineren Nachrichtenlängen deutlich den Durchsatzgewinn durch den Einsatz leistungsfähigerer Workstations. Bei Nachrichtenlängen ab 512 Byte kann fast der theoretisch mögliche Durchsatz erreicht werden. Der Bereich um 512 Byte Nachrichtenlänge ist für Messungen der Firewall-Performanz wichtig, da auch die durchschnittliche Nachrichtenlänge im Internet etwa 500 Byte beträgt.[41]

Der berechnete Datagrammdurchsatz (s. Abb. 4.15) zeigt eine Verdoppelung im Vergleich zur "Ultra-1". Die durch den zweiten Prozessor prinzipiell mögliche vierfache Leistung gegenüber der "Ultra-1" kann bei dieser Messung nicht erreicht werden.

In Abbildung 4.16 ist zum Vergleich auch noch die Meßkurve bei einem Schnittstellendurchsatz von 622 MBit/s eingezeichnet. Im Gegensatz zu den 155 MBit/s Messungen, bei denen der theoretisch mögliche Durchsatz bei großen Nachrichtenlängen oberhalb von 1.500 Byte erreicht werden kann, bleibt der gemessene Durchsatz mit 510 MBit/s deutlich unterhalb der theoretisch möglichen 538 MBit/s. Bei Nachrichtenlängen unterhalb von 512 Byte überdecken sich die Kurven für die Netzdurchsätze von 622 MBit/s und 155 MBit/s, was bestätigt, daß in diesem Bereich der Durchsatz allein durch die Leistungsfähigkeit der Workstations begrenzt wird.

---

[40] Die Schwierigkeit, die Last auf die parallelen Prozessoren zu verteilen, wird zunächst nicht weiter betrachtet.

[41] Siehe `http://www.nlanr.net/NA/learn/packetsizes.html`

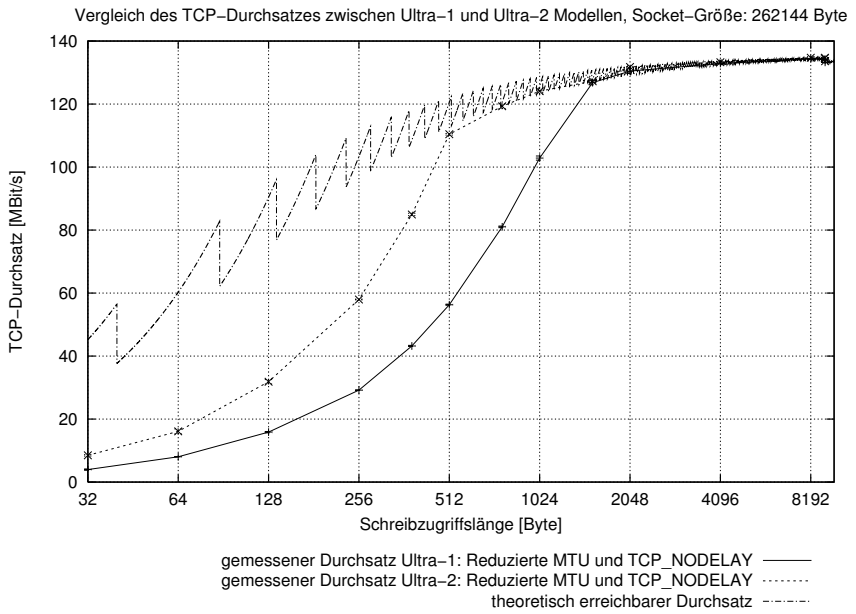


Abbildung 4.14: Durchsatz bei leistungsfähigeren Workstations

4.2.2.9 Schreibzugriffslängen oberhalb der MSS

"Maximum Segment Size" (MSS) bezeichnet die maximale Nutzdatenlänge, die in einem TCP-Segment übertragen werden kann. TCP reduziert die MSS auf die MTU-Größe abzüglich 40 Byte für IP- und TCP-Header. Für Nutzdaten, die größer als die MSS sind, müssen zusätzliche TCP-Segmente generiert werden.

Die Ausschnitte in Abbildung 4.17 zeigen sehr deutlich den Effekt bei Schreibzugriffen, die die MSS (bzw. ganzzahlige Vielfache der MSS) überschreiten. Der gemessene Einbruch tritt an der vorab berechneten Sprungstelle ein. Jedoch sind die Performanzeinbußen deutlich höher, als die berechneten Werte erwarten lassen. Daher ist davon auszugehen, daß die Verarbeitung von Schreibzugriffen mit Längen oberhalb der MSS für das System vergleichsweise aufwendig sein muß. Da wie bereits erwähnt, kaum eine Anwendung Schreibzugriffe von einer Länge oberhalb von 8 KByte durchführt, ist offensichtlich die Protokoll-Implementation ebenfalls auf den Bereich unterhalb von 8 KByte optimiert. Der Bereich oberhalb von 8 KByte Schreibzugriffen zeigt daher ein nicht so günstiges Verhalten. Bei genauer Analyse der gemessenen Werte ist sogar erkennbar, daß bei der Schreibzugriffslänge von 9.140 Byte (MSS) bereits ein geringerer Durchsatz erzielt wird als bei den Schreibzugriffslängen von 4 und 8 KByte.

Bei späteren Firewall-Tests ist der Bereich mit Schreibzugriffslängen oberhalb der MSS praktisch bedeutungslos, da auch bei diesen Schreibzugriffen immer

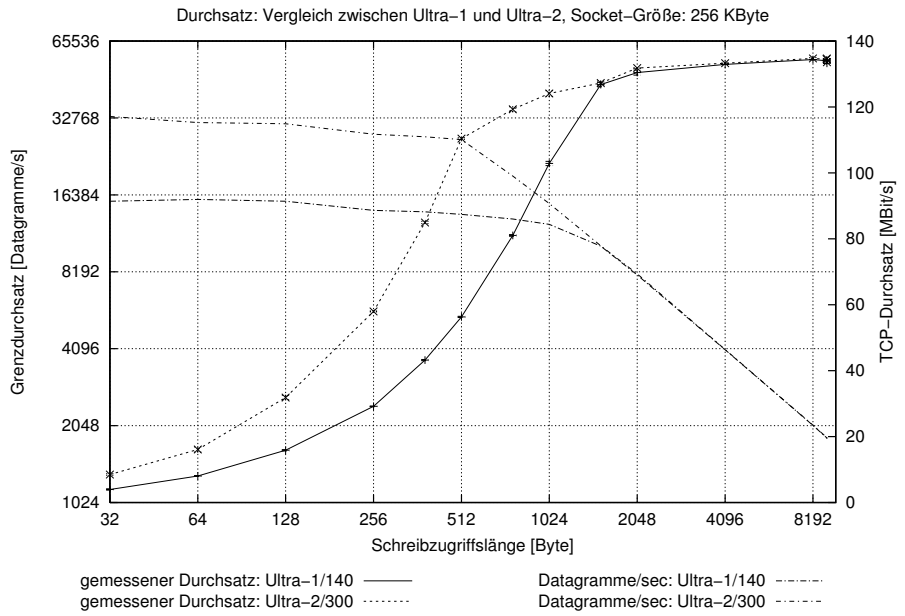


Abbildung 4.15: Datagrammdurchsatz bei "Ultra-2/300"

TCP-Segmente generiert werden, die die MTU-Größe nicht überschreiten. Am Firewall kommen somit immer Dateneinheiten an, die maximal MTU-Größe erreichen. Dabei ist es für den Firewall — dies gilt für Packet Screens, Gateway-Firewalls und Kombinationen von Packet Screens mit Bastionen gleichermaßen — unerheblich, wie die Dateneinheiten generiert werden. Für Messungen der Firewall-Performanz auf Anwendungsebene sollten daher nur Schreibzugriffe im Bereich bis zum Erreichen der MTU-Größe verwendet werden. Für die Messung der Performanz auf Netzwerkebene, etwa bei Messungen des Durchsatzes von Packet Screens, sind Schreibzugriffe oberhalb der MTU-Größe ohnehin ungeeignet, da dies zu einem Gemisch aus unterschiedlich langen Dateneinheiten führt. Nur mit Schreibzugriffen, deren Länge maximal die MTU-Größe erreicht, wird ein Strom von Datagrammen fester Länge erzeugt.

4.2.2.10 "Localhost"-Messung des Datendurchsatzes

Zur Ermittlung des Grenzdurchsatzes an TCP-Segmenten pro Sekunde wurde bereits eine Messung nach "Localhost" hinzugezogen (s. Abb. 4.13). "Localhost" (IP-Adresse: `127.0.0.1`) bezeichnet eine Pseudo-IP-Adresse, über die jeder IP-Knoten auf seine eigenen Netzwerkdienste zugreifen kann, ohne daß Datagramme über eine physikalische Netzwerkschnittstelle gesendet werden müssen. Bei der Kommunikation mit "Localhost" werden wie bei Verbindungen über ein reales Netz die zu sendenden Daten in TCP-Segmente verpackt und mit IP-Headern

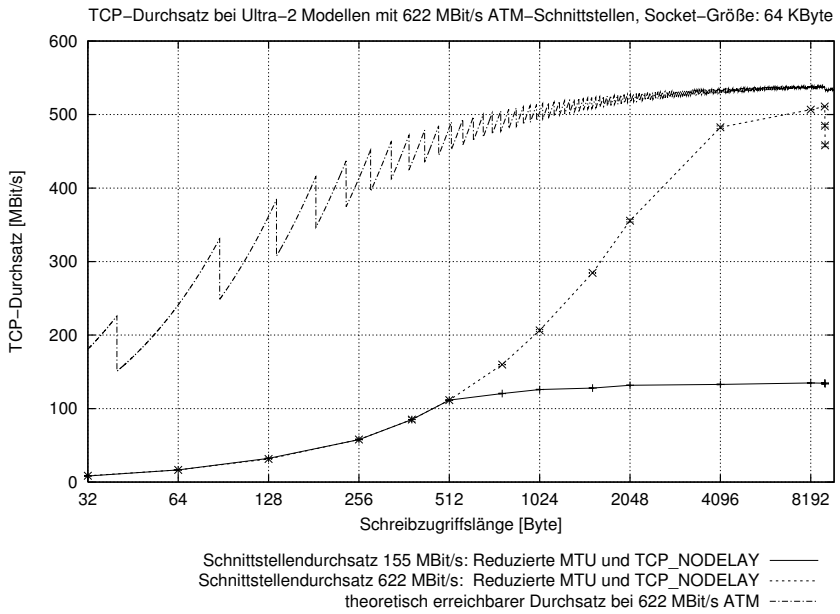


Abbildung 4.16: Durchsatz über eine 622 MBit/s ATM-Schnittstelle

versehen. Anschließend werden die IP-Datagramme jedoch nicht versendet, sondern so weiterverarbeitet, als wären sie gerade von einer Netzwerkschnittstelle empfangen worden. Durch Messungen von Verbindungen an "Localhost" entfallen daher die Durchsatzbeschränkungen des Netzes und die Performanz des IP-Hosts wird meßbar. So kann ermittelt werden, welches der maximale Durchsatz einer Workstation ist und ob Netzwerkschnittstellen mit höherem Durchsatz, etwa 622 MBit/s ATM-Schnittstellen, überhaupt ausgenutzt werden können.

Abbildung 4.18 zeigt die gemessenen Durchsätze auf TCP-Verbindungen an "Localhost" jeweils in Grundeinstellung für 8 und 32 KByte Socket-Größe. Die MTU-Größe des "Localhost-Pseudo-Interfaces" ist in der Grundeinstellung der verwendeten Solaris 2.6 Systeme auf 8.232 Byte voreingestellt. Diese MTU-Größe ermöglicht das Senden von bis zu 8.192 Byte (8 KByte) großen Nachrichten in einem TCP-Segment (20 Byte TCP- und 20 Byte IP-Protokollkopf).

Anders als bei den Messungen über ATM wird nicht bei Erreichen der MTU-Größe der maximale Durchsatz erzielt. Die Obergrenze für die Kopierleistung der in der Messung verwendeten Workstations wird bei 32 KByte Socket-Größe und Schreibzugriffen einer Länge ab 64 KByte erreicht.[42] Der gemessene Wert liegt

[42] Der bei genauerer Betrachtung erkennbare Einbruch (um etwa 5 MBit/s) im Bereich von 64 KByte Schreibzugriffslänge ist auf ungünstige Aufteilung der Daten auf TCP-Segmente zurückzuführen. Hier wurden mehrere Meßpunkte um das achtfache der MTU-Größe gruppiert.

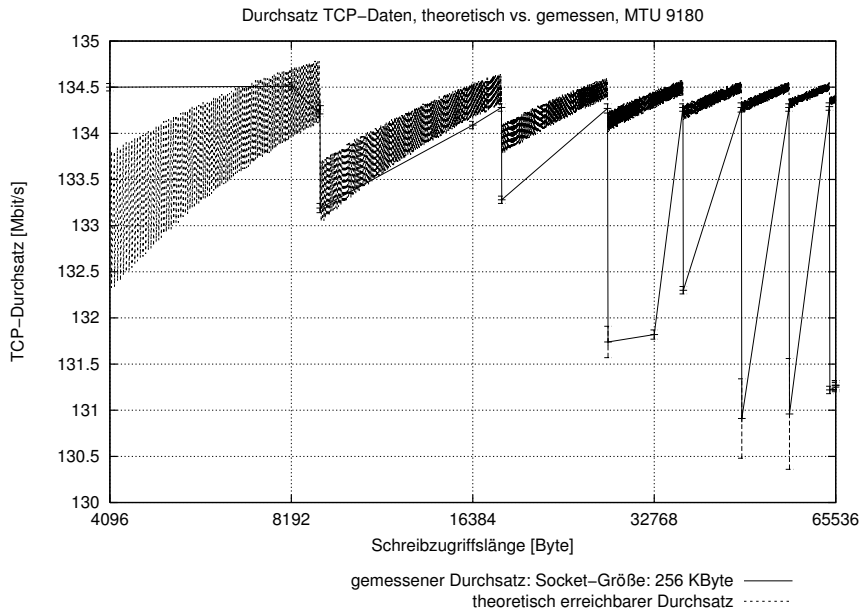


Abbildung 4.17: Vergleich mit theoretisch erreichbarem Durchsatz (Ausschnitt)

bei etwa 400 MBit/s.[43] Bei Erreichen der MTU-Größe werden demgegenüber nur etwa 300 MBit/s erreicht.[44] Diese Werte liegen etwa bei der Hälfte bis zwei Drittel des in der nächsten Skalierungsstufe von ATM (OC12c, 622 MBit/s) möglichen Durchsatzes. Aufgrund der Doppelbelastung durch Sende- und Empfangsprozeß muß die tatsächliche Sende- bzw. Empfangsleistung etwa doppelt so hoch sein wie ein gemessener Wert.

Der maximale Durchsatz wird oberhalb der MTU-Größe erreicht, weil bei "Localhost"-Messungen keine Optimierungen für ein bestimmtes Netz notwendig sind, sondern die Daten im Speicher der Workstation übertragen werden. Dabei führt natürlich jede zusätzliche Belastung des Systems zu einer Verringerung des Durchsatzes. Bei 8 statt 256 KByte Schreibzugriffen ist die 32-fache Anzahl von Schreibzugriffen des Meßwerkzeuges notwendig. Da ein Firewall neben dem Senden und Empfangen der Daten durch die Zugriffskontroll- und Überwachungsfunktionen

[43] Bei der Socket-Größe von 64 KByte konnte mit 64 und 256 KByte Schreibzugriffen sogar Durchsätze von 423 und 430 MBit/s erreicht werden. Die Meßkurve für diese Socket-Größen zeigen jedoch ein sehr ungleichmäßiges Verhalten, so daß benachbarte Meßpunkte nur Durchsätze unterhalb von 400 MBit/s erreichten und Konfidenzintervalle nicht eingehalten werden konnten. Die Erhöhung der Socket-Größe auf 256 KByte führte zu noch stärkeren Schwankungen.

[44] Der an dieser Stelle in beiden Kurven erkennbare Rückgang des Durchsatzes um etwa 50 MBit/s hängt mit einem zusätzlichen Meßpunkt zusammen, bei dem die MTU-Größe geringfügig überschritten wird. Dadurch werden bei diesem Meßpunkt 10 Byte lange TCP-Segmente erzeugt, die sich natürlich ungünstig auf den Durchsatz auswirken.

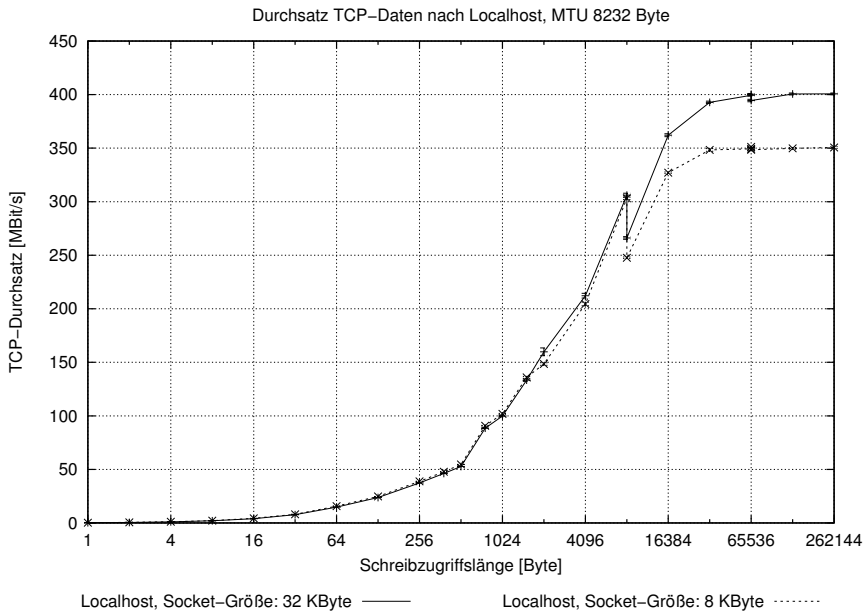


Abbildung 4.18: Maximaler Durchsatz auf Verbindungen an "Localhost"

belastet wird, ist damit zu rechnen, daß die hier gemessene Obergrenze in einem Firewall noch deutlich unterschritten wird.

4.2.2.11 Datendurchsatz bei UDP

UDP ("User Datagram Protocol") ist ein verbindungsloses, datagrammorientiertes Protokoll, das alternativ zum verbindungsorientierten TCP zur Übertragung von Daten verwendet werden kann. Im Gegensatz zu TCP gibt es bei UDP keine Flußkontrolle und keine Fehlerkorrektur, daher eignet sich UDP weniger zum Übertragen großer Datenmengen. UDP wird aus diesem Grund in erster Linie für Status- und Kontrollinformationen wie bei DNS [Mockapetris 87a, Mockapetris 87b] und Syslog [Garfinkel et al. 96] verwendet, bei denen kleine Datenmengen mit niedrigem Aufwand übertragen werden sollen. UDP kommt daher bei der Messung der Firewall-Performanz nur eine geringe Bedeutung zu; die Diskussion an dieser Stelle kann entsprechend knapp ausfallen.

Da es bei UDP zum Verlust der Daten während der Übertragung kommen kann, ist die Messung des Durchsatzes sowohl für die gesendeten UDP-Datagramme als auch für die empfangenen UDP-Datagramme notwendig. Die Differenz zwischen den beiden Werten (s. Abb. 4.19) entspricht der "verloren gegangenen" Anzahl von UDP-Datagrammen. Bei einer Nutzdatenlänge von 1.024 Byte wird weniger als die Hälfte der übertragenen Datagramme empfangen. Sobald jedoch der ma-

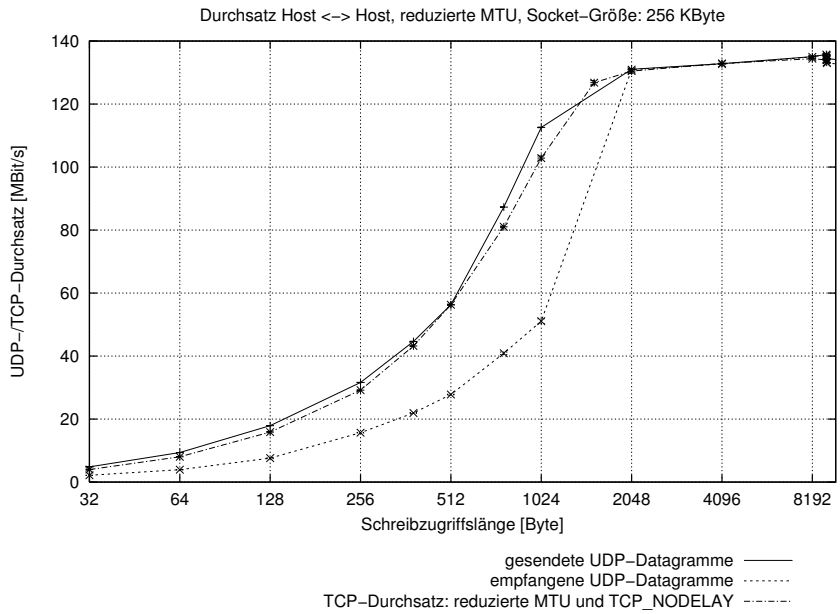


Abbildung 4.19: Maximaler Durchsatz über UDP

ximale Durchsatz des ATM-Netzes bei einer Schreibzugriffslänge von 2.048 Byte erreicht wird, überdecken sich die beiden Kurven wieder. Auch die zum Vergleich eingezeichnete Kurve des TCP-Datendurchsatzes überdeckt sich oberhalb der Schreibzugriffslänge von 2.048 Byte mit den Kurven des UDP-Datendurchsatzes.

Die in der Abbildung 4.19 zu erkennenden Differenzen zwischen gesendeten und empfangen UDP-Datagrammen sind auf einen geringeren Datagrammdurchsatz des Empfängers zurückzuführen. Das Empfangen der UDP-Datagramme ist deutlich aufwendiger als das Senden, so daß die UDP-Datagramme sogar in erster Linie beim Empfänger und nicht im Netz verlorengehen. Bei TCP wird in diesem Fall der Sender über die Flußkontrolle gebremst, damit nicht mehr Dateneinheiten übertragen werden, als empfangen werden können.

4.2.3 Nachrichtenlaufzeit

Neben dem Datendurchsatz ist die Nachrichtenlaufzeit ein weiterer wichtiger Eckwert für die Performanz von Firewalls. Die Nachrichtenlaufzeit sei hier definiert als die Zeit, die für die Übertragung einer Nachricht vom Sender zum Empfänger benötigt wird. Als Nachricht werden beliebige, von einer Anwendung generierte Daten bezeichnet. Daher wird auch die von den Endsystemen zum Generieren und Auswerten von Protokoll-Dateneinheiten benötigten Zeiten mit zur Nachrichtenlaufzeit gerechnet. Die Nachrichtenlaufzeit hängt somit ab von der Verar-

beitungszeit in den Endsystemen, der Verzögerung in den Netzkomponenten, der Nachrichtenlänge, dem Datendurchsatz des Netzes, der Ausbreitungsgeschwindigkeit des Signals im Medium und der Entfernung. Für die späteren Messungen der Auswirkungen von Firewall-Komponenten auf die Nachrichtenlaufzeit ist es wichtig, einen Referenzwert für die Nachrichtenlaufzeit auf einer direkten Verbindung zwischen zwei Systemen im Testnetz zu erhalten.

Da bei ATM-Netzen der Datendurchsatz sehr hoch ist und somit die Zeit zum Versenden einer Nachricht fester Länge selbst einen immer geringeren Einfluß auf den Durchsatz hat, wird die Nachrichtenlaufzeit ein immer wichtigerer Aspekt. Wenn eine Nachricht von 1.000 Byte Länge gesendet werden soll, so dauert die reine Sendezeit[45] bei 134 MBit/s nur etwa 60 Mikrosekunden (μs bzw. $10^{-6} s$). Verbindungsaufbau und Sicherheitsüberprüfungen im Firewall können dann leicht mehr Zeit benötigen, als die eigentliche Nachrichtenübermittlung.

Das direkte Messen der Nachrichtenlaufzeit würde synchronisierte Uhren beim Sender und Empfänger erforderlich machen. Zudem liegt die Nachrichtenlaufzeit in einem ATM-basierten LAN üblicherweise deutlich unter einer Millisekunde, was zusätzlich eine sehr hohe Genauigkeit der Uhren erforderlich macht.

Über eine indirekte Messung können die Nachrichtenlaufzeiten selbst mit einfacheren Meßwerkzeugen ermittelt werden. Dabei wird gemessen, wie oft innerhalb der Meßdauer (z. B. 30 Sekunden) jeweils ein Byte zwischen zwei Systemen ausgetauscht werden kann. Das Meßwerkzeug sendet dazu ein Byte und wartet danach darauf, wiederum ein Byte zu empfangen. Dieser "Umlauf" ("Round-Trip") wiederholt sich bis zum Ende der Meßdauer. Verbindungsauf- und -abbauzeiten zählen nicht zur Meßdauer (s. Abb. 4.20). Die Anzahl der Wiederholungen gibt Auskunft über die Nachrichtenlaufzeit.

Auf einer TCP-Verbindung zwischen zwei Endsystemen, die über den ATM-Switch miteinander verbunden sind, werden im Testnetz 2.593 $\pm$ 2 Umläufe pro Sekunde erreicht. Mit den leistungsfähigeren "Ultra-2/300"-Systemen kann über eine 622 MBit/s ATM-Schnittstelle mit 4.163 $\pm$ 28 Umläufe pro Sekunde ein um 61% besseres Ergebnis erzielt werden. Wird der Durchsatz auf der Schnittstelle administrationsseitig auf 155 MBit/s reduziert, werden nur 3.980 $\pm$23 Umläufe pro Sekunde erreicht.[46] Da jeder Umlauf aus zwei Nachrichten besteht, deren

[45] Die Sendezeit bezeichnet die Zeit, die benötigt wird, um eine Nachricht von x [Byte] Länge über ein Netz mit dem Durchsatz y [MBit/s] zu übertragen. Nicht eingerechnet sind die Zeiten für Verbindungsaufbau, Wartezeiten bis zum Empfang von Bestätigungen (ACKs) und Verzögerungen in den Endsystemen.

[46] Da der von der Messung verbrauchte Datendurchsatz deutlich unterhalb von 10 MBit/s liegt, kann der unterschiedliche Netzdurchsatz nicht die Ursache für das um 4% niedrigere Ergebnis sein. Es ist vielmehr zu vermuten, daß der Verarbeitungsaufwand im ATM-Treiber der Workstation durch das administrative Begrenzen des Schnittstellendurchsatzes steigt.

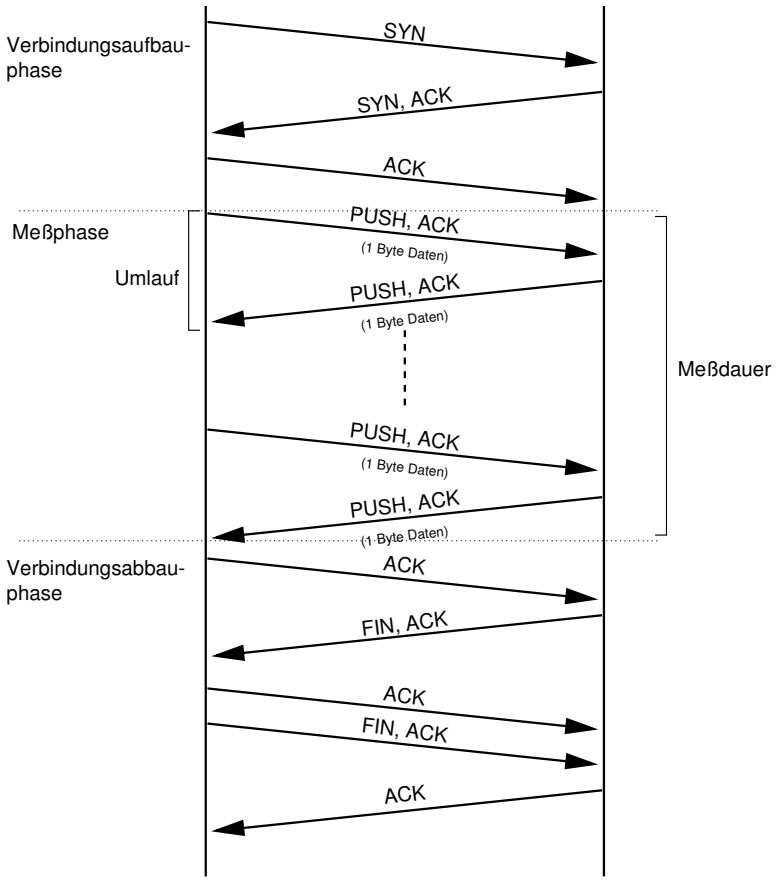


Abbildung 4.20: Umlaufzeitmessung

Nachrichtenlaufzeiten sich nicht überschneiden, kann die Nachrichtenlaufzeit über Gleichung 4.17 berechnet werden.

$$Nachrichtenlaufzeit\,[s] = \frac{1}{2 * Umläufe\ pro\ Sekunde} \tag{4.17}$$

Nach dieser Gleichung läßt sich die Nachrichtenlaufzeit für die Übertragung einer Nachricht vom Sender zum Empfänger (bei "Ultra-1"-Systemen) mit etwa 0.2 Millisekunden ($1,93 * 10^{-4}$ Sekunden) angeben.

Wie bereits die großen Unterschiede zwischen "Ultra-1"- und "Ultra-2"-Systemen bei den Nachrichtenlaufzeiten vermuten lassen, zeigt auch der Vergleich mit der berechneten Ausbreitungsgeschwindigkeit des Signals im Medium, daß die gemessene Nachrichtenlaufzeit allein durch die Verarbeitungszeiten in den Endsystemen und im Switch bedingt sind. Im Testnetz werden ausschließlich UTP-Kabel verwendet, deren Länge 5 Meter nicht überschreitet. Bei Verbindungen über den

Switch ergibt sich somit eine maximale Kabellänge von 10 Metern. Die Ausbreitungsgeschwindigkeit eines Signals in Kupferkabeln sei hier mit etwa 2/3 der Lichtgeschwindigkeit ($c = 300.000\,km/s$) angenommen. Daraus ergibt sich eine Verzögerung von $5*10^{-8}$ Sekunden.[47] Der Anteil der Nachrichtenlaufzeit, der sich aus der Übertragung im Kabel ergibt, ist im Testnetz zu gering, als daß ein Einfluß auf die Nachrichtenlaufzeit meßbar wäre. Die gemessene Nachrichtenlaufzeit ist somit nur durch Systemeffekte bedingt.

Im Gegensatz zu den Durchsatzmessungen zeigen die Parameter Socket-Größe und die `TCP_NODELAY`-Option keine Auswirkungen auf die Ergebnisse. Da jeweils nur ein Byte zwischen den Endsystemen ausgetauscht wird, bleiben die Socket-Größen ohne Bedeutung. Die `TCP_NODELAY`-Option bleibt auch ohne Wirkung, da immer erst Daten gesendet werden, nachdem die Bestätigung für die vorherige Dateneinheit eingetroffen ist. Nagles-Algorithmus, auf den sich die `TCP_NODELAY`-Option auswirkt (s. S. 220), verzögert nur solange das Generieren von TCP-Segmenten, bis alle vorher gesendeten TCP-Segmente bestätigt sind. Da bei der Messung immer die Benutzerdaten des Kommunikationspartners abgewartet werden, bevor eine neue Dateneinheit von dem Meßwerkzeug generiert wird, brauchen nie Daten der Anwendung gepuffert zu werden. Bei den folgenden Messungen der Umlaufzeiten ("Round-Trip Time", RTT) wurde auf die Variation dieser Parameter verzichtet und jeweils die Grundeinstellungen der Systeme verwendet (Socket-Größe von 8 KByte, `TCP_NODELAY`-Option nicht gesetzt).

Einzig die Meßdauer hat geringe Auswirkungen auf die Meßergebnisse, da bei kurzen Messungen (deutlich unter 10 Sekunden) die Abweichungen vom Mittelwert noch recht hoch sind. Für folgende Messungen wird daher eine Meßdauer von 30 Sekunden gewählt.

System	Medium	Umläufe [Umläufe/s]	Nachrichtenlaufzeit [s]
Ultra-1	ATM über Switch	2.593 ±2	$1,93 * 10^{-4}$
Ultra-1	ATM "Cross-Over"	2.807 ±5	$1,78 * 10^{-4}$
Ultra-1	"Localhost"	4.766 ±28	$1,05 * 10^{-4}$
Ultra-1	Ethernet	2.390 ±2	$2,09 * 10^{-4}$
Ultra-2	ATM 155 MBit/s über Switch	3.980 ±23	$1,26 * 10^{-4}$
Ultra-2	ATM 622 MBit/s über Switch	4.163 ±28	$1,20 * 10^{-4}$

Tabelle 4.3: Zusammenfassung der Umlaufmessungen

[47] Diese Verzögerung müßte um den Faktor 100 größer sein, um Auswirkungen auf die letzte angegebene Nachkommastelle der Nachrichtenlaufzeit zu haben. Bei einer angenommenen Kabellänge von einem Kilometer wäre dann eine Verlängerung der Nachrichtenlaufzeit um $6 * 10^{-6}$ Sekunden von $1,93 * 10^{-4}$ Sekunden auf $1,98 * 10^{-4}$ Sekunden zu erwarten.

Bei der Messung der Nachrichtenlaufzeit zeigt sich auch erstmals ein Unterschied zwischen einer Messung über ein "Cross-Over"-Kabel[48] und einer Messung über den ATM-Switch. Über das "Cross-Over"-Kabel kann mit 2.807 ± 5 Umläufe pro Sekunde ein um 8% besseres Ergebnis als bei der Messung über den ATM-Switch erreicht werden.[49] Die Übertragung einer Nachricht vom Sender zum Empfänger dauert über das "Cross-Over"-Kabel nach Gleichung 4.17 nur noch $1,78 * 10^{-4}$ Sekunden. Aus der Differenz der beiden Laufzeiten über den Switch und über "Cross-Over"-Kabel erhält man die durch den Switch bedingte Verzögerung einer Nachricht: $1,48 * 10^{-5}$ Sekunden. Dieses Ergebnis zeigt, daß die Verzögerung durch den Switch selbst mit relativ einfachen Meßwerkzeugen sichtbar gemacht werden kann. Im Gegensatz zu den Verzögerungen, die bei späteren Messungen der Nachrichtenlaufzeiten durch verschiedene Firewall-Komponenten bedingt werden, sind die Verzögerungen durch den ATM-Switch jedoch vernachlässigbar gering.

4.2.4 Verbindungsaufbauzeit

Die Messung der Nachrichtenlaufzeit (s. Abschnitt 4.2.3) beinhaltet nicht die zum Auf- und Abbau einer TCP-Verbindung benötigten Zeiten. Besonders bei kurzen Verbindungen (z. B. HTTP-Verkehr) dominieren die Verbindungsaufbauzeiten die Gesamtdauer einer Verbindung. Der gesamte Vorgang bestehend aus Verbindungsaufbau, Austausch von Daten und Verbindungsabbau wird als Transaktion bezeichnet. Für Proxy-Server, die auf Anwendungsebene den Verkehr durch einen Firewall kontrollieren, stellt das Management von Verbindungen einen großen Anteil der Gesamtlast dar. Ein wichtiger Eckwert bei Proxy-Servern ist daher der Verbindungsmanagementzeit, die über den Durchsatz der sequentiellen Transaktionen ermittelt werden kann. Zunächst wird die Transaktionsmessung vorgestellt; anschließend wird diskutiert, wie aus dem Transaktionsdurchsatz die Verbindungsaufbauzeit ermittelt werden kann.

Abbildung 4.21 zeigt den prinzipiellen Ablauf einer Transaktionsmessung anhand eines "Time-Sequence" Diagramms. Nach dem TCP-Verbindungsaufbau wird vom Client ein Byte geschickt, das vom Server ebenfalls mit einem Byte beantwortet wird. Dieser Datenaustausch entspricht einem Umlauf der Umlaufzeitmessung (s. Abb. 4.20). Anschließend wird die Verbindung abgebaut.[50] Über die

[48] Mit einem "gekreuzten" Kabel ("Cross-Over"-Kabel) können zwei Systeme direkt, d. h. ohne einen Switch, miteinander verknüpft werden.

[49] Eine Zusammenfassung der Ergebnisse der Umlaufmessungen ist in Tabelle 4.3 wiedergegeben.

[50] Der in Abbildung 4.21 wiedergegebene Nachrichtenfluß ist vereinfacht gegenüber dem tatsächlich während einer Messung zu beobachtenden Fluß, der zusätzliche Empfangsbestätigungen (ACKs) enthält.

Meßdauer wird die Anzahl der abgeschlossenen Transaktionen gemessen. Wichtig ist dabei, daß die Transaktionen sequentiell hintereinander ausgeführt werden. Da jede Transaktion einen großen Teil ihrer Laufzeit mit Warten auf Reaktionen verbringt, könnte durch parallele Bearbeitung von Transaktionen ein deutlich höherer Durchsatz erzielt werden.[51] Die hier vorgestellte sequentielle Transaktionsmessung dient daher primär der Ermittlung von Verbindungsmanagementzeiten.

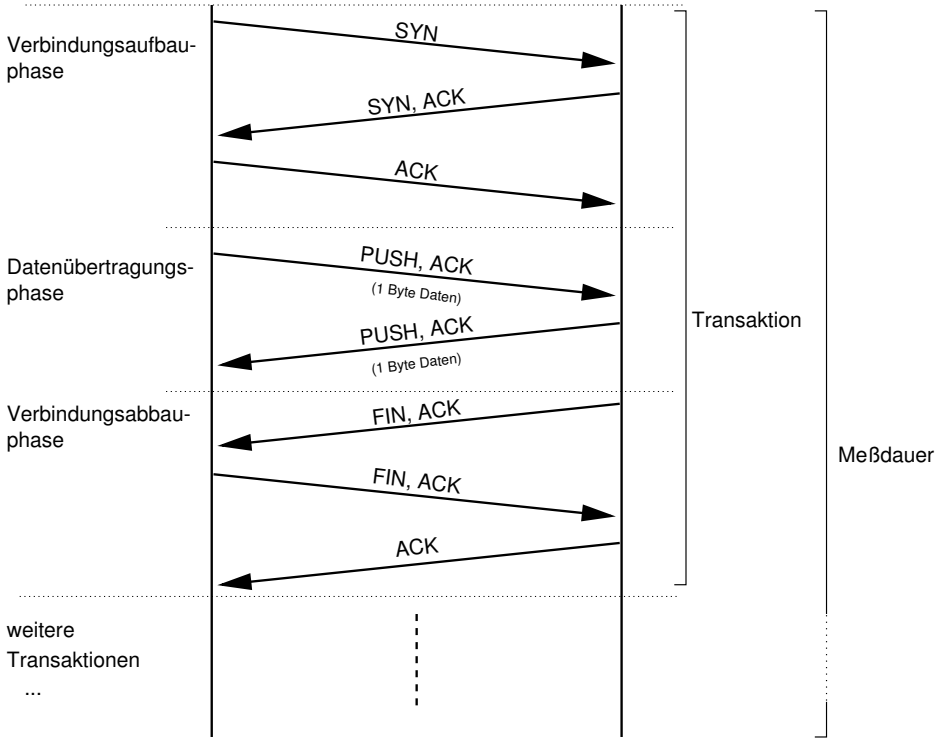


Abbildung 4.21: Transaktionsmessung

Eine Transaktionsmessung belastet die beteiligten Systeme stark. Die ersten Messungen zwischen zwei Hosts in Grundeinstellung brachen bei einzelnen Meßpunkten mit Fehlermeldungen ab und lieferten stark streuende Ergebnisse. Die Ursache konnte in dem Aufbrauchen aller Portnummern für Verbindungen zu dem für die Messungen auf dem Zielsystem verwendeten Port identifiziert werden. Nach dem Beenden einer TCP-Verbindung darf dasselbe Vierer-Tupel aus IP-Adresse und Portnummer des Clients sowie IP-Adresse und Portnummer des Servers (<Sender-IP-Adresse, Sender-Port, Empfänger-IP-Adresse, Empfänger-Port>) nicht sofort wiederverwendet werden, um auszuschließen, daß noch im

[51] Bei Leistungsdaten für Datenbank-Server wird üblicherweise die Performanz bei parallelen Transaktionen angegeben. Um den parallelen Transaktionsdurchsatz bei Proxies messen zu können, wurde ein eigenes Meßwerkzeug entwickelt (s. Abschnitt 4.2.5).

Netz befindliche Dateneinheiten der alten Verbindung die neue Verbindung beeinflussen.[52] Da die IP-Adressen und die serverseitige Portnummer feststehen, verwenden übliche TCP-Implementationen jeweils die nächste freie Portnummer oberhalb der bei der vorigen Verbindung verwendeten Portnummer. Der Client verbraucht dadurch langsam die zur Verfügung stehenden Portnummern von 1.024 bis 65.535. Bei Systemen, die einer normalen Netzlast ausgesetzt sind, muß dieselbe Portnummer erst nach Stunden oder gar Tagen wiederverwendet werden. Um die Zeit zu bestimmen, die abgewartet werden muß, bis ein Port erneut vergeben werden darf, muß abgewogen werden zwischen den beiden Aspekten: wie lange treffen noch Dateneinheiten der alten Verbindung ein und wie schnell werden 60.000 Portnummern verbraucht. Zeiten im Minutenbereich reichen in den meisten Fällen. Bei den im Test verwendeten Solaris 2.6 Systemen ist eine Wartezeit (`tcp_close_wait_interval`) von vier Minuten voreingestellt. Bei einer Transaktionsmessung werden 60.000 Ports jedoch innerhalb von etwas über zwei Minuten verbraucht. Durch das Heruntersetzen des Parameters für die Wartezeit (`tcp_close_wait_interval`) auf 30 Sekunden konnte erreicht werden, daß die Portnummern rechtzeitig wieder freigegeben werden. In einem reinen LAN, wie dem Testnetz, sind Wartezeiten im Sekundenbereich völlig ausreichend, um sicherzustellen, daß keine Dateneinheiten einer abgebauten Verbindung mehr im Netz sind.

Die Meßdauer muß bei Transaktionsmessungen mit zwei Minuten vergleichsweise lang gewählt werden. Bei kürzeren Messungen, wie der 60 Sekunden Messung (s. Abb. 4.22),[53] ergeben sich höhere Meßergebnisse, die bei einer längeren Belastung des Systems nicht dauerhaft erzielt werden können.[54] Bei den Messungen mit zwei, drei und vier Minuten Meßdauer bleiben die Meßergebnisse innerhalb der Vertrauensintervalle, so daß für weitere Messungen eine Meßdauer von 120 Sekunden ausreichend ist.

In Abbildung 4.22 ist weiterhin erkennbar, daß mit einer "Default" Socket-Größe von 8 KByte ein höherer Transaktionsdurchsatz erzielt werden kann als bei größeren Sockets. Die Differenzen zwischen den Socket-Größen von 32, 64 und 256 KByte bleiben innerhalb der Konfidenzintervalle.[55] Die Abbildung 4.23 zeigt,

[52] Die TCP-Verbindung wird für diese Zeit im Endsystem mit `TIME_WAIT` markiert.

[53] Die Abbildung 4.22 zeigt das Verhalten bei einer Messung über ein "Cross-Over"-Kabel. Vergleichsmessungen über den Switch oder nach "Localhost" zeigen qualitativ dasselbe Verhalten.

[54] Die Ursache für die höheren Meßergebnisse können auf asynchrone Operationen zurückzuführen sein. Der `close`-Aufruf zum Abbauen der TCP-Verbindungen kehrt zum Beispiel bereits zurück, bevor die Verbindung abgebaut ist. Das Meßwerkzeug betrachtet eine Transaktion jedoch bereits mit dem `close` als abgeschlossen. Durch die Wahl einer längeren Meßdauer werden die Auswirkungen durch asynchrones Schließen der Verbindung auf das Meßergebnis verringert.

[55] Weitere Messungen über den Switch oder nach "Localhost" zeigen auch hier qualitativ dasselbe Verhalten.

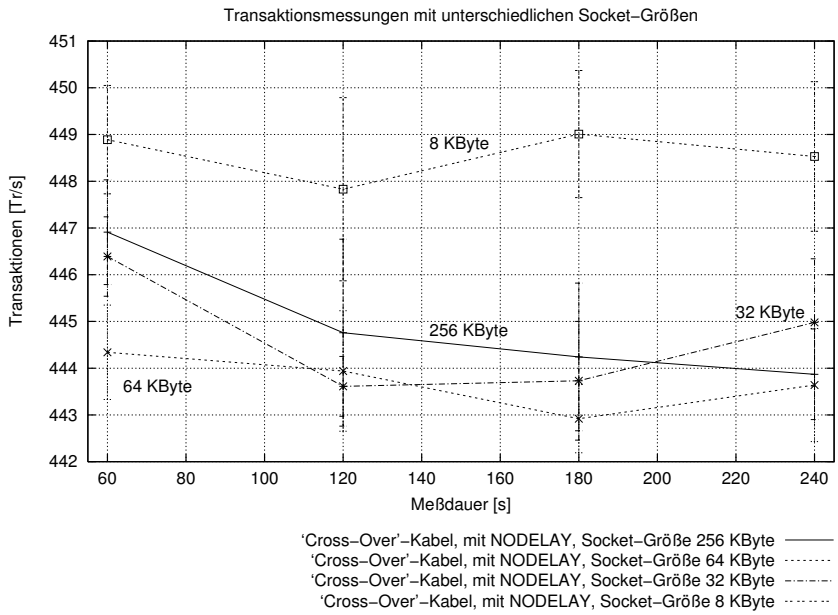


Abbildung 4.22: Transaktionsmessung über "Cross-Over"-Kabel

daß auch das Setzen der `TCP_NODELAY`-Option den Transaktionsdurchsatz verringert. Die Meßergebnisse mit `TCP_NODELAY` liegen mit 80 Transaktionen pro Sekunde (Tr/s) bei Messungen über "Cross-Over" und Switch sowie 100 Tr/s bei "Localhost"-Messungen um etwa 20% unter den Ergebnissen von Messungen in Grundeinstellung. Die Ursache für die geringeren Transaktionsdurchsätze liegt in beiden Fällen – den größeren Socket-Puffern und der `TCP_NODELAY`-Option – bei den zusätzlichen Systemaufrufen, die beim Aufbau jeder Verbindung benötigt werden. Sowohl das Einstellen der Socket-Größe als auch das Auswählen der `TCP_NODELAY`-Option ist über Aufrufe von `setsockopt()` implementiert. Die Applikation[56] wird unterbrochen bis der Systemaufruf zurückkehrt.

Die maximale Anzahl der Transaktionen für unterschiedliche Netze ist in Abbildung 4.23 zusammengefaßt. Alle Messungen wurden mit einer Meßdauer von 120 Sekunden und einer Socket-Größe von 8 KByte durchgeführt. Zusammengefaßt können über den Switch maximal 517 $\pm 1,8$ Transaktionen pro Sekunde erreicht werden. Bei der direkten Koppelung zweier ATM-Schnittstellen über ein "Cross-Over"-Kabel werden maximal 532 $\pm 1,5$ Tr/s gemessen. Bei den Messungen in Abschnitt 4.2.3 hatte sich bereits gezeigt, daß es meßbare Verzögerungen

[56] Die Applikation ist in diesem Fall das Meßwerkzeug "Netperf". Eine ähnliche Verringerung des Transaktionsdurchsatzes zeigten spätere Messungen auch beim HTTP-Meßwerkzeug (s. Abschnitt 4.2.7) und bei "MGproxy". Dort werden Aufrufe von `setsockopt()` für Einstellungen der Sende- und Empfangspuffergrößen (Socket-Größen) verwendet.

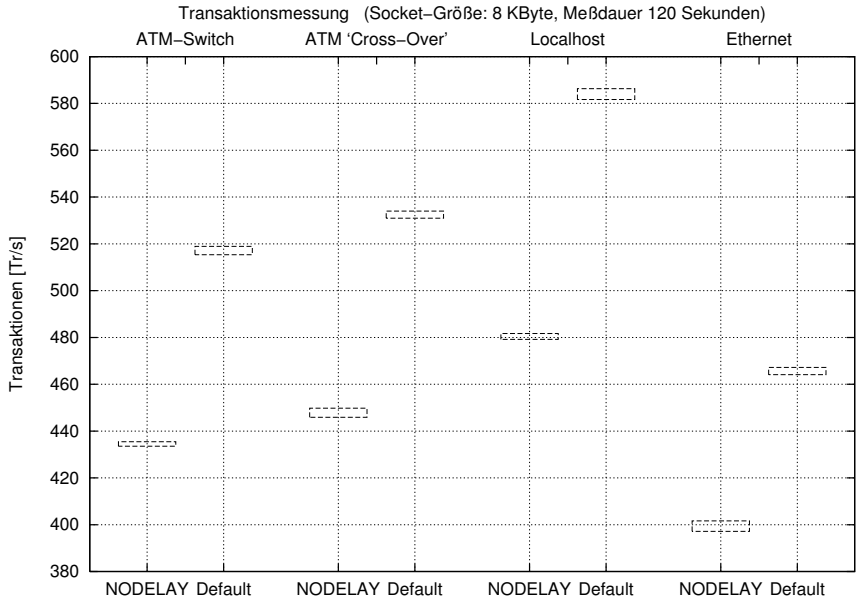


Abbildung 4.23: Transaktionsmessung über unterschiedliche Netze

durch den Switch gibt, die die Nachrichtenlaufzeiten im Vergleich zu einer Messung über "Cross-Over"-Kabel verlängern. Da die Transaktionsmessung durch die Nachrichtenlaufzeiten beeinflußt wird, muß über das "Cross-Over"-Kabel ein besseres Ergebnis erwartet werden als über den Switch. Noch geringere Nachrichtenlaufzeiten sind nur bei Verbindungen an "Localhost" möglich. Hier wird daher auch mit maximal $584 \pm 2,3$ Transaktionen der höchste Meßwert erreicht. Obwohl das klassische Ethernet nur einen sehr viel geringeren Datendurchsatz erreicht, unterscheiden sich die Meßwerte für die Transaktionsmessung mit $466 \pm 2,7$ Tr/s wenig von denen der anderen gemessenen Netze.[57] Messungen zwischen den beiden leistungsfähigeren "Ultra-2"-Systemen zeigen, daß die Leistungsfähigkeit der an der Messung beteiligten Systeme wichtiger ist als der Datendurchsatz oder die Nachrichtenlaufzeit auf dem Netz. Die leistungsfähigeren Systeme ermöglichen bei der Transaktionsmessung ein um 74% höheres Ergebnis. Die Erhöhung des Netzdurchsatzes von 155 auf 622 MBit/s bringt hingegen nur einen vergleichsweise geringen Zugewinn um weitere 5%. Hierdurch zeigt sich, daß der Hauptanteil

[57] Daß der Verbindungsaufbau über ein ATM-Netz kaum schneller ist als über ein klassisches Ethernet, zeigt sich in der Praxis zum Beispiel daran, daß ein "TCP-Portscan" wie er von dem Tool "Satan" (`ftp://ftp.cert.dfn.de/pub/tools/net/satan/`) benutzt wird, über ATM 1 Minute 25 Sekunden benötigte und über Ethernet nur 7 Sekunden länger dauert. Bei einem "TCP-Portscan" werden nacheinander alle Portnummer des Zielsystems angesprochen, um zu überprüfen, welche TCP-basierten Dienste auf dem Zielsystem aktiviert sind.

System	Medium	Transaktions-durchsatz [Tr/s]	Transaktions-länge [10^{-3} s]	Verbindungs-aufbauzeit [10^{-3} s]
Ultra-1	ATM über Switch	517 $\pm 1,8$	1,93	1,55
Ultra-1	ATM "Cross-Over"	532 $\pm 1,5$	1,88	1,52
Ultra-1	"Localhost"	584 $\pm 2,3$	1,71	1,50
Ultra-1	Ethernet	466 $\pm 2,7$	2,15	1,73
Ultra-2	ATM 155 MBit/s über Switch	899 $\pm 2,8$	1,11	0,86
Ultra-2	ATM 622 MBit/s über Switch	944 $\pm 2,8$	1,06	0,82

Tabelle 4.4: Zusammenfassung der Transaktionsmessungen

der Verbindungsaufbauzeit und auch der Nachrichtenlaufzeit während der Verarbeitung in den Endsystemen anfällt und nicht bei der Übertragung im Netz.

4.2.4.1 Verhältnis von Verbindungsauf- und -abbauzeit zur Transaktionsdauer

Die Gesamttransaktionslänge läßt sich einfach aus dem Transaktionsdurchsatz berechnen. Bei den Meßergebnissen im Bereich von 520 Transaktionen pro Sekunde für Verbindungen über den ATM-Switch und über Verbindungen über ein "Cross-Over"-Kabel beträgt die Gesamttransaktionslänge $1,92\ ms$ ($1,92 * 10^{-3}\ Sekunden$). Dieser Wert beinhaltet noch die für den Datenaustausch von einem Byte benötigte Umlaufzeit. Um die reinen Verbindungsaufbauzeit zu erhalten, muß man folglich die im vorigen Abschnitt gemessene Umlaufzeit ($3,86 * 10^{-4}$ Sekunden) von der Gesamttransaktionslänge abziehen. Für den Auf- und Abbau einer TCP-Verbindung wird im Testnetz somit eine konstante Zeit von $1,54 * 10^{-3}$ Sekunden ($1,92 * 10^{-3}\ Sekunden\ -\ 3,86 * 10^{-4}\ Sekunden$) benötigt. Da der Verbindungsabbau bei der Transaktionsmessung asynchron erfolgt, kann bereits die nächste Transaktion begonnen werden, bevor die Verbindungsabbauphase (s. Abb. 4.21) abgeschlossen ist. Daher wird durch diese Berechnung praktisch nur die Verbindungsaufbauzeit ermittelt.[58] Die Verbindungsaufbauzeit ergibt sich somit aus Gleichung 4.18.

[58] Auch bei der Bewertung der Performanz von Firewalls sind die Verbindungsaufbauzeiten entscheidend, da auch bei praktischen Anwendungen der Verbindungsabbau asynchron erfolgen kann und somit für den Benutzer keine Auswirkungen auf die (empfundene) Performanz hat.

$$Verbindungsaufbauzeit\,[s] = \frac{1}{Transaktionsdurchsatz} - \frac{1}{Umläufe\ pro\ Sekunde} \tag{4.18}$$

Bei kurzen Nachrichten dominiert die konstante Verbindungsaufbauzeit die Gesamttransaktionslänge. Erst bei längeren Nachrichten hat die Nachrichtenübertragungszeit einen nennenswerten Anteil an der Gesamttransaktionslänge. Bei welcher Nachrichtenlänge der Anteil der Datenübertragung an der Gesamttransaktionslänge größer als der Verbindungsoverhead wird, hängt vom maximalen Datendurchsatz für die jeweilige Nachrichtenlänge ab. In Abbildung 4.24 wird der konstanten Verbindungsauf- und -abbauzeit von $1,54 * 10^{-3}$ Sekunden die Nachrichtenübertragungszeit in Abhängigkeit von der Nachrichtenlänge entgegengestellt. Die Nachrichtenübertragungszeit wurde dabei anhand der Datendurchsatzmessungen in Abschnitt 4.2.2 ermittelt.[59] Den Kurven in Abbildung 4.24 liegen die Meßergebnisse jeweils einer Host-zu-Host-TCP-Datendurchsatzmessung mit 256 KByte Socket-Größe über ATM (s. Abb. 4.5.1) und über Ethernet zugrunde. Erkennbar ist, daß bei der Nachrichtenübertragung über ATM bis 4 KByte Nachrichtenlänge praktisch die gesamte Transaktionslänge für Verbindungsauf- und -abbau benötigt wird. Erst bei ungefähr 25 KByte langen Nachrichten reicht die Nachrichtenübertragungszeit an Verbindungsauf- und -abbauzeit heran. Bei Verbindungen über Ethernet wird für die Nachrichtenübertragung zum Vergleich sehr viel mehr Zeit benötigt, so daß die Kurve hier sehr viel steiler ansteigt.

Diese Messungen zeigen, wie wichtig die Verbindungsaufbauzeit in Hochgeschwindigkeitsnetzen ist. Wie bereits im Abschnitt 3.1 diskutiert, verringern sich die Verbindungsaufbauzeiten in Hochgeschwindigkeitsnetzen kaum, da in lokalen Netzen fast ausschließlich die Leistungsfähigkeit der Endsysteme die Verbindungsaufbauzeit bestimmt (s. o.) und zudem auch in WANs die Datagrammlaufzeiten praktisch nicht verringert werden können, weil diese durch die Ausbreitungsgeschwindigkeit des Signals im Übertragungsmedium begrenzt sind. Bei den hohen Datendurchsätzen in einem ATM-Netz ist die Datenübertragung "billig" im Verhältnis zu den mehrfachen Nachrichtenumläufen, wie sie beim TCP-Verbindungsaufbau und bei vielen Anwendungsprotokollen (SMTP, FTP, ...) notwendig sind. Diese Erkenntnis, daß häufiger Verbindungsauf- und -abbau (z. B. HTTP) und unnötige Nachrichtenumläufe in Hochgeschwindigkeitsnetzen zu starken Performanzeinbußen führen, muß sich auf die Entwicklung zukünftiger Protokolle auswirken. In Hochgeschwindigkeitsnetzen wie ATM ist ein Vorgehen erfolgversprechend, bei dem alle im Normalfall benötigten Daten *immer* übertragen werden, auch wenn dadurch in *einigen* Fällen Daten gesendet werden, die vom Empfänger

[59] Dieses Vorgehen läßt Anlauf-Aspekte (wie "Slow-Start") bei der Datenübertragung über TCP außer acht.

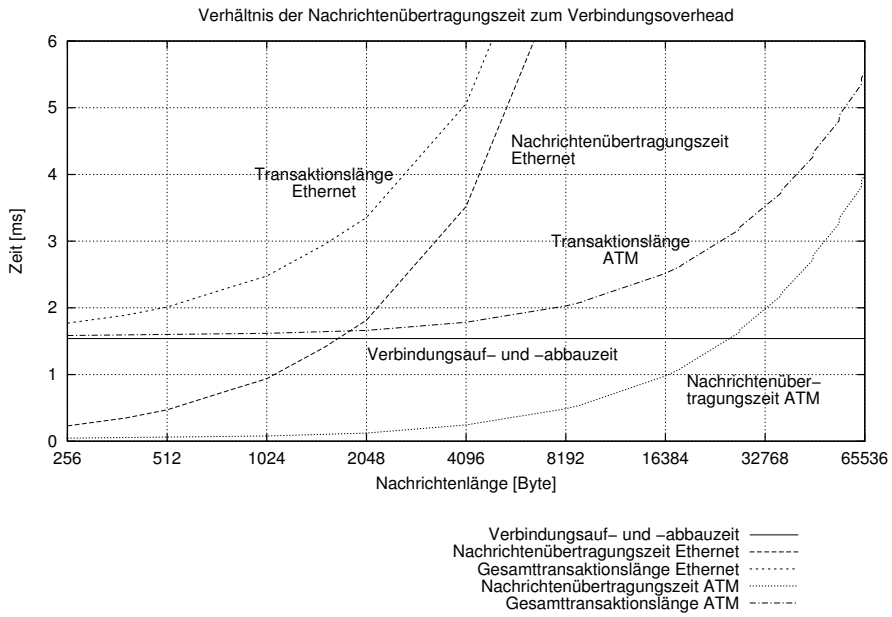


Abbildung 4.24: Anteil des Verbindungsauf- und -abbaus an der Übertragung

nicht benötigt werden oder nicht ausgewertet werden können. Bei den heutigen Protokollen werden wichtige Daten meist nur auf Anforderung, also nach einem zusätzlichen Nachrichtenumlauf, versendet, um so die Menge der übertragenen Daten zu minimieren.

Wie sich Firewalls in bezug auf Verbindungsauf- und -abbauzeiten verhalten, zeigen die Messungen in den Abschnitten 4.3.5, 4.4.1.3 und 4.4.2.3. Da bei Firewalls jedoch noch die Überprüfungsfunktion hinzukommt, ist bereits zu erwarten, daß die Verbindungsaufbauzeiten durch den Einsatz von Firewalls stark zunehmen werden.

4.2.5 HTTP-Meßwerkzeug

Mit den bisherigen Messungen können über die Performanz von Firewalls auf Anwendungsebene noch keine ausreichenden Aussagen getroffen werden. Zwar sind auch hier Datendurchsatz, Nachrichtenlaufzeiten und Verbindungsaufbauzeiten wichtige Performanzdaten; da ein Proxy jedoch für jede Verbindung einen eigenen Verbindungskontext führen muß, reicht es nicht aus, Messungen nur über *eine* Verbindung durchzuführen. Zusätzlich werden auch Aussagen über die maximale Anzahl paralleler Verbindungen, die maximale Anzahl paralleler Transaktionen sowie die Gesamt- und Einzeldurchsätze bei parallelen Verbindungen benötigt

(s. Abschnitt 4.1). Zudem müssen die verwendeten Meßwerkzeuge für Messungen der Performanz von Proxy-Servern das jeweilige Protokoll unterstützen.

Ein bei Performanztests von Firewalls in Zeitschriften häufig verwendeter Ansatz ist es, auf jeweils beiden Seiten des Firewalls eine große Anzahl kostengünstiger PCs über Fast-Ethernet anzuschließen und von den PCs HTTP- und FTP-Aufträge generieren zu lassen. Durch den Einsatz geeigneter "Edge-Devices"[60] am Übergang zwischen Ethernet und ATM-Netz kann dieser Ansatz auch auf Firewalls in ATM-Netzen übertragen werden. Der Vorteil dieses Ansatzes ist neben dem Einsatz größtenteils kostengünstiger Komponenten die Realitätsnähe, da auch in der Praxis Endsysteme weiterhin über Fast-Ethernet angeschlossen werden und der direkte Anschluß von Endsystemen an ATM-Netze die Ausnahme sein wird. Problematisch ist dieser Ansatz im Hinblick auf die Qualität der Ergebnisse. Bei Experimenten im DFN-FWL konnten mit PCs in einem Fast-Ethernet keine zu den im ATM-Netz durchgeführten Messungen vergleichbaren Vertrauensintervalle erzielt werden.[61] Ein Vergleich der Performanzeigenschaften unterschiedlicher Firewall-Setups ist daher mit über Fast-Ethernet angeschlossenen PCs nur bedingt möglich.

Ein noch schwerwiegenderes Problem stellt die fehlende Synchronisation zwischen den einzelnen Verbindungen über den Firewall dar. Wenn nur ein Teil der parallelen Verbindungen gleichzeitig geöffnet werden kann und die übrigen Verbindungen erst später geöffnet werden, sind die von den ersten Verbindungen erzielten Einzeldurchsätze entsprechend höher, da der insgesamt zur Verfügung stehende Gesamtdurchsatz nur auf eine geringere Anzahl von Verbindungen aufgeteilt werden muß. Im Extremfall macht nur ein sehr kleiner Teil der Verbindungen gleichzeitig Fortschritt. Nur über eine Synchronisation der einzelnen Verbindungen kann sichergestellt werden, daß auch die angeforderte Anzahl von Verbindungen gleichzeitig aktiv ist.

Folgende Anforderungen muß das Meßwerkzeug erfüllen:

Messungen mit parallelen Verbindungen: Das Meßwerkzeug soll zur Messung des **Gesamtdatendurchsatzes** bei mehreren parallelen Verbindungen eingesetzt werden. Zusätzlich sollen die **Einzeldurchsätze** gemessen werden, damit Aussagen über die Verteilung der Ressourcen (Fairneß) gemacht werden kann. Der erreichte **Transaktionsdurchsatz** muß ebenfalls gemessen werden.

[60] Als "Edge-Device" wird ein System bezeichnet, der einen Übergang zwischen Ethernet und ATM-Netz bildet. Das "Edge-Device" überträgt hierzu Ethernet-Pakete über eine virtuelle Verbindung an LANE- oder MPOA-Clients im ATM-Netz. [ATM Forum 97d]

[61] Bei der Wiederholung konnte ein Meßergebnis um 10 bis 20 MBit/s (bzw. um 50%) von der vorigen Messung abweichen.

Transaktionen festgelegter Länge oder festgelegter Dauer: Für die auf den Verbindungen ausgeführten Transaktionen muß es möglich sein, eine bestimmte Datenmenge vorzuschreiben, die transferiert werden muß. Zusätzlich muß es auch möglich sein, die Zeitdauer jeder Transaktion vorzuschreiben.

Synchronisation zwischen parallelen Verbindungen: Wie bereits dargestellt, ist die Synchronisation zwischen den parallelen Verbindungen notwendig, um zu garantieren, daß während der Messung die geforderte Anzahl von parallelen Verbindungen aktiv ist. Meßwerte dürfen nur dann aufgezeichnet werden, wenn die geforderte Anzahl von Verbindungen gleichzeitig aktiv ist.

Wiederholbarkeit der Messung: Wiederholungen von Messungen mit denselben Parametern müssen Ergebnisse innerhalb der Konfidenzintervalle liefern. Zur Berechnung von Mittelwerten, Standardabweichungen und Konfidenzintervallen ist die automatische Wiederholung von Messungen durch das Meßwerkzeug erforderlich.

Effiziente Implementation: Das Meßwerkzeug darf nicht selbst zum Engpaß werden. Es muß einen hohen Datendurchsatz, eine große Anzahl paralleler Verbindungen und einen hohen Transaktionsdurchsatz erreichen, damit eine ausreichende Anzahl von Clients simuliert werden kann.

Verwendung von HTTP: Zunächst soll die Performanz von Proxies mit einem möglichst einfachen und gut verstandenen Protokoll gemessen werden. Die Wahl fiel daher auf HTTP, für das auch mehrere Proxies verfügbar sind. Für spätere Messungen sollen aber auch Erweiterungen für Telnet und FTP möglich sein.

Erweiterungsmöglichkeiten: Das Meßwerkzeug muß leicht an neue Meßszenarien anpaßbar sein. Dies kann zum Beispiel die bereits angesprochene Unterstützung weiterer Protokolle wie Telnet und FTP sein. Implementiert wurde hier eine Integration von SSL zur Authentisierung und zur Verschlüsselung der übertragenen Daten (s. Abschnitt 4.2.7).

Da keines der verfügbaren Meßwerkzeuge die gestellten Anforderungen erfüllte, wurde im Rahmen des DFN-FWLs ein Meßwerkzeug für die Bewertung der Performanz von Proxy-Servern entwickelt.[62]

[62] Die beschriebene Analyse der Anforderungen führte zu meiner Idee für dieses Meßwerkzeug. Die Spezifikation wurde von meinem Kollegen Carsten Benecke umgesetzt. Die Entwurfskriterien und die Implementierung des Meßwerkzeuges sind in [Benecke et al. 98b] ausführlich dokumentiert. Die in nachfolgenden Abschnitten diskutierten Messungen mit diesem Werkzeug wurden wieder von mir durchgeführt.

Das Meßwerkzeug besteht aus einem Client und einem Server. Vom Client werden HTTP-Anfragen [Berners-Lee et al. 96] generiert, die vom Server mit legalen HTML-Dokumenten [Berners-Lee et al. 95] beantwortet werden. Im Gegensatz zu regulären HTTP-Clients und -Servern werden die übertragenen HTML-Dokumente aus Performanzgründen jedoch nicht vom HTTP-Meß-Server von einer Festplatte eingelesen und auch vom Meß-Client nicht angezeigt. Beide Operationen sind aufwendig und können daher einen Engpaß bei der Messung bilden. Stattdessen werden die angeforderten Dokumente vom Server anhand einfacher Regeln generiert. Der Client muß nur den Empfang der vom Server gesendeten Daten sicherstellen, kann die Daten danach aber sofort verwerfen. Dadurch wird gewährleistet, daß das Meßwerkzeug nicht selbst zum Engpaß bei der Messung der Performanz von Proxies wird.

Je nach Anforderungen der Messung kann als Transaktion eine Nachricht bestimmter Länge gesendet werden oder vom Server werden über eine festgelegte Zeit Daten gesendet. Die gewünschte Transaktion wird dem Server vom Client über eine URL [Berners-Lee et al. 94] mitgeteilt, in der die jeweiligen Parameter codiert sind.[63] Der Client protokolliert den bei der Übertragung erzielten Datendurchsatz (MBit/s) und den Transaktionsdurchsatz (Tr/s). Jedes Meßexperiment wird zur statistischen Absicherung mehrfach ("N") wiederholt. Über den gemessenen Werten werden Mittelwert, Standardabweichung und Konfidenzintervall berechnet.

Synchronisation: Wie bereits Eingangs dargestellt, ist eine Synchronisation der parallelen Verbindungen erforderlich. Die Synchronisation wird dadurch erleichtert, daß alle Verbindungen vom selben Prozeß verwaltet werden. Jeder Thread wiederholt fortlaufend die gewünschte Transaktion und meldet über eine Semaphore, sobald die erste Transaktion abgeschlossen wird. Nachdem alle Threads ihre erste Transaktion durchgeführt haben, ist die "Anlaufphase" (s. Abb. 4.25) abgeschlossen und es beginnt die "Meßphase". Jeder Thread beginnt bei seiner nächsten Transaktion mit der Aufzeichnung der Meßergebnisse.[64] Ist die geforderte Anzahl von Transaktionen ("n") vom Thread abgearbeitet worden, erfolgt wieder eine Meldung. In der nun folgenden "Auslaufphase" führt der Thread solange weitere Transaktionen durch, bis auch alle anderen Threads ihre Meßphase beendet haben. Durch die Anlaufphase wird sichergestellt, daß die Aufzeichnung von Meßdaten erst beginnt, sobald die geforderte Anzahl von Threads aktiv ist.

[63] Fordert der Client zum Beispiel die URL `http://test/k8.html` an, generiert der Server eine 8 KByte lange Nachricht und sendet diese an den Client zurück. Bei `http://test/s30.html` werden 30 Sekunden lang Daten vom Server an den Client gesendet. Eine ausführlichere Vorstellung der verwendeten Syntax findet sich in [Benecke et al. 98b].

[64] In Abbildung 4.25 sind die Transaktionen, deren Meßergebnisse aufgezeichnet werden, fett dargestellt.

Mit der Auslaufphase wird erreicht, daß während der Aufzeichnung von Meßdaten die geforderte Anzahl paralleler Verbindungen auch dann erreicht wird, wenn einzelne Threads ihre Meßphase bereits beendet haben.

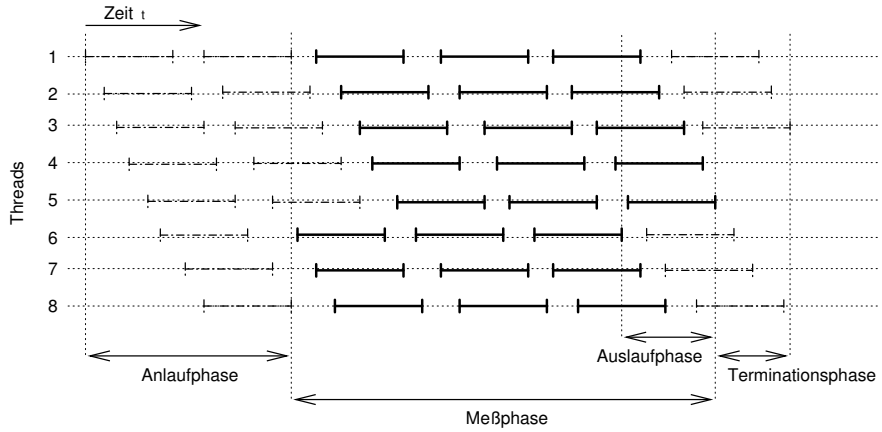


Abbildung 4.25: Synchronisation paralleler Anwendungsmessungen

Durch die Anlauf- und Auslaufphasen kann allerdings nicht sichergestellt werden, daß die vorhandenen Betriebsmittel gleichmäßig auf die einzelnen Threads verteilt werden. So kann es trotzdem vorkommen, daß ein Thread einen höheren Durchsatz erzielt und die geforderte Anzahl der Transaktionen schneller beendet als andere Threads. Diese ungleichmäßige Verteilung ist nicht unbedingt von Nachteil, denn sie entspricht dem Verhalten in realen Firewall-Umgebungen, wo unterschiedlich leistungsfähige Endsysteme über Verbindungen mit unterschiedlichen Datendurchsätzen über den Firewall kommunizieren. Anstelle einer Vermeidung ist es daher wichtiger, diese Effekte meßbar zu machen. Hierfür wird ausgegeben, wieviele Transaktionen jeder Thread bis zum Ende der Auslaufphase durchführen konnte und wie stark die Datendurchsätze der einzelnen Threads voneinander abweichen.

Überwachung der Messungen: Durch die Synchronisation der Messungen über die verschiedenen Meßphasen droht die Gefahr, daß einzelne Threads ihre Anlauf- oder Meßphase nicht abschließen können und dadurch einzelne Messungen endlos laufen würden. Solche Fälle treten insbesondere bei Verlust von Dateneinheiten und bei fehlerhaftem Verhalten des zu vermessenden Proxy-Servers auf. Da der Meß-Client nach dem Absetzen der "`GET`-Anfrage" dem HTTP-Protokoll [Berners-Lee et al. 96] entsprechend nur noch passiv auf Daten und den anschließenden Verbindungsabbau durch den Server wartet, kann der Verlust mehrerer Datagramme des Servers dazu führen, daß der wartende Client den Verbindungsabbau des Servers nicht bemerkt. Zwar erwartet TCP [Postel 81b] auch beim

Verbindungsabbau eine Bestätigung der Gegenseite, jedoch ist der Verbindungsabbau nach einem "Time-Out" auch ohne Bestätigung möglich, um nach einem Systemausfall des Kommunikationspartners eine Verbindung beenden zu können. Da für eine Messung oft mehrere Millionen Transaktionen durchgeführt werden müssen, tritt selbst ein unwahrscheinlicher Fall des Verlustes mehrerer Dateneinheiten leicht ein. Um solche Fälle zu erkennen, überwacht ein weiterer Thread des Meßwerkzeuges, der im folgenden als "Kontroll-Thread" bezeichnet wird, den Fortschritt der einzelnen Threads. Wird innerhalb einer festgelegten Zeitspanne[65] auf einer Verbindung kein Fortschritt erzielt, wird die Verbindung vom Kontroll-Thread abgebrochen. Der Durchsatz dieser fehlgeschlagenen Transaktion wird vom Meßwerkzeug mit 0 MBit/s gewertet. Dieser Abbruch von Verbindungen, die über längere Zeit keine Fortschritt erzielt haben, entspricht dem typischen Benutzerverhalten.

Effiziente Implementation: Damit vom Meßwerkzeug eine möglichst große Zahl paralleler Verbindungen bedient werden kann, können jeder einzelnen Verbindung nur geringe Betriebsmittel zur Verfügung gestellt werden. Eine effiziente Implementation von Client und Server ist über eine "multithreaded" Programmierung möglich. Dies bedeutet, daß für die Bearbeitung jeder Verbindung nicht, wie bei Servern allgemein üblich, ein eigener Prozeß generiert wird, sondern daß jeder Verbindung innerhalb desselben Prozesses lediglich ein eigener "Ablauffaden" (Thread) zugeordnet wird. Auf diese Weise benötigt jede Verbindung nur geringe Ressourcen, und entsprechend mehr parallele Verbindungen können vom Meßwerkzeug generiert werden.

Bei den in den folgenden Unterabschnitten vorgestellten Meßergebnissen wird das HTTP-Meßwerkzeug (Meß-Client und Meß-Server) auf den leistungsfähigeren "Ultra-2/300"-Systemen eingesetzt (s. Abschnitt 4.2.2.8).

4.2.6 Maximale Anzahl paralleler Verbindungen

Ein wichtiger Aspekt bei Firewalls auf Anwendungsebene ist die maximale Anzahl von Verbindungen, die zur selben Zeit (parallel) über den Firewall geführt werden können. Da für jede Verbindung auf dem Firewall ein eigener Kontext geführt werden muß, steigt mit jeder zusätzlichen parallelen Verbindung der Verwaltungsaufwand auf dem Firewall. In der Regel ist auf dem Firewall für jede Verbindung ein eigener Proxy-Server Prozeß oder Thread notwendig. Das Meßwerkzeug muß daher viele parallele Verbindungen über den Firewall öffnen und bedienen können. Für Messungen ergibt sich hierbei das Problem, daß die parallelen Verbindungen synchronisiert werden müssen, um sicherzustellen, daß während

[65] Der bei den folgenden Messungen verwendete voreingestellte Wert beträgt 60 Sekunden.

der Meßdauer die gewünschte Anzahl paralleler Verbindungen vom Meßwerkzeug bedient wird.

Die maximale Anzahl paralleler Verbindungen kann mit dem HTTP-Meßwerkzeug ermittelt werden. Hierzu werden vergleichsweise lange Transaktionen (30 Sekunden) gewählt, von denen jeder Thread nur eine geringe Anzahl von Wiederholungen (n=3) durchzuführen braucht. Nach jedem Meßpunkt wird die Anzahl der vom Meßwerkzeug verwendeten Threads erhöht. Mit dieser Messung kann die maximale Anzahl paralleler Verbindungen über einen zwischen Meß-Client und Meß-Server installierten Firewall bestimmt werden.

Die Messung der maximalen Anzahl paralleler Verbindungen verläuft zwischen den beiden "Ultra-2"-Systemen und auch zwischen den weniger leistungsfähigen "Ultra-1"-Systemen bis zu maximal 30 parallelen Threads problemlos. Bei allen Meßpunkten entspricht der Gesamtdurchsatz aller Threads dem theoretisch möglichen Durchsatz auf einem 155 MBit/s ATM-Netz. Zudem kann bei allen Meßpunkten ein sehr kleines Konfidenzintervall eingehalten werden.

Abbildung 4.26 zeigt den Transaktionsdurchsatz eines Threads (Einzeltransaktionsdurchsatz) in Abhängigkeit von der Anzahl der Threads. Bei der für diese Messung gewählten festen Transaktionslänge von 30 Sekunden kann jeder Thread zunächst noch maximal eine Transaktion in 30 Sekunden durchführen.[66] Erst oberhalb von 50 parallel laufenden Threads steigt die CPU-Last auf den Systemen so stark, daß das Meßwerkzeug auf der Client-Seite nach einer abgeschlossenen Transaktion nicht sofort die nächste Transaktion beginnen kann und auch auf der Server-Seite das Meßwerkzeug nicht exakt nach 30 Sekunden eine Transaktion beendet wird.[67] Als Folge dieser Verzögerungen auf Client- und auf Server-Seite fällt der Einzeltransaktionsdurchsatz leicht ab. Trotz des zurückgehenden Einzeltransaktionsdurchsatzes steigt der Gesamttransaktionsdurchsatz (Summe aller Einzeltransaktionsdurchsätze) innerhalb des gesamten Meßbereichs weiter an. Mit steigender Anzahl von Threads steigt auch die CPU-Last auf den Meßsystemen. Ab 60 Threads werden vermehrt Transaktionen wegen Zeitüberschreitungen ("Time-Out") vom Kontroll-Thread abgebrochen. Dies wirkt sich auf die aufsummierten Transaktions- und Datendurchsätze aus, da abgebrochene Transaktionen Betriebsmittel verbrauchen, ohne zu den beiden Durchsätzen beizutragen. Kontrollen[68] zeigen jedoch, daß die Anzahl paralleler Verbindungen weiterhin mit der Anzahl der Threads übereinstimmt. Oberhalb von 200 Threads

[66] Bei den Meßpunkten unterhalb von 10 Threads verläuft die Kurve nur deshalb nicht gleichmäßig, weil sich hier Rundungsfehler stark bemerkbar machen.

[67] Normalerweise schließt der Server eine Verbindung nach 30 Sekunden und beendet damit die Transaktion. Durch eine hohe CPU-Last auf dem Server kann es vorkommen, daß einzelne Transaktionen deutlich länger als 30 Sekunden dauern.

[68] Da ein Eingriff zu geringen Veränderungen der Meßergebnisse führen kann, werden diese Stichproben bei zusätzlichen Einzelmessungen aufgezeichnet.

können erstmals nicht mehr alle zur Berechnung von Mittelwert und Standardabweichung geforderten 15 Wiederholungen (N=15) pro Meßpunkt durchgeführt werden. Wegen zu häufiger Zeitüberschreitung ("Time-Out") bricht das Meßwerkzeug Messungen ab, andere Messungen laufen jedoch immer noch bis zum Ende durch. Trotz dieser vorzeitigen Abbrüche der Messungen mit 250 und 300 Threads ist es bei der folgenden Messung mit 400 Threads wieder möglich alle 15 Wiederholungen ohne Fehler abzuschließen. Bei den "Ultra-2"-Systemen können auf diese Weise Messungen mit bis zu 800 Threads durchgeführt werden.[69] Die weniger leistungsfähigen "Ultra-1"-Systeme reichen nur für Messungen bis maximal 400 Threads aus. Die Unterschiede in der Leistungsfähigkeit der "Ultra-1"- und "Ultra-2"-Systeme ist zu gering, um sie anhand eines unterschiedlichen Kurvenverlaufes identifizieren zu können.[70] Die unterschiedliche Leistungsfähigkeit zeigt sich nur darin, daß die Messungen unterschiedlich früh abgebrochen werden. Bei der Messung mit den "Ultra-1"-Systemen kann zuletzt mit 400 Threads die geforderte Anzahl von "N=15" Wiederholungen des Meßexperiments ohne Fehler durchgeführt werden; bei den "Ultra-2"-Systemen ist dies bis 800 Threads möglich. Messungen mit bis zu 1.000 Threads laufen zwar auf beiden Systemen an, können auch zunächst die geforderte Anzahl von Verbindungen aufbauen, aber einzelne Threads werden dann im Laufe der Messung vom Betriebssystem nicht ausreichend häufig berücksichtigt, so daß sehr viele "Time-Outs" auftreten und die Messung vor dem Meßende abbricht.

In der im vorigen Abschnitt beschriebenen Auslaufphase führt ein Thread, der bereits die geforderte Anzahl an Transaktionen bearbeitet hat, weitere Transaktionen durch, damit auch andere Threads bei gleichbleibend hoher Last ihre Meßphase beenden können. Werden einzelne Threads während der Messung vom Betriebssystem benachteiligt, so benötigen diese Threads deutlich länger, um die geforderte Anzahl an Transaktionen zu beenden. Daher kann auch die Länge der Auslaufphase bzw. die Gesamtanzahl von Transaktionen, die im Mittel von einem Thread durchführen werden müssen, Auskunft über die Belastung der Meßsysteme während einer Messung geben. In Abbildung 4.27 steigt die durchschnittliche Anzahl der Transaktionen zusammen mit der Server CPU-Last[71] oberhalb von 50 Threads stark an. Bei 800 Threads führt jeder Thread im Mittel etwa 23 Transaktionen durch, bevor die Meßphase beendet ist. Da die Meßphase beendet wird, wenn der letzte Thread die geforderte Anzahl von "n=3" (!) Transaktionen

[69] Die Ursache dafür, daß eine Messung mit 300 Threads vorzeitig abgebrochen wird und eine Messung mit 400 Threads mit der geforderten Anzahl von Wiederholungen abgeschlossen werden kann, ist auf Wechselwirkungen zwischen der Aktivierung der Threads durch das Betriebssystem und dem Eintreffen von Dateneinheiten für den jeweiligen Thread zurückzuführen.

[70] Aus diesem Grund sind in Abbildung 4.26 nur die Meßergebnisse mit "Ultra-2"-Systemen aufgezeichnet. Die Kurven für "Ultra-1"-Systeme überdecken sich mit den dargestellten Kurven.

[71] Der Meß-Server wird durch die Messung stärker belastet als der Client.

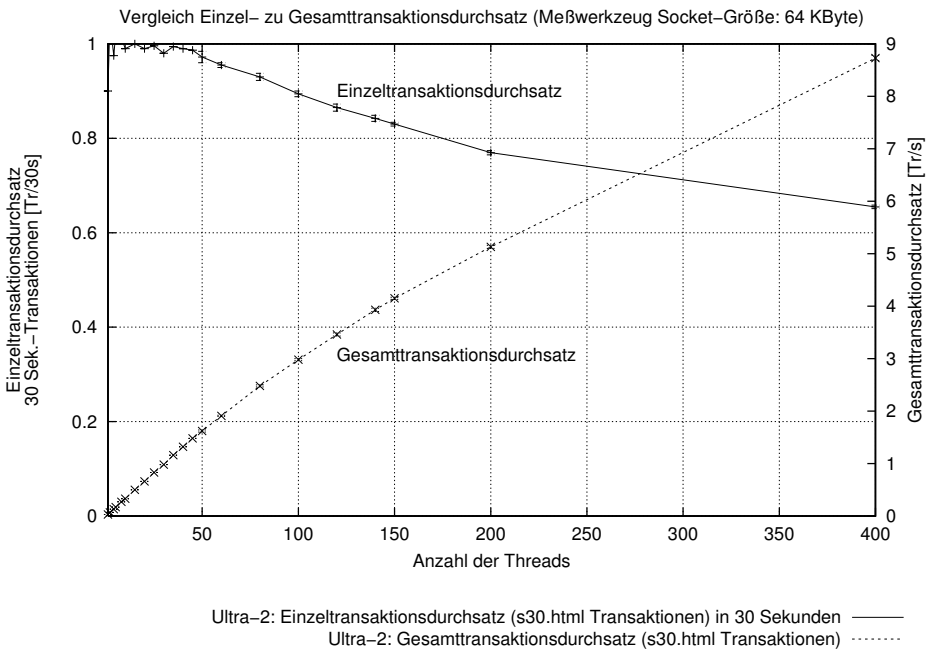


Abbildung 4.26: Vergleich von Einzel- zu Gesamttransaktionsdurchsatz

durchgeführt hat, zeigt dies die ungleichmäßige Verteilung der Ressourcen durch das Betriebssystem an die einzelnen Threads.[72]

Die Ungleichverteilung der Betriebsmittel auf einzelne Threads zeigt sich auch in der Kurve des aufsummierten Datendurchsatzes über alle Threads. Dieser Gesamtdatendurchsatz ist oberhalb von 60 Threads höher als der Datendurchsatz der Netzwerkschnittstelle (s. Abb. 4.28). Da die am meisten vom Betriebssystem begünstigten Threads den höchsten Durchsatz erzielen, können diese Threads ihre Meßphase schnell beenden.[73] Werden später andere Threads bevorzugt, können diese Threads kurzzeitig einen hohen Durchsatz erzielen, da die anderen Threads aufgrund der Benachteiligung kaum Anteile am Durchsatz der Schnittstelle beanspruchen. Die Synchronisationsmechanismen des Meßwerkzeuges reichen folglich bei mehr als 100 Threads nicht mehr aus, um die Korrektheit des Ergebnisses beim aufsummierten Datendurchsatz zu gewährleisten.

[72] Diese ungleichmäßige Verteilung geht einher mit dem Anstieg der CPU-Last auf dem Server (s. Abb. 4.27). Die CPU-Last gibt die über einen Zeitraum (hier 15 Minuten) gemittelte Anzahl von Prozessen oder Threads an, die gleichzeitig die CPU beanspruchen.

[73] Weiter verstärkt wird dieser Effekt durch die Auswirkungen der Überwachungsfunktion, die Transaktionen wegen längerer Inaktivität abbricht ("Deadlock-Checker"). Die übrigen Transaktionen erreichen dann bei einem weniger belasteten System einen höheren Datendurchsatz, so daß der Gesamtdatendurchsatz ansteigt.

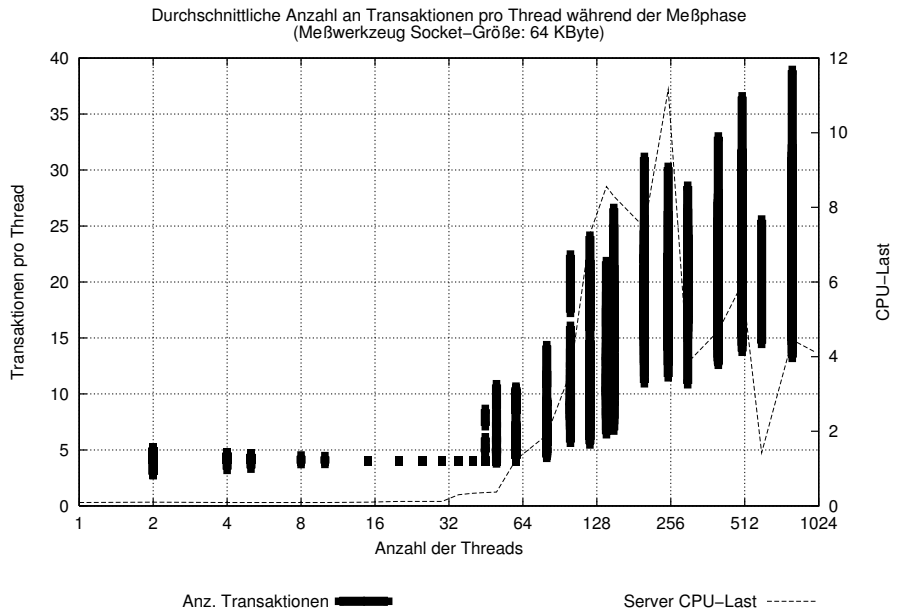


Abbildung 4.27: Transaktionen pro Thread während der Meßphase

Auch wenn der Gesamtdatendurchsatz und die Transaktionsdurchsätze bei Messungen mit mehr als 100 Threads nur sehr ungenaue Ergebnisse liefern, eignen sich Messungen mit mehreren 100 Threads trotzdem zur Ermittlung der maximalen Anzahl paralleler Verbindungen. Selbst mit 1.000 parallelen Threads bestätigen Stichproben mit `netstat`,[74] daß wirklich 1.000 Verbindungen gleichzeitig geöffnet sind.

Einschränkungen: Bei dieser Messung werden Transaktionen fester Dauer (30 Sekunden) durchgeführt, um es auch den übrigen Threads zu ermöglichen, innerhalb dieser Zeit ihre Verbindung zu eröffnen und so die geforderte Anzahl paralleler Verbindungen über mehrere Sekunden geöffnet zu halten. Bei Messungen mit sehr vielen Threads stellt sich dabei die Frage, ob über eine geöffnete Verbindung überhaupt nennenswerte Datenmengen übertragen werden, bevor die Verbindung nach 30 Sekunden wieder geschlossen wird. Die während einer Transaktion übertragene Datenmenge wird zwar vom Meßwerkzeug nicht ausgegeben, kann aber aus der Verbindungsdauer und dem erzielten Durchsatz berechnet werden. Hierbei zeigte sich, daß bei einer Messung mit 1.000 Threads immerhin noch 0,06 MBit/s bei einer einzelnen Transaktion erreicht werden. Bei einer Transaktionslänge von 30 Sekunden werden also über 200 KByte übertragen.

[74] `netstat` ist ein UNIX-Befehl, der eine Liste der aktiven Verbindungen erstellt.

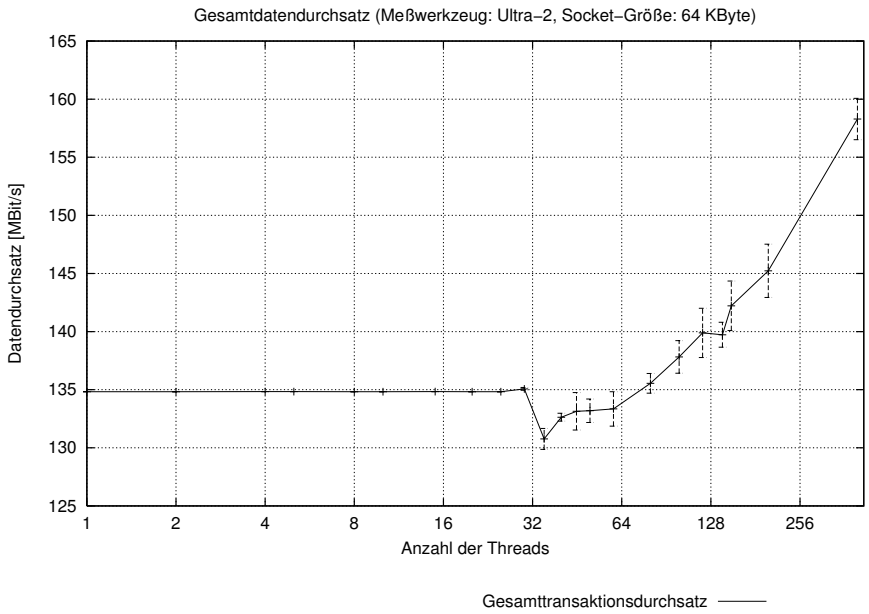


Abbildung 4.28: Aufsummierter Datendurchsatz über alle Threads

Die Befürchtungen, daß auf einer Verbindung keine nennenswerten Datenmengen mehr übertragen werden, ist also unbegründet.

Eine tatsächliche Einschränkung der Meßergebnisse ergibt sich hingegen aus dem verwendeten HTTP-Protokoll. Bei HTTP erfolgt nach der Anfrage des Clients eine Antwort des Servers. Die gesamte Datenmenge wird innerhalb kürzester Zeit unter Ausnutzung des zur Verfügung stehenden Datendurchsatzes gesendet. Eine HTTP-Verbindung erzeugt daher auf den Meßsystemen und zu vermessenden Proxy-Servern eine viel höhere Last als zum Beispiel interaktive Telnet- oder "rlogin"-Verbindungen, die auch Phasen längerer Inaktivität verzeichnen können und auf denen der Datendurchsatz zudem deutlich unterhalb des zur Verfügung stehenden Netzdurchsatzes bleibt. Es ist daher anzunehmen, daß die maximale Anzahl paralleler Verbindungen über einen Firewall bei Telnet und "rlogin" deutlich höher ist als bei HTTP.[75]

Weitere Beobachtungen: Beim Auftreten von "Time-Outs" werden die entsprechenden Verbindungen abgebrochen. Diese "Time-Outs" treten teilweise bereits beim Verbindungsaufbau auf. Ab 800 Threads meldet das System auf der

[75] Ein Meßwerkzeug, daß die Performanz bei interaktive Telnet- oder "rlogin"-Verbindungen ermittelt, kann auf der Basis des vorhandenen HTTP-Meßwerkzeuges entwickelt werden. Hierfür ist lediglich die Entwicklung einer geeigneten Sende-Routine erforderlich, die das Verhalten von interaktive Telnet- oder "rlogin"-Verbindungen emuliert.

Server-Seite die Häufung von "Time-Outs" als möglichen "Denial of Service"-Angriff. Bei der "SYN-Flooding"-Attacke (s. Abschnitt 2.1.1.4) sendet ein Angreifer lediglich das erste Segment eines TCP-Verbindungsaufbaus ("SYN"), ohne anschließend eine Verbindung aufzubauen. Bereits durch das Senden einer geringen Anzahl von SYN-Segmenten können die Betriebsmittel des Servers aufgebraucht werden. Im Falle der hier beschriebenen Messung kann das serverseitige System einen Angriff nicht von dem Verhalten bei einer Messung unterscheiden.

Fazit: Das Meßwerkzeug eignet sich mit bis zu 800 parallelen Verbindungen zur Ermittlung der maximalen Anzahl paralleler (HTTP-)Verbindungen durch einen Firewall. Mit bis zu 200 Threads laufen die Messungen stabil. Darüber können nur einzelne Messungen ohne Fehler beendet werden. Bei Messungen mit bis zu 50 Threads liefert diese Messung zudem verläßliche Ergebnisse über den Transaktionsdurchsatz. Mit bis zu 30 Threads stimmt auch der Gesamtdatendurchsatz mit dem Datendurchsatz der Schnittstelle überein.

4.2.7 Transaktionsdurchsatz bei parallelen Transaktionen

Neben dem Bedienen paralleler Verbindungen wird ein Firewall in noch viel stärkerem Maße beim Verbindungsaufbau belastet. Protokolle wie HTTP stellen selbst ohne aufwendige Authentisierung durch den Firewall wegen des häufigen Verbindungsauf- und -abbaus eine hohe Belastung dar. Im Gegensatz zu der im vorigen Abschnitt diskutierten maximalen Anzahl paralleler Verbindungen, bei der lange Transaktionen gewählt werden, sind für die Messung des maximalen Transaktionsdurchsatzes bei parallelen Transaktionen von jedem Thread sehr kurze Transaktionen zu wiederholen. Dies ergibt eine gänzlich andere Belastung.

In Abschnitt 4.2.4 wurde bereits eine Transaktionsmessung mit dem Meßwerkzeug "Netperf" vorgestellt. Mit "Netperf" werden minimale Transaktionen (Verbindungsaufbau, ein Byte Nachrichtenaustausch, Verbindungsabbau) sequentiell wiederholt. Aus der Anzahl der Transaktionen, die das Meßwerkzeug in einem festgelegten Zeitintervall durchführt, kann die Transaktionslänge und die Verbindungsaufbauzeit berechnet werden. Bei dieser Transaktionsmessung wird ein Hauptteil der Zeit mit Warten auf Antworten des Kommunikationspartners verbracht. Diese Zeit kann effizient für weitere Transaktionen verwendet werden. Daher ist zu erwarten, daß der Transaktionsdurchsatz paralleler Transaktionen über dem Transaktionsdurchsatz bei sequentiellen Transaktionen liegt.

Zur Messung der maximalen Anzahl paralleler Transaktionen werden, wie bei der Transaktionsmessung im Abschnitt 4.2.4, minimale Transaktionen durchgeführt. Bei HTTP sind ein Byte lange Nachrichten jedoch unter Einhaltung der Protokollvorschriften nicht möglich. Für die Anforderung einer HTML-Seite beim

Server benötigt der Meß-Client während der Messung eine etwa 35 Byte lange Nachricht ("GET-Request"). Der Meß-Server schickt daraufhin eine 250 Byte lange HTML-Seite zurück. Da bei der MTU-Größe von 9.180 Byte eine Dateneinheit zur Übertragung von Nachrichten der Länge von 1 bis 9.140 Byte ausreicht, sind keine gravierenden Auswirkungen durch diese etwas längeren Nachrichten zu erwarten. Jeder Thread führt mindestens 500 dieser Transaktionen in seiner Meßphase durch. Der Transaktionsdurchsatz wird für den Bereich von 1 bis 1.000 parallel laufender Threads ermittelt.[76]

Die Ergebnisse der Transaktionsmessung sind in Abbildung 4.29 wiedergegeben. Der Transaktionsdurchsatz bei einem Thread entspricht prinzipiell einer sequentiellen Transaktionsmessung. Aufgrund der größeren Transaktionslänge und der unterschiedlichen Implementation der Meßwerkzeuge erreicht der Durchsatz mit 803 ± 11 Transaktionen pro Sekunde nicht ganz den bei "Netperf" ermittelten Wert von 899 Transaktionen pro Sekunde (s. Tab. 4.4). Der Maximalwert von 1.796 ± 24 Transaktionen wird mit 12 parallelen Threads erreicht. Mit einem Thread kann bei den weniger leistungsfähigen "Ultra-1"-Systemen ein Transaktionsdurchsatz von 464 ± 2 Transaktionen pro Sekunde erreicht werden. Das Maximum von 625 ± 3 Transaktionen pro Sekunde wird bei den "Ultra-1"-Systemen im Bereich von 4 bis 6 Threads erreicht. Wie bereits bei der sequentiellen Transaktionsmessung zur Ermittlung der Verbindungsaufbauzeit (s. Abschnitt 4.2.4) hängt die Performanz auch bei dieser Messung primär von der Leistungsfähigkeit der Endsysteme ab.

Der ebenfalls in Abbildung 4.29 eingezeichnete Gesamtdatendurchsatz aller Threads zeigt denselben Verlauf wie die Kurve des Transaktionsdurchsatzes, weil beide Kurven über die feste Transaktionslänge miteinander zusammenhängen. Erst oberhalb von 140 Threads (ohne Abbildung) laufen beide Kurven nicht mehr exakt übereinander. Hier macht sich die unterschiedliche Meßdauer für den Datendurchsatz und die Anzahl der Transaktionen bemerkbar. Der Datendurchsatz wird immer für die geforderte Anzahl von Transaktionen (hier 500) berechnet, während bei der Transaktionsmessung auch die Anzahl der während der Auslaufphase durchgeführten Transaktionen mitgezählt werden. Solange alle Threads gleichmäßig und häufig genug vom Betriebssystem ausgeführt werden, bleibt die Auslaufphase kurz. Kann jedoch ein Thread nur wenig Fortschritt machen, benötigt er viel länger für die geforderte Anzahl an Transaktionen und die Auslaufphase verlängert sich entsprechend. Die durchschnittliche Anzahl der Transaktionen, die von einem Thread vom Beginn der Meßphase bis zum Ende der Auslaufphase durchgeführt werden mußten, sind in Abbildung 4.30 wiederge-

[76] Selbstverständlich muß bei dieser Transaktionsmessung, wie bereits bei der "Netperf"-Transaktionsmessung (s. Abschnitt 4.2.4), das Aufbrauchen der Portnummern verhindert werden. Das "`tcp_close_wait_interval`" wird deshalb auch bei dieser Messung auf 30 Sekunden verringert.

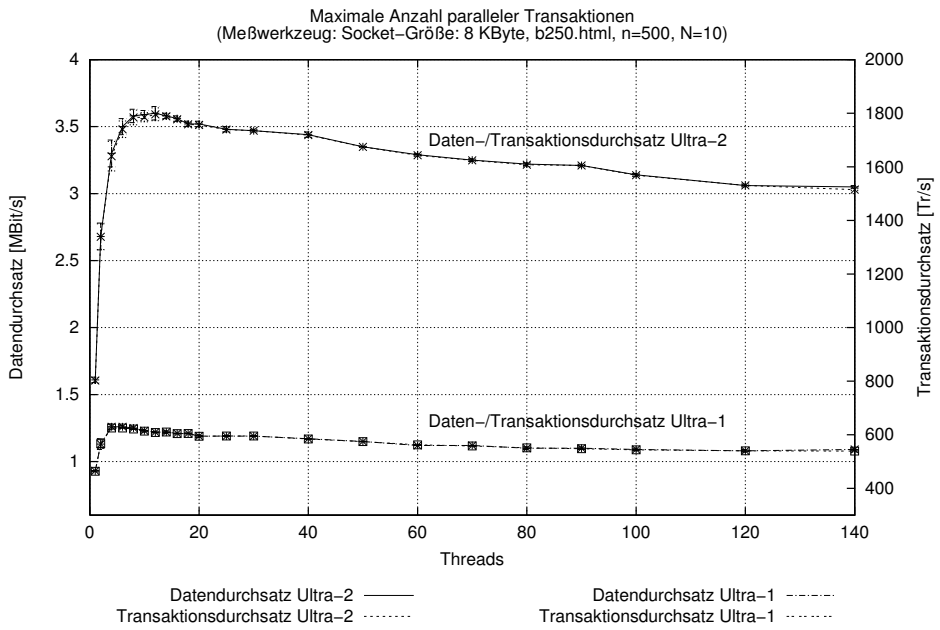


Abbildung 4.29: Transaktionsdurchsatz im Vergleich zum Datendurchsatz

geben. Auch hier steigt oberhalb von 120 Threads die durchschnittliche Anzahl der Transaktionen stark an.

Trotz dieses stetigen Abfalls des Transaktionsdurchsatzes werden mit einer "Ultra-2" selbst bei 1.000 Threads immer noch 1.387 ± 14 Transaktionen pro Sekunde erreicht. Stichproben bei weiteren Testmessungen bestätigen, daß tatsächlich auch hier noch die entsprechende Anzahl offener Verbindungen erreicht wird. Da der maximale Transaktionsdurchsatz jedoch bereits bei 12 Threads erreicht wird und ab 140 Threads die durchschnittlich notwendige Anzahl von Transaktionen zunimmt, ist bei den folgenden Transaktionsmessungen mit Firewall-Komponenten der Bereich bis einschließlich 120 Threads[77] am wichtigsten.

Auswirkungen von Implementationsentscheidungen: Setzen der Socket-Größe: Bei der Messung der maximalen Anzahl paralleler Verbindungen zeigte sich erneut, wie wichtig es ist, die Operationen minimal zu halten, die für jede Verbindung durchgeführt werden müssen. In der ersten Implementation reservierte das Meßwerkzeug für jede neue Verbindung jeweils 64 KByte Sende- und Empfangspuffer (Socket-Größe).[78] Erfolgt diese Anforderung für jede Verbindung

[77] Bei 120 Threads wird mit einer "Ultra-2" immer noch ein Gesamttransaktionsdurchsatz von 1.531 Transaktionen pro Sekunde erreicht.

[78] Zum Einstellen der Socket-Größe werden `setsockopt(..., SOL_SOCKET, SNDBUF, ...)` bzw. `RCVBUF` Funktionsaufrufe verwendet.

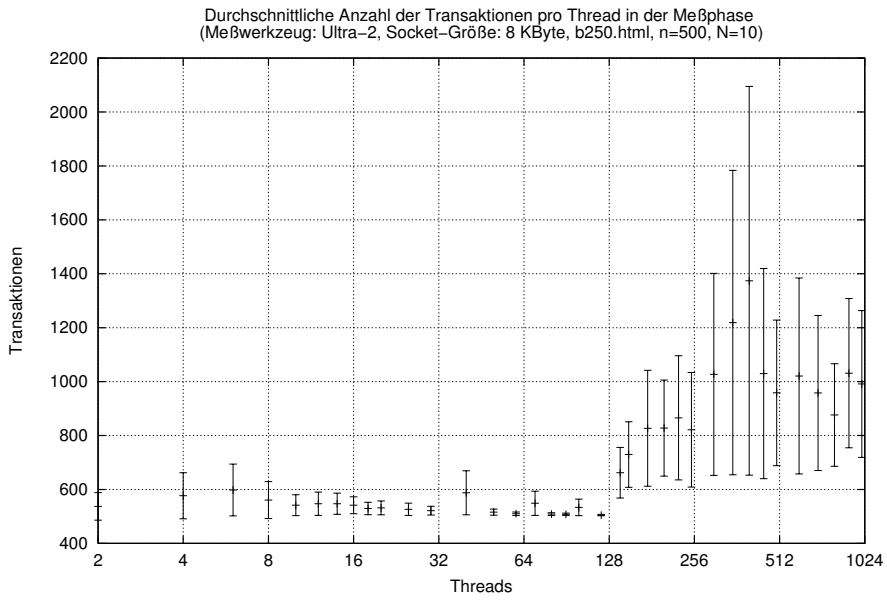


Abbildung 4.30: Durchschnittliche Anzahl der Transaktionen pro Thread

auf der Client- und der Server-Seite, so können mit dieser Implementation nur maximal 1.644 ± 7 Transaktionen pro Sekunde bei 10 Threads (zum Vergleich 1.796 Tr/s) erreicht werden. Mit einem Thread werden dann nur noch maximal 728 ± 1 Transaktionen pro Sekunde im Vergleich zu 803 Tr/s erreicht. Effizienter ist es stattdessen, die Default Socket-Größen vor der Messung auf beiden Systemen mit den gewünschten Werte einzustellen,[79] während der Messung kann auf die jeweiligen Aufrufe dann verzichtet werden. Bei Proxy-Servern sind in der Regel ähnliche Aufrufe enthalten, die hier ebenfalls den "Overhead" beim Verbindungsaufbau erhöhen.

Auswirkungen von Schlüsselaustausch und kryptographischer Authentisierung: Die bisher diskutierten Messungen dienen der Ermittlung des maximalen Transaktionsdurchsatzes bei einfachen HTTP-Transaktionen. Eine einzelne HTTP-Transaktionen stellt dabei für die Endsysteme eine vergleichsweise geringe Belastung dar; erst durch eine größere Anzahl paralleler Threads entsteht auf den Endsystemen eine nennenswerte Last. Dies sieht ganz anders aus, wenn zusätzlich eine kryptographische Authentisierung der Kommunikationspartner erforderlich wird und die Daten zudem vor der Übertragung verschlüsselt werden müssen.

[79] Die Sende- und Empfangs-Socket-Größen können über `ndd -set /dev/tcp tcp_xmit_hiwat` bzw. `tcp_recv_hiwat` systemweit eingestellt werden.

Um die Auswirkungen kryptographischer Verfahren auf die Performanz bewerten zu können, wird die SSL-Bibliothek "SSLeay"[80] in Meß-Client und -Server integriert. Die "SSLeay"-Bibliothek stellt Funktionen zum Einsatz der Protokolle SSL (Version 2 und 3) [Freier et al. 96] sowie TLS [Dierks et al. 97] zur Verfügung. Nach dem TCP-Verbindungsaufbau führen Meß-Client und -Server über diese SSL-Funktionen einen Schlüsselaustausch und eine Authentisierung durch. Die beim Schlüsselaustausch (über RSA oder "Diffie-Hellman") vereinbarten Schlüssel werden zur Sicherung von Vertraulichkeit und Integrität bei der nachfolgenden Datenübertragung eingesetzt. Zum Einsatz kommen dabei symmetrische Verschlüsselungsverfahren wie DES, 3DES, IDEA, RC2 und RC4 sowie kryptographische Prüfsummen wie MD5 und SHA-1. Die für eine Verbindung einzusetzenden Verfahren werden zwischen Client und Server anhand der vom Administrator konfigurierten Vorgaben ausgehandelt.

Eine Authentisierung ist bei SSL und TLS nur durch den Server beim Client erforderlich. Eine Authentisierung des Clients beim Server ist optional möglich, ist für den Ablauf des Protokolls jedoch nicht erforderlich. Da bei den heute gebräuchlichen Web-Servern und "Web-Browsern" ausschließlich eine Authentisierung des Servers beim Client erfolgt, wurde auf die Implementierung der optional zusätzlich möglichen Authentisierung des Clients beim Server vorerst verzichtet, um eine Interoperabilität des Meßwerkzeuges mit gebräuchlichen Web-Servern und "Web-Browsern" zu gewährleisten. Durch den Verzicht auf die Authentisierung des Clients beim Server ist es beispielsweise möglich, den Meß-Server mit gebräuchlichen "Browsern" wie "Netscape Communicator" oder "Microsoft Internet Explorer" zu testen. Durch eine Ergänzung von Meß-Client und -Server kann später bei geänderten Szenarien auch die Authentisierung des Clients beim Server implementiert und untersucht werden. Außerdem wurde bisher auch noch keine Überprüfung von Zertifikatsketten implementiert, so daß der Meß-Client bei der Authentisierung des Servers auf die Authentizität des vorliegenden öffentlichen Schlüssels des Servers vertrauen muß.

Für die Messungen zur Auswirkung der Authentisierung auf den Transaktionsdurchsatz werden die bereits vorgestellten Transaktionsmessungen mit 250 Byte langen HTML-Seiten durchgeführt. Eingesetzt wird dabei die derzeit gebräuchlichste SSL-Version (Version 3). Der Schlüsselaustausch und die Authentisierung des Servers erfolgen über RSA. Für die anschließende Datenübertragung werden 3DES und SHA-1 eingesetzt. Da zur Messung des Transaktionsdurchsatzes nur sehr kurze Transaktionen (250 Byte) durchgeführt werden, fällt der Aufwand zur Verschlüsselung bei der Datenübertragung im Verhältnis zur Authentisierung beim Verbindungsaufbau kaum ins Gewicht, so daß bei der hier durchgeführten Messung des maximalen Transaktionsdurchsatzes nur die Parameter des Schlüs-

[80] "SSLeay Version 0.9.0b" verfügbar unter
`ftp://ftp.cert.dfn.de/pub/tools/net/ssleay/SSLeay-0.9.0b.tar.gz`

selaustausches und der Authentisierung variiert werden müssen.[81] Als Parameter können die zum Schlüsselaustausch eingesetzten Verfahren (RSA oder "Diffie-Hellman") und die Schlüssellängen variiert werden.

Für Schlüsselaustausch und Authentisierung werden drei unterschiedlich lange RSA-Schlüssel (512, 1024 und 2048 Bit Schlüssellänge) generiert. Die Ergebnisse der Messung zwischen zwei "Ultra-2"-Systemen sind in Abbildung 4.31 dargestellt.

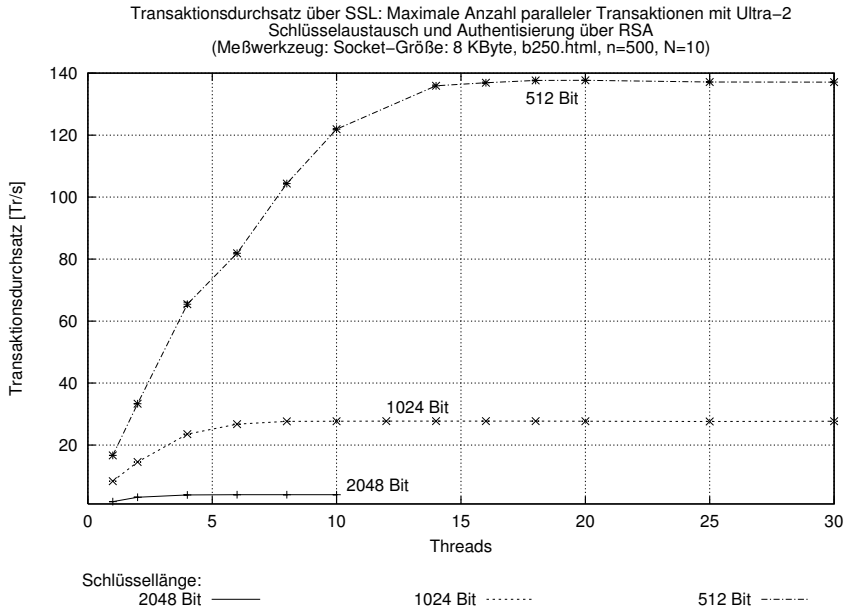


Abbildung 4.31: Transaktionsdurchsatz über SSL

Die Messung zwischen zwei "Ultra-2"-Systemen, bei der ein 2048 Bit langer Schlüssel für den Schlüsselaustausch und die Authentisierung verwendet wurde,

[81] Bei den 30 Sekunden langen Transaktionen, die zur Messung der maximalen Anzahl paralleler Verbindungen im Abschnitt 4.2.6 verwendet werden, überwiegt hingegen der Aufwand durch die Verschlüsselung der übertragenen Daten gegenüber dem Aufwand für den Schlüsselaustausch und die Authentisierung. In Ergänzung zu den bereits durchgeführten Messungen mit 30 Sekunden langen einfachen HTTP-Transaktion wäre es in Abschnitt 4.2.6 möglich gewesen, die Auswirkungen kryptographischer Verfahren auf den Datendurchsatz durch Messungen mit den von SSL unterstützten, unterschiedlichen Verfahren (DES, 3DES, IDEA, RC2 und RC4) zu ermitteln. Auf diese Messungen wurde jedoch aus zwei Gründen verzichtet. Zum einen konnte die SSL-Version des Meßwerkzeuges erst nach Ende der eigentlichen Meßexperimente fertiggestellt werden, so daß in der verbleibenden Zeit nur die wichtigsten Messungen durchgeführt werden konnten. Zum anderen sollen die Messungen Aufschluß über die Auswirkungen kryptographischer Verfahren auf die *Performanz von Firewalls* geben. Daß es hierfür nicht erforderlich ist, die Meßwerkzeuge um kryptographische Verfahren zu ergänzen, zeigen die Messungen im Abschnitt 4.4.2.8.

zeigt einen geringen Anstieg des Transaktionsdurchsatzes bis 6 Threads, bei denen der maximale Durchsatz von 3,97 ± 0,00 Transaktionen pro Sekunde erreicht wird. Diese Messung wurde nach der Meßreihe mit 10 Threads abgebrochen, da aufgrund des geringen Transaktionsdurchsatzes für jeden einzelnen Meßpunkt sehr viel Zeit benötigt wird, bis alle Threads die jeweils geforderten 500 Transaktionen durchgeführt haben. Zum Zeitpunkt des Abbruchs lief diese Messung bereits über 15 Stunden.[82] Auf eine Fortsetzung der Messung mit 2048 Bit langen Schlüsseln kann daher verzichtet werden.

Die Meßkurve mit der derzeit gebräuchlichsten Schlüssellänge von 1024 Bit zeigt zunächst einen Anstieg bis zum Meßpunkt mit 6 Threads, bei dem 26,74 ± 0,02 Tr/s erreicht werden, und steigt dann nur noch flach weiter an bis zum Meßpunkt mit 12 Threads, bei dem das Maximum von 27,73 ± 0,01 Tr/s erreicht wird. Dieser Wert wird auch beim letzten Meßpunkt (50 Threads) nur geringfügig mit 27,59 ± 0,01 Tr/s unterschritten.

Die Schlüssellänge von 512 Bit wird üblicherweise in den sogenannten "Export-Versionen" kryptographischer Software verwendet.[83] Mit 512 Bit langen Schlüsseln kann ein Anstieg bis 14 Threads (135,92 ± 0,26 Tr/s) beobachtet werden. Danach gibt es bis 20 Threads nur noch einen geringfügigen Anstieg bis zum maximalen Durchsatz von 137,68 ± 0,08 Tr/s.

Dieselben Meßreihen werden auch mit den weniger leistungsfähige "Ultra-1"-Systemen durchgeführt. Dabei verhinderte allerdings ein Fehler in der SSL-Bibliothek den Einsatz von 2048 Bit langen Schlüsseln beim Meß-Server, so daß nur die Messungen mit 512 und 1024 Bit langen Schlüsseln durchgeführt werden konnten.[84] Die Meßergebnisse mit den "Ultra-1"-Systemen liegen mit maximal 6,74 ± 0,00 Tr/s bei 1024 Bit langen Schlüsseln und 34,71 ± 0,01 Tr/s bei 512 Bit langen Schlüsseln um einen Faktor von etwa 4 unter den Ergebnissen der "Ultra-2"-Systeme. Da sich die Prozessor-Leistung der "Ultra-1"- und "Ultra-2"-Systeme ebenfalls um den Faktor 4 unterscheidet, zeigt dieser Vergleich, daß die Prozessor-Leistung der Endsysteme den Transaktionsdurchsatz beschränkt und Datendurchsatz oder Nachrichtenlaufzeiten im Netz keinen wesentlichen Einfluß auf diese Messung haben. Die maximalen Transaktionsdurchsätze in verschiedenen Konfigurationen sind in Tabelle 4.5 zusammengefaßt.

---

[82] Im gleichen Zeitraum kann mit einer Schlüssellänge von 1024 Bit bereits die Meßreihe mit 50 Threads abgeschlossen werden.

[83] Die US-amerikanischen Gesetze stufen kryptographische Verfahren als Waffen ein, deren Export engen Restriktionen unterliegt. Problemlos möglich ist derzeit nur der Export von Verfahren, deren Schlüssellängen nach heutigem Stand der Technik unzureichend sind. Beim RSA-Verfahren dürfen Schlüssellängen von bis zu 512 Bit aus den USA exportiert werden. Bei symmetrischen Verschlüsselungsverfahren wie DES und RC4 liegt die Grenze bei 40 Bit. Das Brechen von Schlüsseln dieser Länge gilt als leicht möglich.

[84] Der Versuch, den Fehler in der Bibliothek zu korrigieren, wurde als zu aufwendig verworfen.

	Transaktionsdurchsatz	
Schlüssellänge	Ultra-2 [Tr/s]	Ultra-1 [Tr/s]
2048 Bit	3,97 ± 0,00	—
1024 Bit	27,73 ± 0,01	6,74 ± 0,00
512 Bit	137,68 ± 0,08	34,71 ± 0,01
ohne SSL	1.796 ± 24	625 ± 3

Tabelle 4.5: Transaktionsdurchsatz bei RSA-Authentisierung

Eine Untersuchung der CPU-Last auf den Meßsystemen zeigt auch bei dieser Messung eine ungleichmäßige Verteilung zwischen Client und Server. Die Verbindungsannahme, die bei Messungen ohne SSL den Server sehr stark belastet und dadurch zu einer ungleichmäßigen Verteilung der Last führt, macht sich erst bei Messungen mit mehr als 30 Threads bemerkbar. Da bei den mit SSL durchgeführten Messungen nur Meßreihen bis maximal 30 Threads untersucht werden, können also Effekte durch die Verbindungsannahme vernachlässigt werden. Daß trotzdem auch bei Messungen mit SSL der Server stärker belastet ist als der Client, muß daher auf das SSL-Protokoll zurückzuführen sein. Dies hängt voraussichtlich damit zusammen, daß sich — dem üblichen Vorgehen bei SSL folgend — nur der Server beim Client authentisiert. Da die hierfür beim Server erforderliche Generierung von RSA-Signaturen deutlich aufwendiger ist als die beim Client erforderliche Überprüfung der RSA-Signaturen, ergibt sich die beobachtete ungleichmäßige Lastverteilung. Abbildung 4.32 zeigt die CPU-Last auf dem Meß-Server. Während der Messung bleibt die nicht abgebildete CPU-Last auf dem Meß-Client immer unter 0,5 (d. h. 50% Leerlauf) und sinkt sogar mit zunehmender Anzahl von Threads auf eine Last um 0,1 (d. h. 90% Leerlauf). Auf dem Client ergibt sich eine sinkende CPU-Last, da der Client länger auf Antworten des immer stärker belasteten Servers warten muß. So wird beispielsweise bei 2048 Bit langen RSA-Schlüsseln mit 10 Threads bereits eine CPU-Last von 10 auf dem Server erreicht. Dies bedeutet, daß alle 10 Threads fast ununterbrochen die CPU des Servers beanspruchen könnten, um Berechnungen mit den RSA-Schlüsseln durchzuführen. Die ungleichmäßige Verteilung der Last wird sich in der Praxis ungünstig bemerkbar machen, da typischerweise mehrere Clients mit demselben Server kommunizieren, während eine gleichzeitige Kommunikation eines Systems mit mehreren Servern praktisch nicht vorkommt. Somit wäre es günstiger, wenn der Server aufwendige Berechnungen an die Clients abgeben könnte.

Fazit: Wie erwartet kann bei parallelen Transaktionsmessungen ein höherer Transaktionsdurchsatz als bei der sequentiellen Transaktionsmessung erzielt werden. Der Maximalwert von 1.796 Tr/s wird bei 12 parallel laufenden Threads

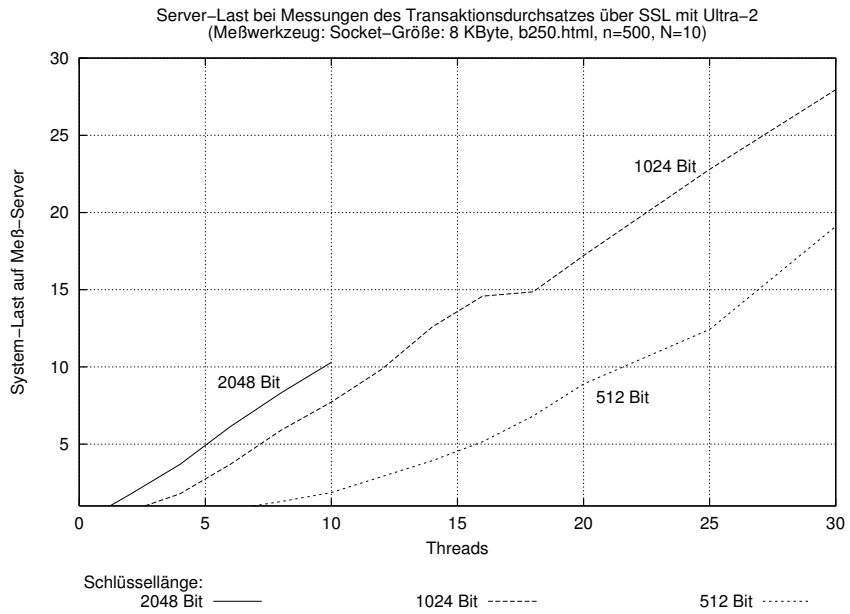


Abbildung 4.32: CPU-Last auf dem Meß-Server bei Transaktionsdurchsatzmessungen über SSL

erreicht und ist mehr als doppelt so hoch wie der bei der sequentiellen Transaktionsmessung erreichte Wert von 803 Transaktionen pro Sekunde. Obwohl das Meßwerkzeug Messungen mit bis zu 1.000 parallelen Threads erlaubt, soll für weitere Messungen der Meßbereich auf bis zu 120 Threads begrenzt werden, da für Messungen mit mehr als 120 Threads die Synchronisationsmechanismen nicht ausreichen. Schlüsselaustausch und kryptographische Authentisierungsverfahren haben sehr starke Auswirkungen auf den Transaktionsdurchsatz. Der maximale Transaktionsdurchsatz bei der gebräuchlichen Schlüssellänge von 1024 Bit erreicht lediglich knapp 28 Tr/s.

4.2.8 Transaktionsmessung mit längeren Transaktionen

Minimale Transaktionen eignen sich zur Bestimmung des maximalen Transaktionsdurchsatzes. In typischen Firewall-Umgebungen werden hingegen Transaktionen im Bereich von 2 bis 200 KByte überwiegen.[85] Die im vorigen Abschnitt beschriebene Messung zur Ermittlung des maximalen Transaktionsdurchsatzes bei parallelen Transaktionen soll daher in diesem Abschnitt durch längere Transaktio-

[85] Typische mittlere Transaktionslängen bei HTTP liegen im Bereich von 4 bis 8 KByte (s. Anhang C.1). Einige Transaktionen sind extrem groß (mehrere MByte Bild- oder Audio-Daten), was zu großen Standardabweichungen des Mittelwerts führt.

nen erweitert werden. Ziel ist es, sowohl den Datendurchsatz als auch den Transaktionsdurchsatz in Abhängigkeit von der Anzahl der parallelen Threads und in Abhängigkeit der Transaktionslänge zu bestimmen. Die Anzahl der Threads wird von 1 bis 140 variiert.[86] Der Meßbereich der Transaktionslängen beginnt bei 250 Byte[87] und reicht bis 512 KByte.

Abbildung 4.33 zeigt den aufsummierten Datendurchsatz (Gesamtdatendurchsatz) in Abhängigkeit von der Transaktionslänge für 1, 2, 10 und 20 Threads. Mit zunehmender Transaktionslänge sinkt der Anteil des Verbindungsaufbaus an der Gesamttransaktionslänge. Ein Thread (unterste Kurve) kann aber trotzdem selbst bei 512 KByte langen Transaktionen den zur Verfügung stehenden Netzdurchsatz nicht ausnutzen. Auffällig ist der Einbruch bei 64 KByte langen Transaktionen, bei dem der Durchsatz auf 10 MBit/s sinkt. Dieser Effekt wird im Abschnitt 4.2.9 untersucht.

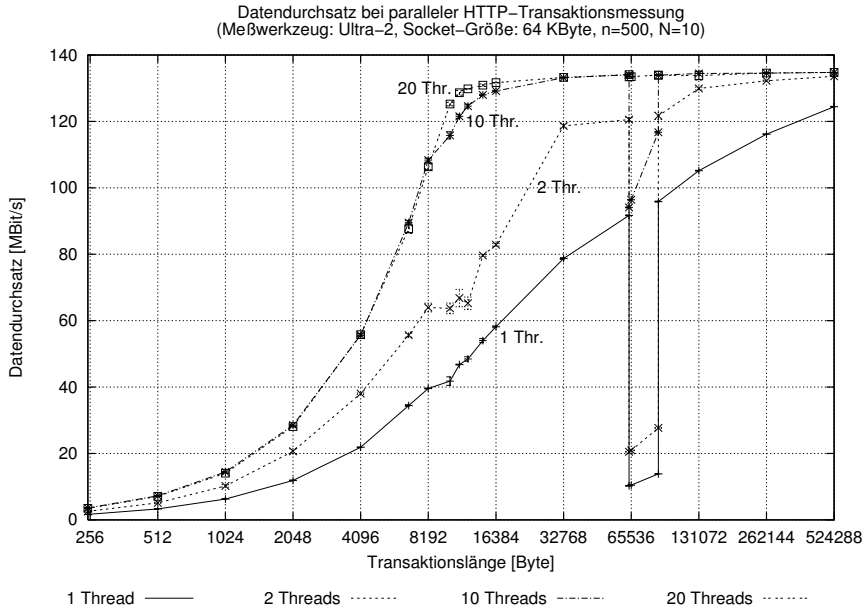


Abbildung 4.33: Gesamtdatendurchsatz bei längeren Transaktionen

Sind mehrere Threads gleichzeitig aktiv, so kann der aufsummierte Gesamtdurchsatz über alle Threads bei Transaktionslängen von mindestens 11 KByte mit 129 MBit/s den theoretisch möglichen maximalen Datendurchsatz fast erreichen.

[86] Im Vergleich zu Messung des maximalen Transaktionsdurchsatzes des vorigen Abschnitts wird bei der Messung in diesem Abschnitt die Anzahl der Threads begrenzt, da sich bereits gezeigt hat, daß oberhalb von 120 Threads die Meßgenauigkeit des Datendurchsatzes abnimmt.

[87] 250 Byte beträgt bei der HTML-Messung die minimale Transaktionslänge.

Der Verlauf der Kurven ab 20 Threads ähnelt dann dem Kurvenverlauf bei einer "netperf"-Messung über TCP (s. Abb. 4.4), was dafür spricht, daß unterhalb von 11 KByte Transaktionslänge die Leistungsfähigkeit der zur Messung verwendeten Systeme den maximale Datendurchsatz begrenzen. Oberhalb dieses Punktes wird der Datendurchsatz durch den maximalen Netzdurchsatz begrenzt. Die Kurve mit 20 Threads zeigt zudem bei 64 KByte Transaktionslänge keinen geringeren Durchsatz mehr als bei den benachbarten Meßpunkten, da bei 20 parallelen Threads auf jeden einzelnen Thread nur ein Durchsatz von unter 10 MBit/s entfällt. Dieser Durchsatz kann — wie die Messung mit einem Thread zeigt — auch trotz des Einbruchs an dieser Stelle erreicht werden. Die Kurven von 20 bis einschließlich 60 Threads überdecken sich und sind daher in Abbildung 4.33 nicht dargestellt. Bei diesen Messungen mit längeren Transaktionen nimmt bereits bei 70 Threads die Meßgenauigkeit ab, so daß im Bereich von 11 bis 64 KByte Transaktionslänge Durchsätze gemessen werden, die oberhalb des Datendurchsatzes der Netzschnittstelle liegen.

Der Transaktionsdurchsatz in Abhängigkeit von der Transaktionslänge ist in der Abbildung 4.34 wiedergegeben. Der bei der Messung mit minimalen HTTP-Transaktionen (s. Abschnitt 4.2.7) ermittelte Transaktionsdurchsatz von knapp unter 1.800 Transaktionen pro Sekunde bei 12 parallelen Threads kann bei der Kurve mit 10 Threads (s. Abb. 4.34) fast erreicht werden. Auch für längere Transaktionen, deren Länge aber noch deutlich unter der MTU-Größe von 9.180 Byte liegt, wird immer noch fast der gleiche Transaktionsdurchsatz erreicht.[88] Da die Daten mit einem Schreib- bzw. Lesezugriff vom Meßwerkzeug bearbeitet werden und anschließend in *einer* Dateneinheit verpackt werden, sind unterschiedliche Transaktionslängen kaum relevant.[89] Das Ergebnis hängt fast ausschließlich vom Transaktionsdurchsatz der beiden Endsysteme ab. Erst bei längeren Transaktionen beginnt der Netzdurchsatz den Transaktionsdurchsatz zu beschränken. Dies zeigt sich im Abfall des Transaktionsdurchsatzes bei Transaktionslängen oberhalb von 8 KByte in Abbildung 4.34.[90]

Auch bei Messungen mit mehr als 10 Threads sinkt der Transaktionsdurchsatz für Transaktionslängen im Bereich von 250 Byte bis 8 KByte nur geringfügig, da das Meßsystem durch die zusätzlichen Threads belastet wird, aber durch die höhere Parallelität kein höherer Durchsatz mehr erreicht werden kann. Alle Meßkurven mit mehr als 10 Threads zeigen denselben Kurvenverlauf wie die Kurve mit 10

[88] Bei geringeren MTU-Größen, etwa 1.500 Byte im Ethernet, ist hier entsprechend früher mit einer Abnahme des Transaktionsdurchsatzes zu rechnen.

[89] Da die vom Client zum Server übertragene URL ebenfalls unterhalb der MSS bleibt, kann auch diese in einer Dateneinheit übertragen werden. Entsprechend gering sind die Auswirkungen auf die Performanz.

[90] Der Einbruch bei 64 KByte Transaktionslänge konnte bereits bei der Auswertung des Datendurchsatzes (Abb. 4.33) beobachtet werden.

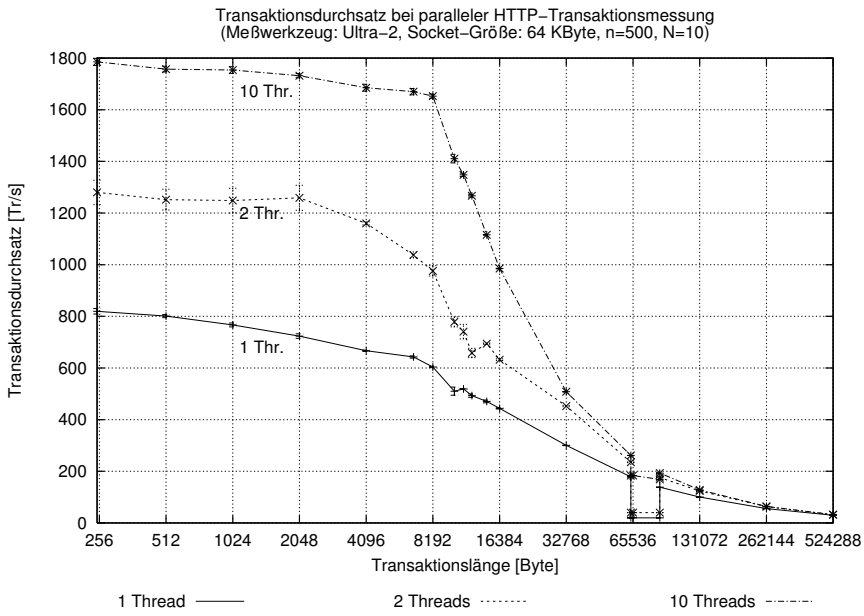


Abbildung 4.34: Gesamttransaktionsdurchsatz bei 1, 2 und 10 Threads

Threads in Abbildung 4.34. — auf eine Darstellung wird daher an dieser Stelle verzichtet.

Die Betrachtung des Gesamtdatendurchsatzes in Abhängigkeit von der Anzahl der Threads (s. Abb. 4.35) zeigt deutlich, wie wenig bei 250 Byte Transaktionslänge vom möglichen Durchsatz der Netzwerkschnittstelle verbraucht wird. Selbst bei Transaktionslängen von bis zu 8 KByte wird der Durchsatz bei gleichmäßigem Kurvenverlauf nur durch die Performanz der Endsysteme bestimmt. Anders ist der Verlauf ab der Transaktionslänge von 11 KByte, da hier mit 129 MBit/s der maximale Durchsatz der Netzwerkschnittstelle fast erreicht wird. Dieser Wert wird gleichmäßig auch bei noch größerer Anzahl paralleler Threads erreicht, da die zusätzliche Belastung durch weitere Threads von den Meßsystemen leicht verkraftet werden kann. Demgegenüber sinkt bei den kürzeren Transaktionen der Datendurchsatz mit der Zunahme der Threads wieder leicht ab, da der Datendurchsatz hier durch die Leistungsfähigkeit der Meßsysteme begrenzt ist und sich somit die größere Anzahl von Threads ungünstig auf den erreichten Datendurchsatz auswirkt. Bei Transaktionslängen ab 11 KByte werden die Auswirkungen der hohen Belastung der Meßsysteme erst oberhalb von 80 Threads erkennbar. Hier reichen bei den 11 und 16 KByte Transaktionslängen die Synchronisationsmechanismen des Meßwerkzeuges nicht mehr aus, so daß bei 80 bzw. 90 Threads Datendurchsätze oberhalb des Schnittstellendurchsatz gemessen werden. Bei den 512 KByte langen Transaktionen tritt eine etwas andere Belastung auf, da hier der Verbindungsaufbau eine geringere Rolle spielt, dafür aber die Schreib-

zugriffe der parallelen Threads synchronisiert werden müssen. Der Einbruch in der 512 KByte Kurve bei 40 bis 60 Threads entspricht daher dem bereits bei der Messung der maximale Anzahl paralleler Verbindungen (s. Abschnitt 4.2.6) beobachteten Einbruch (s. Abbildung 4.28).

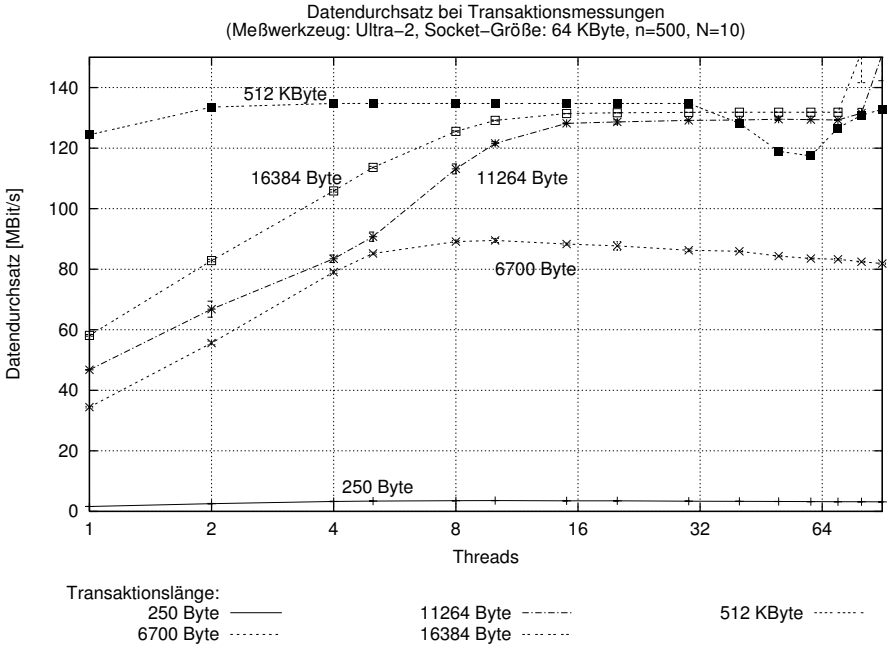


Abbildung 4.35: Datendurchsatz bei längeren Transaktionen

Fazit: Die Messungen mit Transaktionslängen von 250 Byte bis 512 KByte decken den Bereich der üblichen Transaktionslängen gut ab. Dabei zeigte sich, daß bei Transaktionslängen von 250 Byte bis 8 KByte ein Transaktionsdurchsatz von knapp 1.800 Tr/s erreicht werden kann. Bei längeren Transaktionen beginnt der Netzdurchsatz die Meßergebnisse zu beeinflussen. Die Verbindungsaufbauzeiten verhindern jedoch die Ausnutzung des zur Verfügung stehenden Netzdurchsatzes. Erst ab 10 Threads kann der Netzdurchsatz während der Verbindungsaufbauzeit durch parallele Threads ausgenutzt werden.

4.2.9 Tuning von TCP-Parametern

Die Meßwerkzeuge verwenden für die Übertragung von Daten das "Transmission Control Protocol" (TCP) [Postel 81b]. Die Hauptaufgabe dieses Protokolls ist es, auf Basis des verlustbehafteten, paketorientierten Internet-Protokolls (IP)

eine gesicherte[91] verbindungsorientierte Kommunikation zu ermöglichen. Durch das Einfügen von Sequenznummern kann die Vollständigkeit und korrekte Reihenfolge der Daten vom Empfänger überprüft werden. Der Empfänger bestätigt dem Sender den Erhalt der Daten über Empfangsbestätigungen ("Acknowledge", ACK). Die Flußkontrolle ist eine weitere wichtige Aufgabe von TCP, mit der eine Überlastung des Netzes verhindert werden soll.

Seit der Entwicklung und Optimierung von TCP haben sich die Einsatzbereiche entscheidend geändert. Die Wechselwirkungen zwischen Hochgeschwindigkeitsnetzen und TCP können bei ungünstiger Wahl der TCP-Parameter zu hohen Performanzverlusten führen [Comer et al. 94, Heidemann 97, Moldeklev et al. 95].

Flußkontrolle: Bei der Überlastung eines Netzes ("Congestion") müssen von Netzkomponenten (z. B. Routern) Dateneinheiten vernichtet werden. Da diese Dateneinheiten von TCP erneut gesendet werden müssen, kann eine erneute Überlastung und Vernichtung von Dateneinheiten die Folge sein. TCP versucht daher eine Überlastung des Netzes zu vermeiden ("Congestion Avoidance"), indem es nur so viele Daten sendet, wie vom Netz übertragen und vom Empfänger aufgenommen werden können. Die sogenannte "Fenstergröße" gibt die Datenmenge an, die gesendet werden darf, bevor auf eine Empfangsbestätigung gewartet werden muß. Durch Anpassungen der Fenstergröße ist es möglich den Durchsatz des Senders an den Netzdurchsatz anzupassen. Bei Hochgeschwindigkeitsnetzen sind große Fenster besonders wichtig, da sehr große Datenmengen vom Netz übertragen werden können, bevor die Empfangsbestätigung beim Sender ankommen kann (vgl. "Bandwidth x Delay Product", Seite 223). Da in dem Testnetz eine Überlastung der Netzkomponenten ausgeschlossen werden kann, ist die Fenstergröße hier nur durch den eigenen Sendepuffer ("Send-Socket-Size") und durch den Empfangspuffer ("Recv-Socket-Size") des Kommunikationspartners begrenzt.

Da am Anfang einer neuen Verbindung noch keine Aussagen über den Netzdurchsatz gemacht werden können, beginnt der Sender zunächst mit der konservativen Annahmen, daß maximal ein Datensegment maximaler Länge ("Maximum Segment Size", MSS) gesendet werden darf. Im folgenden Verlauf der Verbindung wird dann nach Erhalt jeder Empfangsbestätigung die Fenstergröße verdoppelt, so daß im folgenden also 2, 4 und 8 Segmente unbestätigt gesendet werden können. Dieses langsame Öffnen des Fensters wird als "Slow-Start" [Stevens 97] bezeichnet.

[91] "Gesichert" bedeutet hier, daß auch bei Verlust oder Verzögerung von IP-Datagrammen alle über TCP gesendeten Daten beim Empfänger in der richtigen Reihenfolge ankommen. Hierfür ist es notwendig, den Verlust von Dateneinheiten zu erkennen und ein wiederholtes Senden verlorener Dateneinheiten auszulösen. Gegen beabsichtigte Manipulationen der Dateninhalte bietet TCP keinen Schutz.

Aggregation ("Nagle"): Über den bereits im Abschnitt 4.2.2.2 beschriebenen "Nagle"-Algorithmus [Nagle 84] versucht TCP, die Übertragung vieler kleiner Segmente zu verhindern, indem mehrere kleine Schreibzugriffe zu einer größeren Dateneinheit aggregiert werden.

Der "Nagle"-Algorithmus erzielt die besten Ergebnisse bei interaktiven Verbindungen (Telnet, "rlogin" usw.) bei denen jeder Tastendruck des Benutzers in einem Schreibzugriff resultiert und somit viele kleine Schreibzugriffe zu größeren Dateneinheiten aggregiert werden können. Soll wie bei der HTTP-Messung auf einer TCP-Verbindung hingegen eine feste Datenmenge übertragen werden, verwenden Applikationen in der Regel bereits Schreibzugriffe, die nah an der maximalen Segmentlänge liegen, so daß der "Nagle"-Algorithmus nicht wirken kann.

Zusammenspiel mit Applikationen ("Deferred Acknowledge"): Eine weitere Funktion von TCP ist die Verzögerung der Bestätigung ("Deferred Acknowledge") [Clark 82], mit der einer Applikation Zeit gegeben werden soll, auf neue Daten zu reagieren. Viele Anwendungen senden kurz nach dem Empfang bereits eine Antwort an den Sender zurück. Bei der Übertragung der Antwort kann die verzögerte Bestätigung für vorher erhaltene Dateneinheiten mitgesendet und so zur Reduktion des Verkehrsaufkommens beigetragen werden.

Auf Verbindungen mit (überwiegend) unidirektionalem Datenfluß, wie den bei der Messung verwendeten HTTP-Transaktionen oder bei FTP-Datenverbindungen, kann dieses Verfahren keinen Performanzgewinn bringen, weil der Empfänger der Daten nie Antworten zurücksendet.

Es ist zu vermuten, daß der in Abbildung 4.33 im Bereich um 64 KByte Transaktionslänge beobachtete Einbruch durch eine oder mehrere der beschriebenen Mechanismen in TCP verursacht wird. Um die Ursache genauer zu untersuchen, wird die Transaktionsmessung für verschiedene TCP-Parameter wiederholt. Dabei reicht es aus, die Messungen mit nur einem Thread zu wiederholen, da hierbei das Problem am deutlichsten erkennbar ist.

Durch die Verringerung des "Deferred-Acknowledge-Intervalls" (Solaris TCP-Parameter: `tcp_deferred_ack_interval`) vom Standardwert 50 Millisekunden (ms) auf den bei Solaris minimal zulässigen Wert von 1 ms wird die Zeit verkürzt, mit der TCP das Senden einer Bestätigung verzögern kann. Das "Deferred-Acknowledge-Intervall" wird nur auf der Workstation verringert, auf der der HTTP-Client ausgeführt wird. Da bei der Messung nur vom Server zum Client nennenswerte Datenmengen fließen, können TCP-Parameter, die den Empfang von Dateneinheiten optimieren, nur Client-seitig zu meßbaren Wirkungen führen. Die Folge dieser Einstellung ist eine Verringerung des Durchsatzeinbruchs in dem untersuchten Bereich (s. Abb. 4.36).

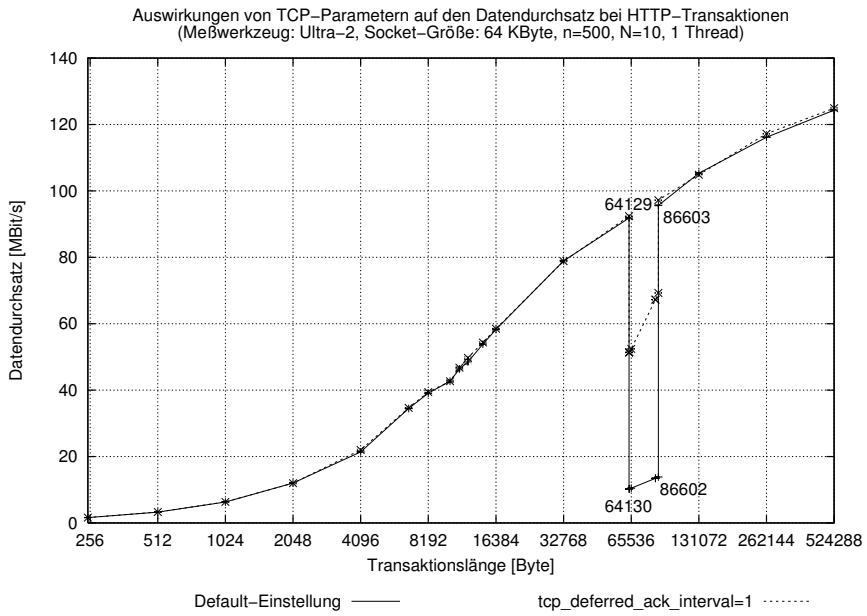


Abbildung 4.36: Auswirkungen von TCP-Parametern auf den Datendurchsatz: "Deferred-Acknowledge-Intervall"

Da die Verzögerung der Bestätigung über die Reduktion des Intervalls nicht vollständig verhindert werden kann, bleibt der Einbruch immer noch erkennbar. Nach [Braden 89] muß spätestens jedes zweite eintreffende Segment bestätigt werden. Die TCP-Implementation von Solaris erlaubt bei TCP-Verbindungen zwischen Systemen, die im selben lokalen Netz angeschlossen sind, sogar eine Bestätigung nur jedes achten Segments. Bei einer MSS von 9.140 Byte können mit 7 Segmenten 63.980 Byte übertragen werden. Zusammen mit dem ersten 149 Byte langen Datensegment, mit dem das Meßwerkzeug den Kopf einer HTML-Seite in einem einzelnen Schreibzugriff überträgt, ergibt sich eine Datenlänge von 64.129 Byte, die in 8 Segmenten maximal übertragen werden können. Diese Zahl stimmt mit dem Beginn des Einbruchs in Abbildung 4.36 exakt überein. Bevor das für Transaktionslängen über 64.129 Byte notwendige neunte Segment gesendet werden kann, muß auf eine Bestätigung des Empfängers gewartet werden.

Die maximale Anzahl der Segmente, nach der eine Bestätigung gesendet werden muß, kann bei Solaris über den TCP-Parameter `tcp_deferred_acks_max` gesteuert werden. Wird der Wert auf 1 reduziert, muß jedes eintreffende TCP-Segment sofort bestätigt werden. Durch die Bestätigung jedes Segments wird ab 32 KByte insgesamt eine Erhöhung des Datendurchsatzes erzielt (s. Abb. 4.37). Trotzdem bleibt ein kleiner Bereich (84.064 – 86.602 Byte) und zusätzlich kommt im oberen Bereich (523.278 – 525.322 Byte) ein weiterer Einbruch hinzu.

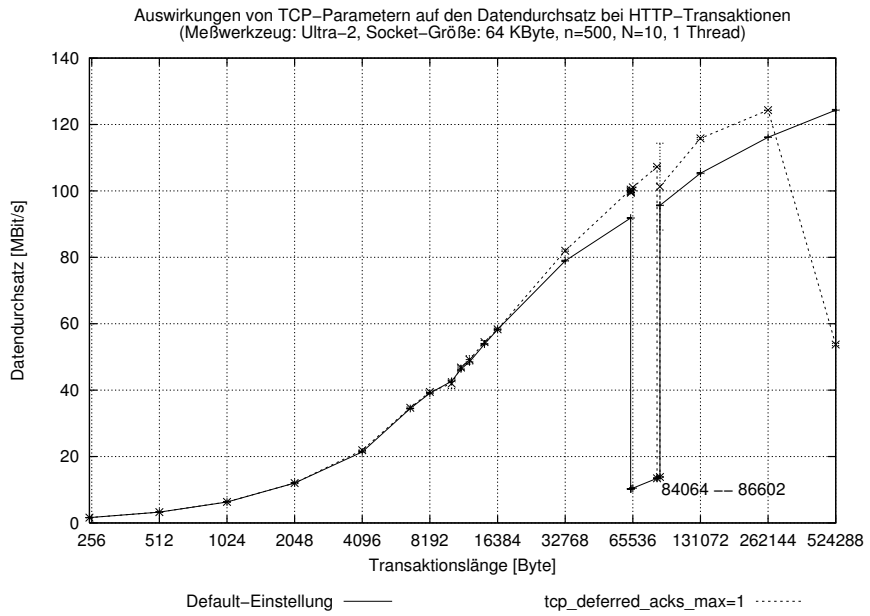


Abbildung 4.37: Auswirkungen von TCP-Parametern auf den Datendurchsatz: "Deferred Acknowledge"

Diese beiden Einbrüche im Meßbereich lassen sich durch eine Vergrößerung der Sende- und Empfangspuffer von 64 KByte auf 256 KByte vermeiden (s. Abb. 4.38). Weitere Einzelmessungen zeigen, daß die Vergrößerung des *Sende*puffers auf dem Server zur Vermeidung der Einbrüche ausreicht und Vergrößerungen des Empfangspuffers keine Auswirkungen zeigen. Daher ist anzunehmen, daß trotz sofortiger Empfangsbestätigung durch den Empfänger die Applikation auf der Sendeseite kurzzeitig wegen eines gefüllten Puffers gestoppt werden muß. Bei größerem Puffer kann die gesamte Datenmenge an TCP übergeben werden und das Problem tritt nicht auf.

Die Erhöhung der Socket-Größe allein bringt nicht dasselbe Ergebnis. Abbildung 4.39 zeigt, daß eine Vergrößerung der Socket-Größe ohne gleichzeitige Verhinderung von "Deferred-ACKs" den Einbruch sogar noch über den Meßbereich hinaus verlängert.

Nachdem das "Deferred-ACK" zusammen mit der unzureichenden Größe des Sendepuffers als Ursache identifiziert wurde, sollen die Auswirkungen der anderen Parameter nur kurz erwähnt werden.

Das "Slow-Start"-Verfahren zeigt in erste Linie Auswirkungen bei kürzeren Transaktionen. So wird durch die Erhöhung des Parameters `tcp_slow_start_initial` von 1 auf 4 dem Sender erlaubt bis zu 4 TCP-Segmente ohne Abwarten der

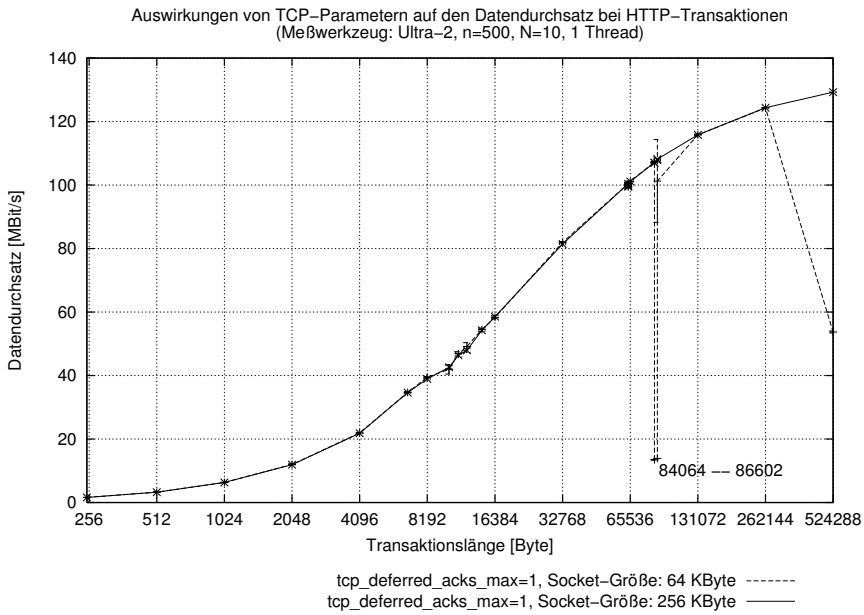


Abbildung 4.38: Auswirkungen von TCP-Parametern: "Deferred Acknowledge" bei 256 KByte Socket-Größe

Bestätigung sofort zu senden. Dadurch wird der Punkt, bei dem aufgrund ausbleibender Bestätigungen nicht mehr weitergesendet werden kann, früher erreicht.

Die Verzögerung des Sendens über das "Nagle"-Verfahren, um durch Aggregation größere Segmente senden zu können, hat keine Auswirkungen auf die HTTP-Transaktionen. Der erste Schreibzugriff, in dem das Meßwerkzeug den HTML-Kopf[92] überträgt, muß trotz geringer Länge sofort übertragen werden, da noch keine unbestätigten Segmente ausstehen. Anschließende Schreibzugriffe entsprechen der MSS und können daher auch trotz eventuell noch ausstehender Bestätigungen nicht zur Aggregation verzögert werden. Am Ende der Transaktion erfolgen zwar eventuell zwei kurze Schreibzugriffe des Meßwerkzeuges,[93] bei denen eine Aggregation stattfinden kann, da die Verbindung von der Applikation aber sofort danach geschlossen wird, können auch diese Daten nicht durch das "Nagle"-Verfahren aufgehalten werden.

[92] Der HTML-Kopf enthält die nach dem HTML-Standard [Berners-Lee et al. 95] erforderlichen Deklarationen, die bei der vom Meßwerkzeug generierten HTML-Seite eine Länge von 149 Byte haben.

[93] Jede vom Meßwerkzeug generierte Seite endet mit einem festgelegten Text mit einer Länge von 99 Byte. Um insgesamt die geforderte Transaktionslänge zu erreichen, paßt das Meßwerkzeug die Länge des letzten Schreibzugriffs, der vor dem Seitenende erfolgt, an. Dieser Schreibzugriff kann daher kleiner als die MSS sein.

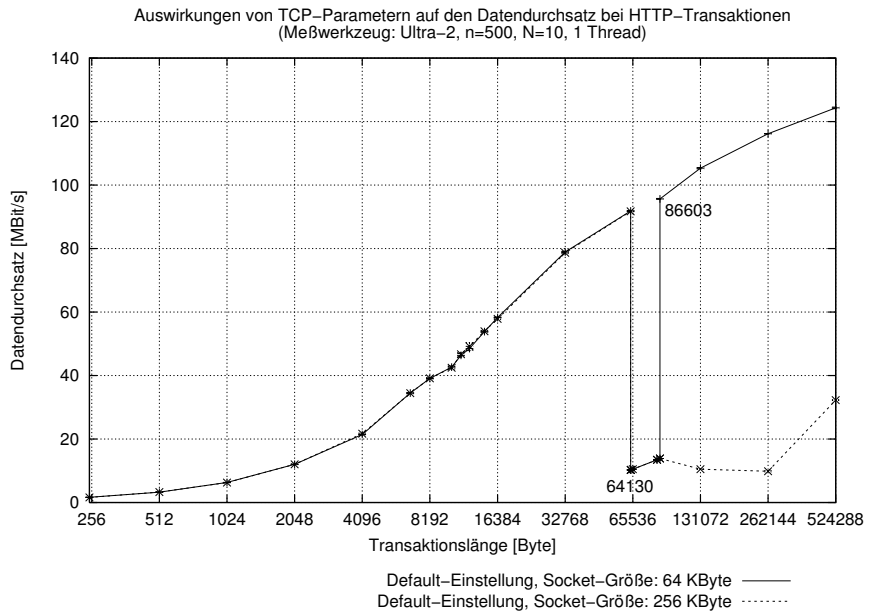


Abbildung 4.39: Auswirkungen von TCP-Parametern: Socket-Größe

Fazit: Die Messungen haben gezeigt, daß "Deferred Acknowledges" bei bestimmten Transaktionslängen zu erheblichen Einbrüchen führen können. Da "Deferred Acknowledges" bei unidirektionalem Datenverkehr keine Vorteile erzielen können, ist es empfehlenswert, dieses Verfahren abzuschalten. Des weiteren hat sich erneut gezeigt, wie wichtig ausreichend große Socket-Größen in Hochgeschwindigkeitsnetzen sind. Die Wahl geeigneter Default-Parameter bei Firewalls kann die Performanz auf einer einzelnen Verbindungen steigern. Bei den für die Messung der Firewall-Performanz wichtigeren Messungen mit einer größeren Anzahl paralleler Verbindungen zeigen diese Optimierungen auf den erzielten Gesamtdurchsatz keine Auswirkungen.

Nachdem in diesem Abschnitt die Messungen vorgestellt und die jeweiligen Eckwerte ermittelt wurden, folgt in den nächsten beiden Abschnitten eine Analyse der Performanz der beiden wichtigsten Firewall-Komponenten — Packet Screen und Proxy-Server.

4.3 Performanz von Packet Screens

Die im vorigen Abschnitt beschriebenen Systeme des Testnetzes werden für diesen ersten Firewall-Performanztest in zwei virtuelle Netze geteilt. Beide virtu-

ellen Netze nutzen denselben ATM-Switch.[94] Eine mit zwei ATM-Schnittstellen ausgestattete Workstation (`bora`) dient als Packet Screen. Die Software "IP-Filter" (Version 3.2)[95] ermöglicht die Spezifikation von einfachen bis zu höchst komplexen Filterregeln. Die Konfiguration des Testnetzes ist in Abbildung 4.40 wiedergegeben.

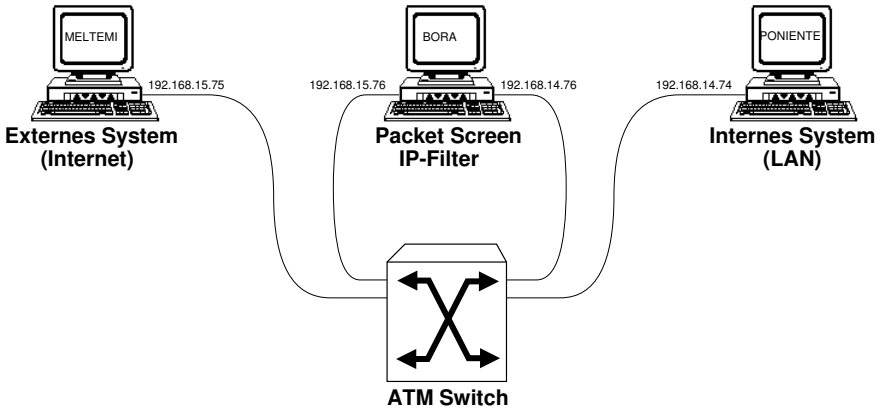


Abbildung 4.40: Testnetz-Konfiguration für Packet Screen Messungen

Bei der Performanz einer Packet Screen sind Datagrammdurchsatz, Datendurchsatz und Nachrichtenlaufzeit zu ermitteln. An einem Verbindungsaufbau ist eine Packet Screen nicht direkt beteiligt, daher sind Verbindungsaufbauzeiten und Transaktionsdurchsätze nicht relevant. Da Dateneinheiten bei einer statischen Packet Screen unabhängig von den bereits vorher verarbeiteten Dateneinheiten ausgewertet werden, reicht es aus, Messungen über eine einzelne TCP-Verbindung durchzuführen. Demgegenüber wäre dieses Vorgehen bei einer dynamischen Packet Screens (s. Abschnitt 2.3.1.1) nicht ausreichend, da abhängig von bereits empfangenen Dateneinheiten unterschiedliche Entscheidungen getroffen werden. Die Belastung durch das Anlegen und Löschen von Kontexten für Datagrammflüsse oder Verbindungen kann eine Performanzmessung auf nur einer Verbindung nicht messen. Für Performanzmessungen bei dynamischen Packet Screens sind die für Proxy-Server entwickelten Meßwerkzeuge (s. Abschnitt 4.2.5) besser geeignet.

Für die Messungen werden dieselben Meßskripte mit Variation von Socket-Größe und Schreibzugriffslänge genutzt, mit denen bereits die Meßreihen zur Ermitt-

[94] Diese Nutzung desselben ATM-Switches hat keine Folgen für die Performanz dieses Aufbaus, da der "Backplane"-Durchsatz des Switches für mehrere 622 MBit/s ATM-Schnittstellen ausgelegt ist und somit die zwei 155 MBit/s Netze keine wesentliche Last darstellen. Auf die ebenfalls mit der gemeinsamen Nutzung desselben Switches verbundenen Sicherheitsprobleme wurde bereits in Abschnitt 3.4.2 eingegangen.

[95] Verfügbar über `ftp://coombs.anu.edu.au/pub/net/ip-filter/`

lung der Eckdaten der Systeme des Testnetzes ermittelt wurden (s. Abschnitt 4.2). Da die Belastung der Packet Screen von der Anzahl der IP-Dateneinheiten abhängt, werden die Messungen bei der Packet Screen mit reduzierter MTU-Größe durchgeführt, wodurch die Aggregation mehrerer Schreibzugriffe zu einem großen TCP-Segment verhindert wird.

Die Performanz von Packet Screens hängt primär von der Anzahl der konfigurierten Filterregeln ab. Je mehr Filterregeln von der Packet Screen ausgewertet werden müssen, um so mehr Zeit wird für die Überprüfung jeder einzelnen Dateneinheit benötigt. Mehrere Sätze von Filterregeln wurden für die Performanzmessungen der Packet Screen entwickelt. Die Filterregeln sind so aufgebaut, daß die letzte Filterregel das Weiterleiten der Dateneinheiten der Messung erlaubt. Dadurch wird sichergestellt, daß für jede übertragene Dateneinheit alle Filterregeln ausgewertet werden müssen. Diese Vorgehensweise orientiert sich an dem im RFC 1944 [Bradner et al. 96] vorgeschlagenen Vorgehen. Dieser Aufbau der Filterregeln wäre bei "IP-Filter" im Prinzip jedoch nicht notwendig, da "IP-Filter" immer alle Filterregeln auswertet und die Anweisung der letzten auf die Dateneinheit zutreffenden Filterregel schließlich die Entscheidung der Packet Screen bestimmt.[96] Unzureichend erweisen sich die im RFC 1944 geforderten Messungen mit einer und mit 25 Filterregeln; abweichend hiervon wurden Messungen mit bis zu 1.000 Filterregeln durchgeführt. Die Angabe "x Filterregeln" bezieht sich dabei auf die Filterregeln mit "`block`"-Anweisungen, die vor der letzten Zeile mit der "`pass`"-Anweisung für die zur Messung verwendeten Datagramme stehen.[97] Die "`pass`"-Anweisung gilt für beide Übertragungsrichtungen, für die zur Messung gesendeten Dateneinheiten vorgegebener Länge und für die in Gegenrichtung gesendeten Bestätigungen.[98]

Da die meisten Packet Screen Konfigurationen asymmetrisch sind — interne Hosts dürfen alles, externe Hosts werden stark reglementiert — ist zu erwägen,

---

[96] Da immer alle Filterregeln überprüft werden müssen, ist es bei dieser Packet Screen Implementation auch nicht möglich, durch geschickte Sortierung der Filterregeln eine Optimierung zu erreichen. Bei anderen Packet Screen Implementationen ist es üblich, die Auswertung der Regeln abzubrechen, sobald eine Filterregel zutrifft. Bei diesen Implementationen kann durch eine Sortierung, die zu einer möglichst frühzeitigen Auswertung der häufiger zutreffenden Regeln führt, eine Performanzoptimierung erreicht werden. Wenn die erste Regel Dateneinheiten, die zu einer bereits offenen TCP-Verbindung gehören, betrifft, kann ein sehr hoher Performanzgewinn erreicht werden. Natürlich muß bei der Sortierung immer darauf geachtet werden, daß zwischen einzelnen Regeln Abhängigkeiten bestehen können, die abhängig von der Reihenfolge zu unterschiedlichen Filterergebnissen ("Allow","Deny") führen können. (vgl. [Ellermann 94, S. 102])

[97] "100 Filterregeln" bezeichnet eine Filterkonfiguration mit 100 Zeilen "`block`"-Anweisungen, gefolgt von einer Zeile mit einer "`pass`"-Anweisung für die vom Meßwerkzeug gesendeten Dateneinheiten. Die "`pass`-Zeile wird also nicht mitgezählt. Dies hat sich im nachhinein als ungünstig erwiesen, da diese Regel ja effektiv ausgewertet werden muß.

[98] Beispiele für die eingesetzten Filter sind im Anhang C.2 enthalten.

über die Anforderungen in [Bradner et al. 96] hinaus auch die Performanz von asymmetrischen Konfigurationen zu untersuchen. Dabei trifft für die Datagramme der Meßverbindung in einer Richtung bereits die erste Filterregel zu. In der anderen Übertragungsrichtung trifft die letzte Filterregel zu. Bei der hier verwendeten Packet Screen Software "IP-Filter" wird die Auswertung der Filterregeln jedoch nicht bei der ersten zutreffenden Regel abgebrochen, so daß unterschiedliche Positionen der Filterregeln in der Auswertungsreihenfolge keine Auswirkungen auf den erreichbaren Durchsatz haben.

4.3.1 Datendurchsatz über eine Packet Screen

Bei den Messungen zum maximalen Datendurchsatz über eine Packet Screen wurde zunächst ohne konfigurierte Filterregeln gemessen. Wie Abbildung 4.41 zeigt, kann über die Packet Screen bereits ohne konfigurierte Filterregeln unterhalb einer Schreibzugriffslänge von 2.048 Byte nicht mehr der volle Durchsatz erreicht werden. Erst im Bereich von 2.048 Byte bis zur MTU-Größe überdecken sich die beiden Kurven, und es wird auch durch die Packet Screen ein maximaler Durchsatz von über 134 MBit/s erreicht.

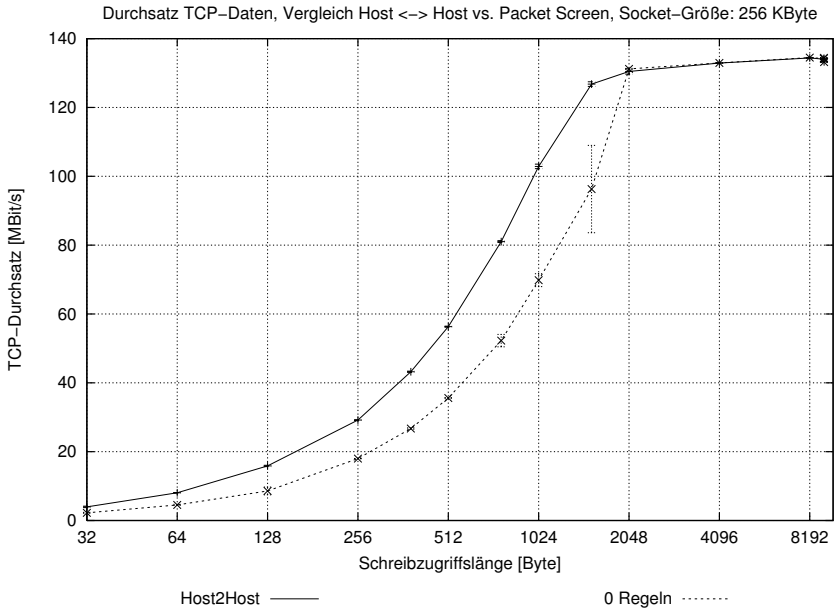


Abbildung 4.41: Durchsatzmessungen TCP über Packet Screen ohne Filterregeln

Dieses erste Ergebnis zeigt, daß die reine Datenübertragung durch die Packet Screen kein Problem ist, da auch über die Packet Screen der maximale Durchsatz von 134 MBit/s erreicht wird. Jedoch bedeutet der geringere Durchsatz bei

kürzeren Schreibzugriffslängen, daß über die Packet Screen nicht so viele Datagramme pro Sekunde übertragen werden können wie auf einer normalen Host-zu-Host-Verbindung. In Abbildung 4.12 (S. 239) wurde bereits dargestellt, wie bei kürzeren Datagrammlängen der zum Erreichen des maximalen Datendurchsatzes erforderliche Datagrammdurchsatz steigt. Um mit zwei KByte langen Nutzdatenlängen einen Durchsatz von 130 MBit/s zu erreichen, müssen nach Gleichung 4.16 ungefähr 8.000 Datagramme pro Sekunde verarbeitet werden. Die direkte Host-zu-Host-Messung erreicht bei 1.536 Byte Datagrammlänge immer noch einen Durchsatz von 127 MBit/s, was über 10.000 Datagrammen pro Sekunde entspricht. Durch die Packet Screen wird bei dieser Datagrammlänge demgegenüber nur noch ein Datendurchsatz von knapp 100 MBit/s erreicht, was einem Datagrammdurchsatz von etwas über 8.000 Datagrammen pro Sekunde entspricht. Dies spricht dafür, daß der maximale Durchsatz der Packet Screen bei etwa 8.000 Datagrammen pro Sekunde liegt. Im Vergleich zu einer direkten Host-zu-Host-Verbindung hat sich der Datagrammdurchsatz durch die Packet Screen halbiert. Die Auswirkungen einer Packet Screen auf den Datagrammdurchsatz werden im Abschnitt 4.3.2 untersucht.

Die genauere Untersuchung der Ursache des großen Konfidenzintervalls bei eine Schreibzugriffslänge von 1.536 Byte (s. Abb. 4.41) macht eine tiefere Analyse erforderlich. Wichtig für die Erklärung des Effektes könnte sein, daß in diesem Bereich der maximale Datagrammdurchsatz gerade erreicht wird. Jedoch zeigt die Vergleichsmessung mit einer Socket-Größe von 64 KByte bei dieser Nutzdatenlänge dieses Verhalten nicht. Eine weitere wichtige Beobachtung findet sich in den System-Logs. Bei Nutzdatenlängen unterhalb von 2.048 Byte wird die Packet Screen offenbar so stark belastet, daß es zu Zustandswechseln der ATM-Schnittstelle zwischen “UP” und “DOWN” kommt. Durch zusätzliche Messungen für weitere Datagrammlängen wird ein geringerer Abstand zwischen den Meßpunkten erreicht und der Bereich kann weiter eingegrenzt werden. Während die Messungen auf direkten Host-zu-Host-Verbindungen auch bei den zusätzlichen Meßpunkten kleine Konfidenzintervalle aufweisen und insgesamt einen sehr gleichmäßigen Kurvenverlauf zeigen (s. Abb. 4.42), wurden bei den Messungen über die Packet Screen zum Teil sehr große Konfidenzintervalle berechnet. Insgesamt ist der Verlauf der Kurve auch nicht so gleichmäßig wie bei den direkten Host-zu-Host-Messungen. Noch größere Abweichungen zeigen sich, wenn in der Packet Screen Filterregeln konfiguriert werden und dadurch die Last auf der Packet Screen steigt. Die Kurve mit zusätzlichen Meßpunkten folgt zwar dem gleichen Verlauf (s. Abb 4.43), jedoch sind zwischen benachbarten Meßpunkten Differenzen von über 30 MBit/s möglich. Da die Abweichungen abhängig von der Datagrammlänge und der Socket-Größe auf den Endsystemen unterschiedlich stark auftreten, muß davon ausgegangen werden, daß durch Füllen des Sendepuffers kurzzeitig das Senden auf der Meßverbindung unterbrochen werden muß. Abhängig von der Socket-Größe und Datagrammlänge treten diese

Zustände mehr oder weniger häufig auf. Während auf der Packet Screen der ATM-Treiber aufgrund hoher Last die ATM-Schnittstelle als "DOWN" meldet, können von der Packet Screen keine weiteren Dateneinheiten abgenommen werden, so daß die Schreibzugriffe auf den Endsystemen sehr schnell die Pufferspeicher füllen. Die beschriebenen Effekte resultieren also aus einem Zusammenspiel aus implementationsspezifischen Problemen bei der Packet Screen (ATM-Treiber, Protokoll-Verarbeitung, Filtersoftware) mit der Pufferung in den Endsystemen.

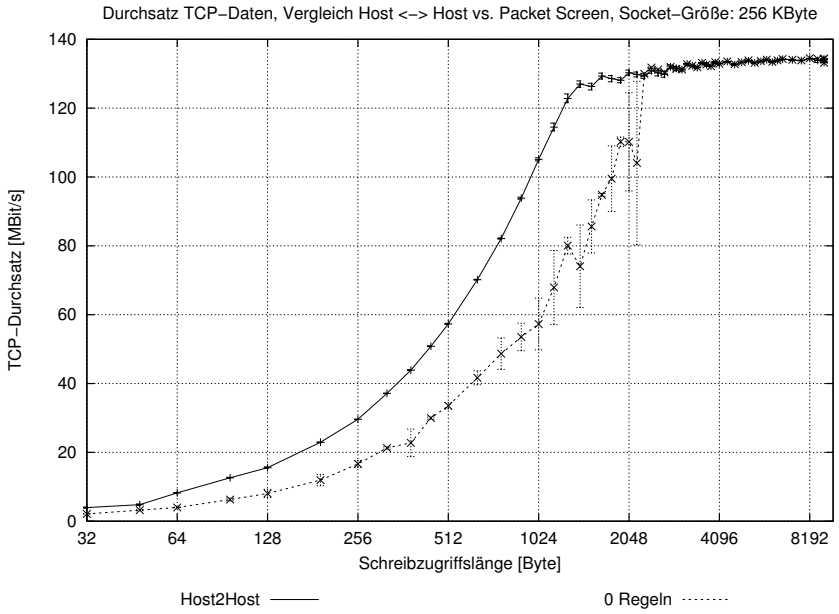


Abbildung 4.42: Durchsatzmessungen TCP über Packet Screen ohne Filterregeln mit zusätzlichen Meßpunkten

Durch das Hinzufügen von Filterregeln steigert sich der Aufwand, der pro Dateneinheit in der Packet Screen entsteht. In den Abbildungen 4.44 und 4.45[99] sind die wichtigsten Ergebnisse der Messungen zusammengestellt. Zum Vergleich enthält jede Grafik auch den Durchsatz einer direkten Host-zu-Host-Verbindung und den Durchsatz, der ohne konfigurierte Filterregeln erreicht wird.

Bereits bei 10 Filterregeln (Abbildung 4.44.1)[100] kann auch mit 2.048 Byte langen Nachrichten nicht mehr der Durchsatz von 130 MBit/s erreicht werden. Diese geringe Anzahl von Filterregeln reicht schon aus, um den Durchsatz um 10 MBit/s zu verringern. Durch das Hinzufügen weiterer Filterregeln sinkt der Durchsatz bei 2.048 Byte Schreibzugriffen weiter ab. Bei 200 Filterregeln (Abb. 4.44.4) ist etwas Merkwürdiges zu beobachten. Bei Schreibzugriffslängen im Bereich von

99 Die Meßkurven sind in voller Größe im Anhang B auf den Seiten 458 bis 462 enthalten.

100 Die Abbildungen werden zeilenweise durchnumeriert.

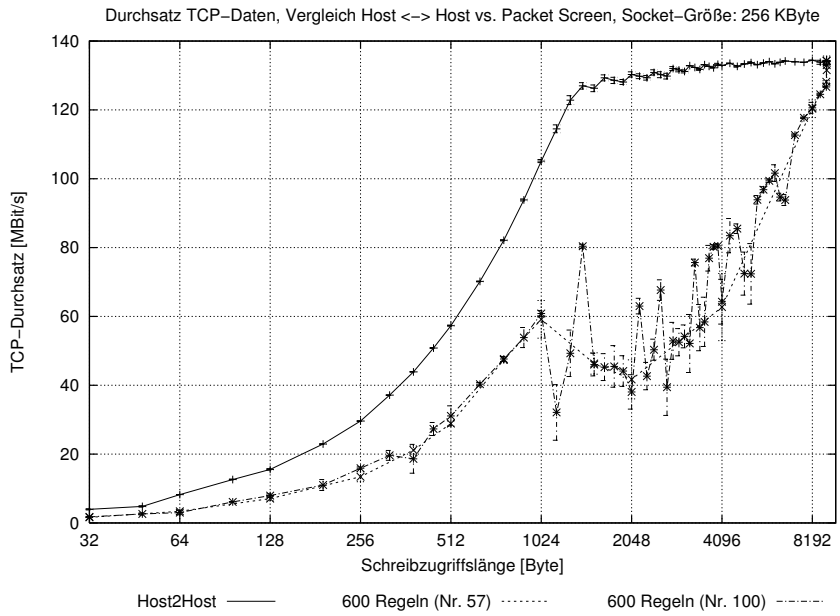


Abbildung 4.43: Durchsatzmessungen TCP über Packet Screen mit 600 Filterregeln mit zusätzlichen Meßpunkten

512 bis 1.024 Byte wird ein höherer Durchsatz erreicht, als bei der Vergleichsmessung mit nur 50 Filterregeln. Hier spielt die Socket-Größe eine wichtige Rolle, denn Vergleichsmessungen mit geringeren Socket-Größen (s. Abb. 4.46.4) zeigen dieses Verhalten nicht. Die Ursache dieses Effektes kann zur Zeit nicht genauer identifiziert werden.

Ab 250 Filterregeln kann auch mit 4.096 Byte Nutzdatenlänge nicht mehr derselbe Durchsatz wie auf einer Host-zu-Host-Verbindung erreicht werden. Bei 600 Filterregeln sind ähnliche Verluste auch mit 8.192 Byte langen Schreibzugriffen zu beobachten. Messungen mit mehr als 1.000 Filterregeln wurden nicht durchgeführt. Selbst bei 1.000 Filterregeln können noch 134 MBit/s erreicht werden; jedoch nur mit 9.140 Byte langen Schreibzugriffen.

Diese Messungen wurden mit einer kleineren Socket-Größe von nur 64 KByte statt 256 KByte wiederholt. Bei der Socket-Größe von 64 KByte (Abb. 4.46 und 4.47)[101] zeigte sich qualitativ dasselbe Verhalten — das Hinzufügen von Filterregeln verringert den Durchsatz im Bereich der Schreibzugriffslängen von 256 bis 2.048 Byte. Erst bei einer großen Anzahl von Filterregeln werden deutliche Abweichungen erkennbar. Die Messungen mit mehr als 500 Filterregeln erreichen den maximalen Durchsatz von über 134 MBit/s nicht mehr. Stattdessen wer-

[101] Die Meßkurven sind in voller Größe im Anhang B auf den Seiten 462 bis 466 enthalten.

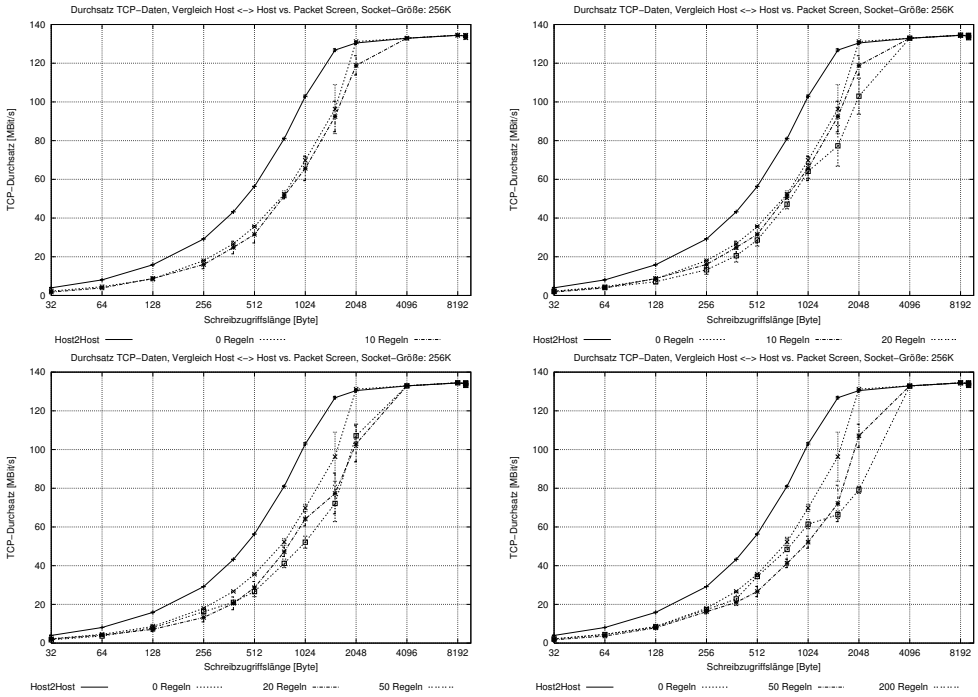


Abbildung 4.44: Durchsatzmessungen TCP über Packet Screen, Socket-Größe 256 KByte

den nur noch 120 MBit/s erreicht. Bei 1.000 Filterregeln sinkt der Wert auf 80 MBit/s.

Beim genaueren Vergleich der Meßergebnisse mit 64 und 256 KByte Socket-Größe fällt auf, daß die bei 256 KByte Socket-Größe beobachteten Effekte, bei denen Messungen mit mehr konfigurierten Filterregeln bei bestimmten Datagrammlängen einen besseren Durchsatz erzielen, weniger häufig vorkommen. Nur bei der Messung mit 1.000 Filterregeln (Abb. 4.46.4) zeigen die Schreibzugriffslängen von 384 und 512 Byte dieses Verhalten. Bei der Socket-Größe von 8 KByte (ohne Abbildung) nimmt dieser Effekt noch weiter ab.

4.3.2 Datagrammdurchsatz über eine Packet Screen

Wie bereits vorhergehend berechnet, können durch die Packet Screen maximal etwa 8.000 Datagramme pro Sekunde verarbeitet werden. Die Abbildung 4.48 zeigt den nach Gleichung 4.16 berechneten Datagrammdurchsatz, der sich aus den

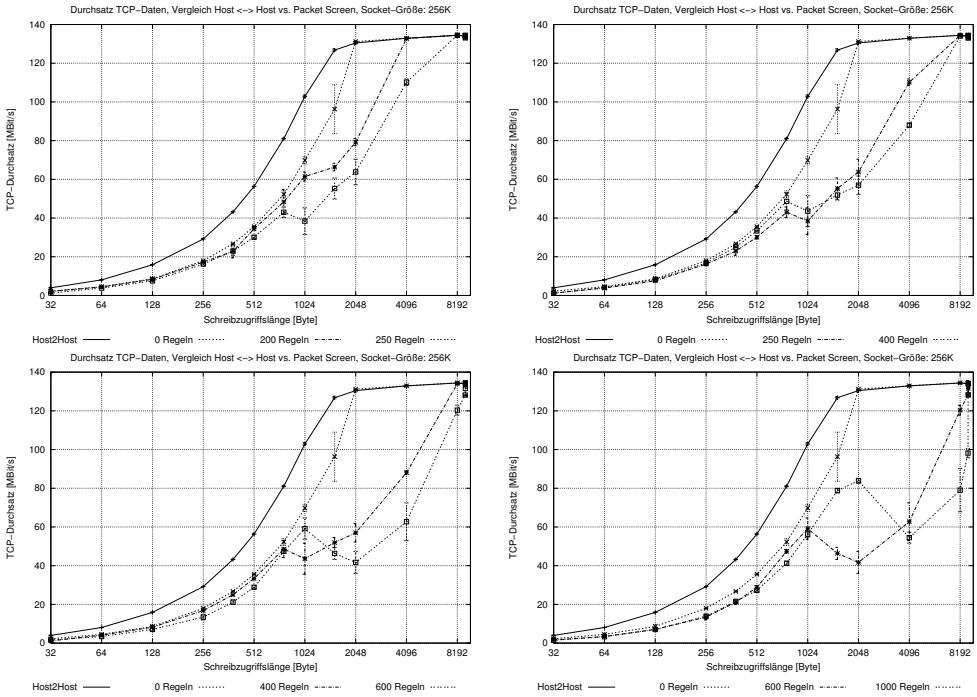


Abbildung 4.45: Durchsatzmessungen TCP über Packet Screen, Socket-Größe 256 KByte (Fortsetzung)

gemessenen Datendurchsätzen ergibt.[102] Zum Vergleich ist ebenfalls die Kurve für eine Host-zu-Host-Messung enthalten. Auch bei dieser Darstellung bestätigt sich, daß der Durchsatz der Packet Screen maximal 8.000 Datagramme/s erreicht. Durch das Hinzufügen weiterer Filterregeln sinkt der Datagrammdurchsatz auf Werte unterhalb von 4.000 Datagramme/s ab.

Die Abbildung 4.48 zeigt, daß unterhalb einer Schreibzugriffslänge von 2.048 Byte nur noch bei der Host-zu-Host-Messung der Datagrammdurchsatz weiter ansteigt. Bei allen Messungen über die Packet Screen sind lediglich Schwankungen zu erkennen. Diese Schwankungen sind zum Teil so groß, daß bei einer größeren Anzahl von Filterregeln an einzelnen Meßpunkten ein höherer Durchsatz erreicht wird. Diese Schwankungen sind darauf zurückzuführen, daß einerseits bei den recht dicht zusammenliegenden Werten schon relativ geringe Meßungenauigkeiten schnell zu Überschneidungen führen, andererseits machen sich auch hier offensichtlich Effekte durch die Socket-Größen bemerkbar. In der Abbildung 4.48

[102] Die Ergänzung weiterer Meßpunkte ändert den Verlauf der Kurven nicht. Die Abbildung 4.48 ist um Meßpunkte an zusätzliche Datagrammlängen ergänzt im Anhang B Abbildung B.25 auf Seite 468 zu finden.

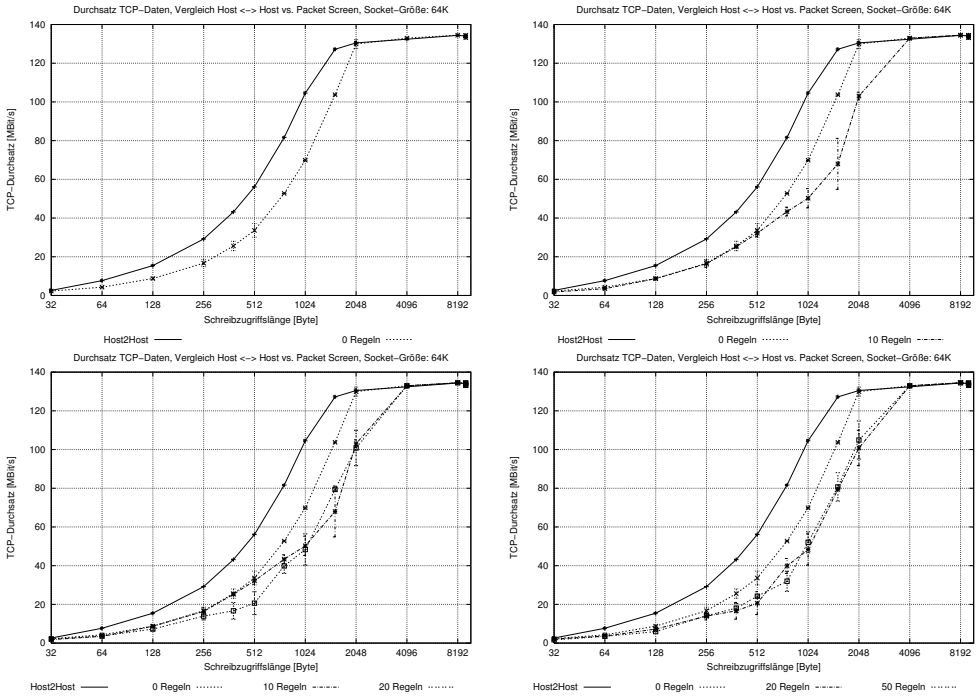


Abbildung 4.46: Durchsatzmessungen TCP über Packet Screen, Socket-Größe 64 KByte

wurde bereits eine Socket-Größe von nur 8 KByte gewählt, weil die Messungen mit größeren Socket-Größen noch stärkere Schwankungen aufwiesen.

Über die Messung des Datendurchsatzes mit anschließender Berechnung nach Gleichung 4.16 läßt sich der Datagrammdurchsatz der Packet Screen nicht besonders genau ermitteln. In Abbildung 4.49 wurde die Host-zu-Host-Messung weggelassen und der Darstellungsbereich erst bei 1.536 Byte langen Schreibzugriffen begonnen. Diese Grafik ermöglicht dadurch zumindest ein ungefähres Abschätzen der Anzahl der Datagramme pro Sekunde, die bei der jeweiligen Filterkonfiguration verarbeitet werden kann.

Anhand von Abbildung 4.49 kann der maximale Datagrammdurchsatz bei der Schreibzugriffslänge von 2.048 Byte abgeschätzt werden. Wie bereits erwähnt steigen die berechneten Werte unterhalb der Schreibzugriffslänge von 2.048 Byte nicht weiter an, so daß die alleinige Betrachtung dieser Schreibzugriffslänge ausreicht. Danach beträgt der Durchsatz ohne konfigurierte Filterregeln etwa 7.500 Datagramme pro Sekunde. Bereits mit nur 10 Filterregeln sinkt der Durchsatz auf einen Wert zwischen 6.500 und 7.000 Datagrammen pro Sekunde. 5.500 Datagramme pro Sekunde sind mit 250 Filterregeln zu erreichen. Dies ist ein um

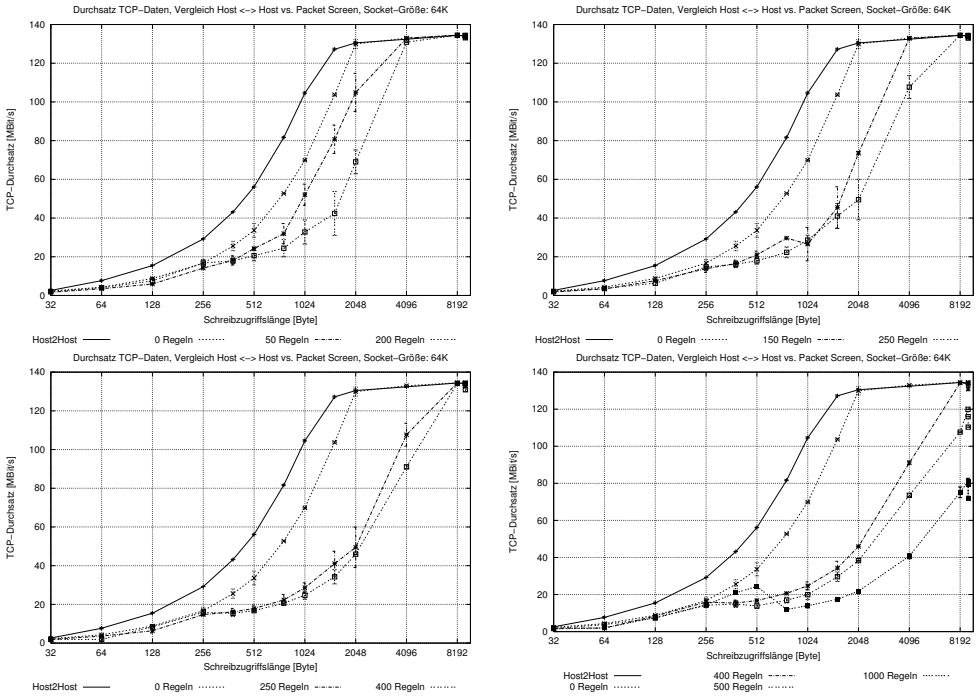


Abbildung 4.47: Durchsatzmessungen TCP über Packet Screen, Socket-Größe 64 KByte (Fortsetzung)

gut ein Viertel (2.000 Datagramme/s) geringerer Durchsatz als ohne konfigurierte Filterregeln. Die für größere Anzahlen von Filterregeln ermittelten Werte sind in der Praxis von geringerer Bedeutung. Sie sind in Tabelle 4.6 der Vollständigkeit halber aufgeführt.

Anzahl der Regeln	Datagramme pro Sekunde
0	7.500
10	6.500-7.000
50	knapp 6.500
250	etwa 5.500
350	4.500-5.000
450	etwa 4.500
600	4.000
1.000	3.000

Tabelle 4.6: Datagrammdurchsatz über Packet Screen

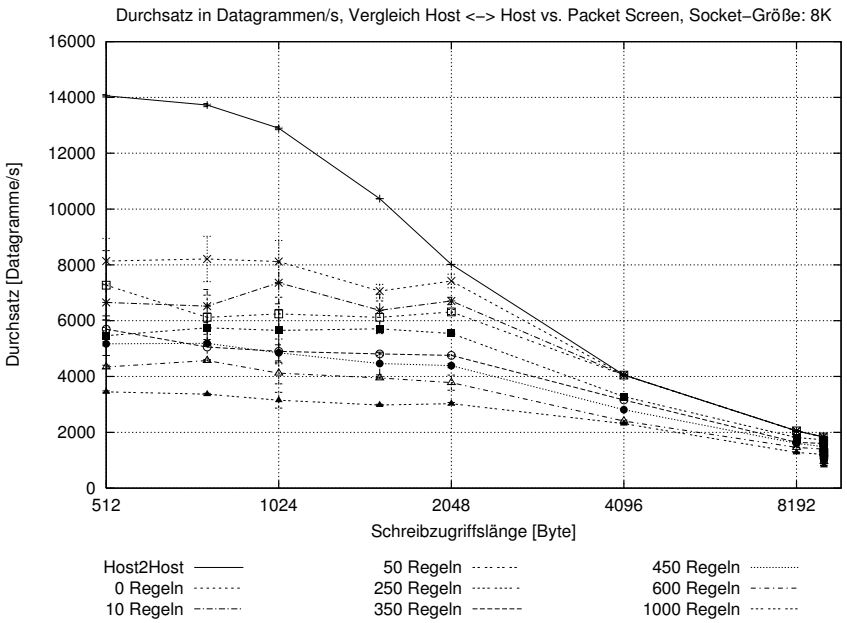


Abbildung 4.48: Datagrammdurchsatz über Packet Screen

Die Abbildung 4.50[103] zeigt eine alternative Darstellung dieser Ergebnisse. Hier wird der Datagrammdurchsatz in Abhängigkeit von der Anzahl der Filterregeln dargestellt; die berechneten Datagrammdurchsätze sind jeweils für eine Schreibzugriffslänge als Kurve dargestellt. In Abbildung 4.50.1 sieht man bei der Schreibzugriffslänge von 2.048 Byte die im vorigen Absatz beschriebenen Auswirkungen der Filterregeln auf die Performanz der Packet Screen. Klar erkennbar ist auch, daß bei 9.140, 8.192 und 4.096 Byte langen Schreibzugriffen so wenig Datagramme pro Sekunde erzeugt werden, daß kaum Auswirkungen durch die Packet Screen erkennbar sind. Bei den kleineren Schreibzugriffslängen von 1.024 und 512 Byte (Abbildung 4.50.2) sind die Meßungenauigkeiten bereits so groß, daß zwar ein ähnlicher Verlauf wie bei der Messung mit 2.048 Byte Schreibzugriffslänge beobachtet werden kann, aber keine genauen Aussagen über den Datagrammdurchsatz getroffen werden können.

Die Berechnung des Datagrammdurchsatzes über die Gleichung 4.16 enthält eine wesentliche Einschränkung. Die Gleichung 4.16 berechnet nur die zur Übertragung der Nutzdaten benötigten Datagramme. Auch die als Empfangsbestätigung gesendeten leeren ACK-Segmente, die bei einer Berechnung anhand des erzielten Datendurchsatzes nicht berücksichtigt werden, verursachen auf der Packet Screen eine zusätzliche Last. Die Anzahl der übertragenen ACKs kann von dem eingesetzten Meßwerkzeug nicht gemessen werden. Da nicht jeder Empfang eines TCP-

[103] Die Meßkurven sind in voller Größe im Anhang B auf den Seiten 466 bis 467 enthalten.

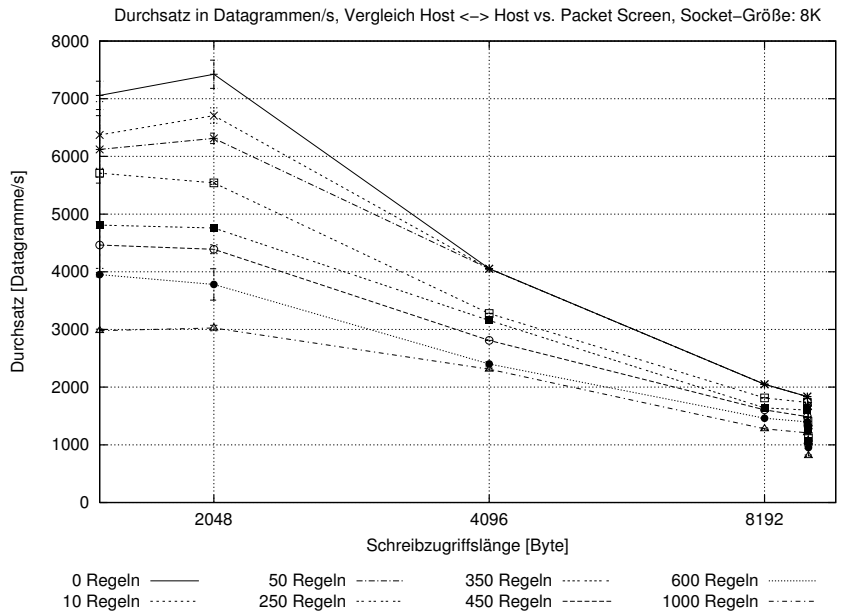


Abbildung 4.49: Datagrammdurchsatz über Packet Screen

Segmentes bestätigt wird, kann die Anzahl der ACK-Segmente nur abgeschätzt werden.

Ein alternativer Ansatz zur Bestimmung des Datagrammdurchsatzes ist über die Paketstatistiken der Packet Screen möglich. Werden vor und nach jeder Einzelmessung die Paketstatistiken der Packet Screen aufgezeichnet, ist dadurch eine sehr exakte Bestimmung des Datagrammdurchsatzes möglich. Die Schwierigkeit dieses Ansatzes liegt in der Synchronisation von Messung und Auslesen sowie in der Auswertung der aufgezeichneten Ergebnisse. In einer Stichprobe werden für unterschiedliche Filterkonfigurationen jeweils zehn Messungen durchgeführt und

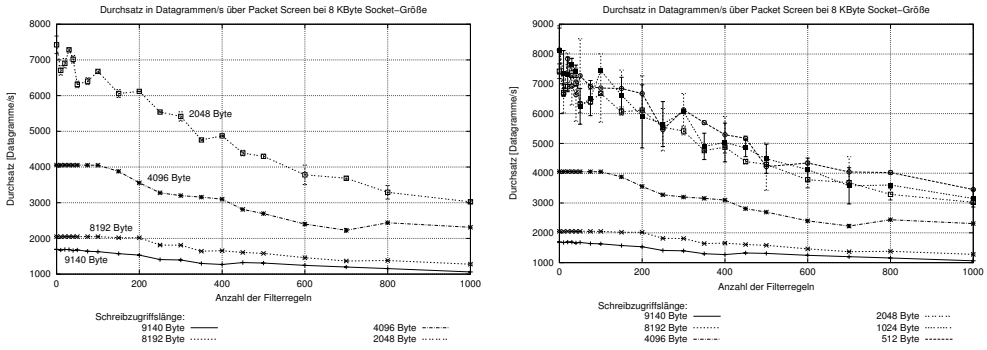


Abbildung 4.50: Datagrammdurchsatz abhängig von der Schreibzugriffslänge

bei jeder Messung die Packet Screen Statistiken ausgelesen. Anschließend wird für jede Einzelmessung der Datagrammdurchsatz nach Gleichung 4.16 mit den aufgezeichneten Statistiken der Packet Screen verglichen. Dabei zeigt sich, daß die von der Packet Screen aufgezeichnete Anzahl der Datagramme bei jedem Meßpunkt um 50% über dem berechneten Datagrammdurchsatz liegt. Dieser Wert ist plausibel, da nach RFC 1122 [Braden 89, S. 96] bei einem Strom von TCP-Segmenten mit MSS-Länge trotz beabsichtigter Verzögerung von Bestätigungen ("Deferred ACKs") spätestens für jedes zweite TCP-Segment eine Bestätigung gesendet werden sollte. In Anbetracht des Aufwands für die Aufzeichnung und Auswertung bei dieser Stichprobenmessung wird darauf verzichtet, den Datagrammdurchsatz durch das Auslesen der Statistiken der Packet Screen in einer zusätzlichen Meßreihe zu ermitteln. Für die Berechnung des Datagrammdurchsatzes durch eine Packet Screen werden im folgenden einheitlich nur die zur Übertragung der Nutzdaten verwendeten Datagramme berücksichtigt. Zusammen mit den in Rückrichtung übertragenen Bestätigungen (ACK) ergibt sich ein um 50% höherer Datagrammdurchsatz der Packet Screen.

Alternativ zu dem vorgestellten Verfahren, bei dem der Datagrammdurchsatz aus dem auf einer TCP-Verbindung gemessenen Datendurchsatz berechnet wird, kann der Datagrammdurchsatz auch aus dem bei UDP erreichten Datendurchsatz (s. Abschnitt 4.2.2.11) berechnet werden. UDP ist ein verbindungsloses Protokoll, das weder über eine Fehlerkorrektur noch über eine Flußkontrolle verfügt. Dadurch ergibt sich gegenüber einer Messung über eine TCP-Verbindung der Vorteil, daß keine unbekannte Anzahl von Empfangsbestätigungen übertragen wird, die im Datendurchsatz nicht enthalten sind. Auch eine Flußkontrolle, durch die TCP einen Sender bei einem überlasteten Netz drosselt, ist bei UDP nicht vorhanden und kann keine Auswirkungen auf den Datagrammdurchsatz haben. Die Berechnung des Datagrammdurchsatzes aus dem Durchsatz (des Empfängers) bei UDP ist daher exakter als eine Berechnung auf der Basis einer TCP-Verbindung, bei der immer eine unbekannte Anzahl von Bestätigungen unberücksichtigt bleibt. Da während einer Messung über UDP von der Packet Screen jedoch Dateneinheiten ausschließlich in einer Richtung übertragen werden, stellt einen UDP-basierte Messung eine sehr ungewöhnliche Anwendung für die Packet Screen dar. Bei einer unidirektionalen UDP-Messung kann es nicht vorkommen, daß bei der Packet Screen gleichzeitig auf beiden Schnittstellen Dateneinheiten eintreffen. Daher ist damit zu rechnen, daß bei einer unidirektionalen UDP-Messung ein höherer Datagrammdurchsatz erzielt werden kann, als es in der Praxis bei bidirektionalem Datenverkehr möglich ist.

4.3.3 Messungen ohne Reduzierung der MTU

Wie bereits in Abschnitt 4.2.2.2 diskutiert, muß die MTU-Größe entsprechend der Schreibzugriffslänge reduziert werden um sicherzustellen, daß keine Aggregation einzelner Schreibzugriffe zu größeren TCP-Segmenten stattfinden kann. Wie wichtig diese Reduktion der MTU-Größe ist, zeigt eine Meßreihe, die zwar mit `TCP_NODELAY`-Option aber ohne reduzierte MTU-Größe durchgeführt wurde.

Der erste Blick auf die Meßergebnisse überrascht, da der Durchsatz für die Meßpunkte im Bereich der Schreibzugriffslängen von 64 bis 1.536 Byte die Vergleichswerte der direkten Host-zu-Host-Messung übertreffen (s. Abb. 4.51). Der Einsatz einer Packet Screen scheint also positive Auswirkungen auf den erreichbaren Datendurchsatz zu haben. Erst die Vergleichsmessung einer direkten Host-zu-Host-Verbindung *ohne* `TCP_NODELAY`-Option überschreitet die Meßergebnisse des Durchsatzes über die Packet Screen.

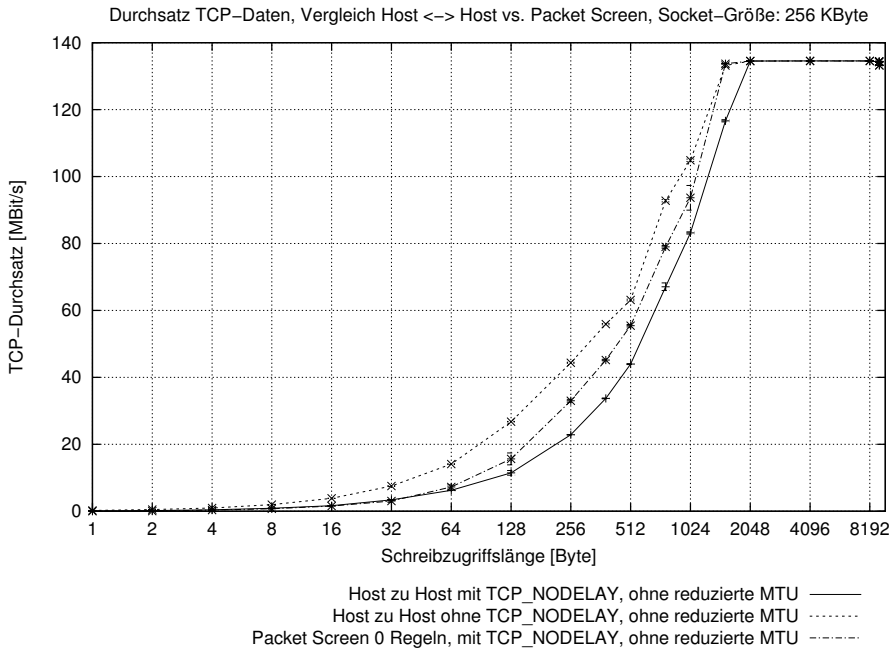


Abbildung 4.51: Durchsatzmessungen TCP über Packet Screen ohne Reduzierung der MTU-Größe

Die Ursache ist auch hier wieder in dem Verhalten der beiden an den Messungen beteiligten Endsysteme begründet, die trotz eingeschalteter `TCP_NODELAY`-Option mehrere Schreibzugriffe zu einem größeren TCP-Segment vereinen (s. Abschnitt 4.2.2.2). Durch die zusätzliche Verzögerung bei der Übertragung durch die Packet Screen kann der Sender nicht alle Schreibzugriffe des Meßwerkzeuges sofort

als Segmente versenden. Da diese Verzögerung die Wartezeit auf die Bestätigungen für gesendete TCP-Segmente verlängert, wird schnell die Anzahl der unbestätigt übertragbaren TCP-Segmente erreicht. Die TCP-Fenstergröße und das "Slow-Start"-Verfahren sind zwei Mechanismen, über die die Anzahl der unbestätigt übertragbaren TCP-Segmente begrenzt wird (s. Abschnitt 4.2.9). Die `TCP_NODELAY`-Option, mit der das beabsichtigte Verzögern nach "Nagles"-Algorithmus [Nagle 84] gesteuert werden kann, greift hier nicht mehr, da äußere Einflüsse die Aggregation mehrerer Schreibzugriffe bedingen. Die Aggregation mehrerer Schreibzugriffe zu einem TCP-Segment kommt in einer Erhöhung des Datendurchsatzes zum Ausdruck.

Bei der Messung ohne reduzierte MTU-Größe sind bei der Socket-Größe von 256 KByte keine Performanzauswirkungen der Packet Screen erkennbar (s. Abb. 4.52.1).[104] Selbst Konfigurationen mit 1.000 Filterregeln erreichen noch einen maximalen Durchsatz von 134 MBit/s. Lediglich bei den Schreibzugriffslängen von 128 Byte und 1.024 Byte sind Abweichungen zu erkennen, die auf unterschiedlich hohe Aggregation der Schreibzugriffe zu größeren TCP-Segmenten zurückzuführen sind. Bei der Schreibzugriffslänge von 128 Byte erreichen die Messungen mit einer größeren Anzahl an Filterregeln nicht ganz den Durchsatz der Vergleichsmessung ohne konfigurierte Filterregeln. Im Bereich von 1.024 Byte langen Schreibzugriffen übertreffen die Messungen mit 100 und 1.000 Filterregeln hingegen den Durchsatz der Messungen ohne Filterregeln und mit 500 Filterregeln — eine klare Abhängigkeit von der Anzahl der Filterregeln ist also nicht erkennbar.

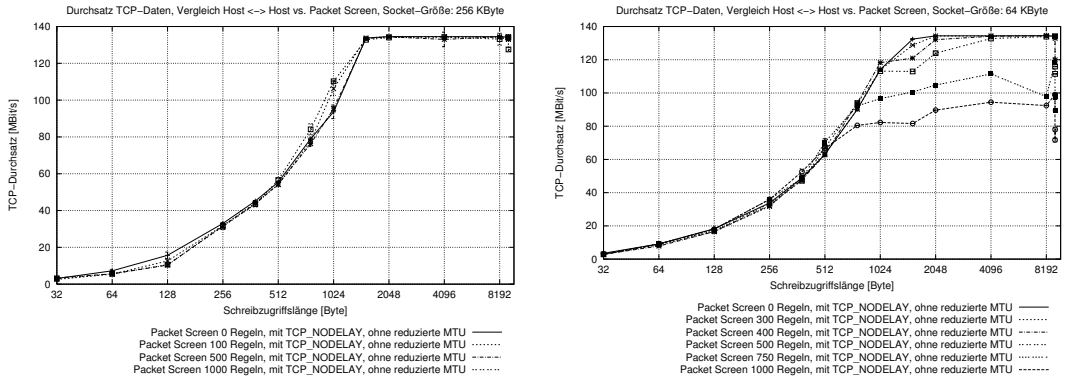


Abbildung 4.52: Durchsatzmessungen TCP über Packet Screen ohne Reduzierung der MTU-Größe

Bei kleineren Socket-Größen zeigen sich aber auch ohne Reduzierung der MTU-Größe Auswirkungen der Packet Screen auf den Datendurchsatz. Bei einer Socket-Größe von 64 KByte sind erste, geringe Auswirkungen auf die Perfor-

[104] Die Meßkurven sind in voller Größe im Anhang B auf den Seiten 467 bis 468 enthalten.

manz ab 300 Filterregeln bei einer Schreibzugriffslänge von 1.536 Byte erkennbar (s. Abb. 4.52.2). Bei 400 Filterregeln wird für 1.536 Byte lange Schreibzugriffe nur noch ein Durchsatz von 120 MBit/s erreicht. Die Messung mit 500 Filterregeln zeigt auch einen Rückgang bei 2.048 Byte Schreibzugriffen, für 8.192 Byte lange Schreibzugriffe wird aber immer noch ein maximaler Durchsatz von knapp 134 MBit/s erreicht. Bei 1.000 Filterregeln kann mit Schreibzugriffslängen von 768 bis 1.536 Byte nur noch ein Durchsatz von knapp über 80 MBit/s erreicht werden. Das Maximum dieser Kurve liegt bei 100 MBit/s.

Bei Messungen mit noch geringeren Socket-Größen setzten die Auswirkungen auf die Performanz bereits bei einer kleineren Anzahl von Filterregeln ein. Bei einer Socket-Größe von 32 KByte ist ein Durchsatzrückgang bereits ab 100 Filterregeln erkennbar (ohne Abbildung). Wird die Socket-Größe weiter verringert auf 8 KByte, der voreingestellten (Default) Socket-Größe bei Solaris 2.6, so kann selbst ohne konfigurierte Filterregeln der Durchsatz einer direkten Host-zu-Host-Verbindung nicht mehr erreicht werden, da durch den kleineren Socket-Puffer die Aggregation erschwert wird.

Die Messungen ohne reduzierte MTU-Größe zeigen, daß geschickte Protokoll-Implementationen auf den Endsystemen die durch Firewalls bedingten Performanzeinbußen zum Teil ausgleichen können. Durch das Bereitstellen ausreichend großer (Socket) Pufferspeicher ist es in der beschriebenen Testumgebung sogar möglich, den Durchsatz einer direkten Host-zu-Host-Verbindung zu übertreffen. Ein weiteres Ergebnis dieser Messungen ist, daß gemessene Werte einer genauen Interpretation bedürfen, um die Abhängigkeiten der verschiedenen Mechanismen untereinander korrekt bewerten zu können.

4.3.4 Nachrichtenlaufzeit über eine Packet Screen

Die Auswirkung der Packet Screen auf die Nachrichtenlaufzeit wird mit einer zusätzlichen Messung ermittelt. Für unterschiedliche Anzahlen von Filterregeln wird jeweils die Anzahl der möglichen Umläufe auf einer TCP-Verbindung gemessen.

Der Einsatz einer Packet Screen zwischen Sender und Empfänger hat sehr starke Auswirkungen auf die Nachrichtenlaufzeit. Bereits ohne konfigurierte Filterregeln können nur noch 1.540 ± 4 Umläufe pro Sekunde gemessen werden. Auf einer

direkten Host-zu-Host-Verbindung über den Switch können im Vergleich dazu 2.593 ±2 Umläufe pro Sekunde erreicht werden (s. Abschnitt 4.2.3).[105]

Die Abbildung 4.53 zeigt die über die Packet Screen gemessenen Umläufe pro Sekunde für unterschiedliche Filterkonfigurationen. Bis einschließlich 40 Filterregeln können noch über 1.400 Umläufe pro Sekunde erreicht werden. Bei 100 Filterregeln sind nur noch 1.277 Umläufe in einer Sekunde möglich. Jenseits von 250 Filterregeln werden 1.000 Umläufe pro Sekunde unterschritten. Bei 1.000 Filterregeln sind schließlich nur noch 495 Umläufe pro Sekunde möglich.

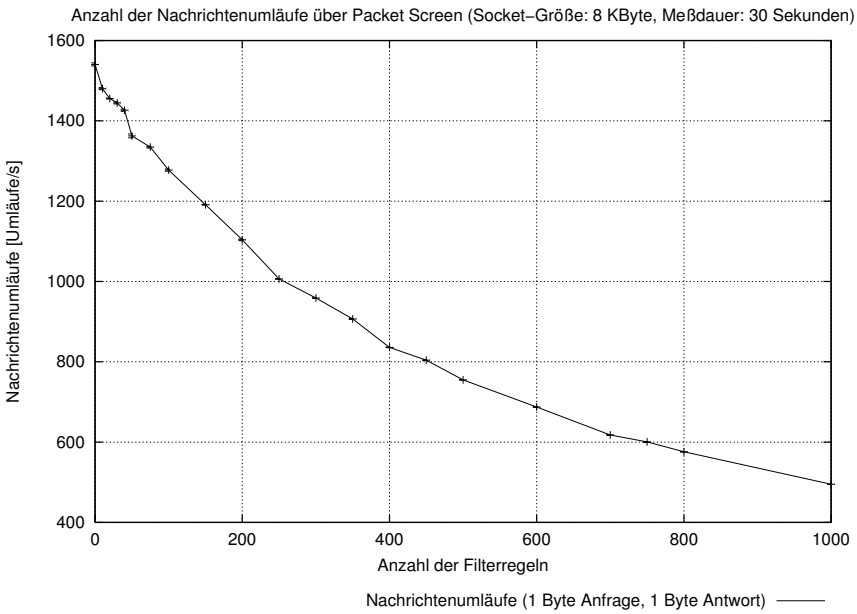


Abbildung 4.53: Nachrichtenumläufe über Packet Screen

Nach der Gleichung 4.17 (S. 249) können die einzelnen Nachrichtenlaufzeiten für die jeweilige Filterkonfiguration berechnet werden. Die Abbildung 4.54 zeigt, daß die Nachrichtenlaufzeit dabei linear mit der Anzahl der Filterregeln zunimmt.

Durch Subtraktion der Nachrichtenlaufzeit durch die Packet Screen ohne konfigurierte Filterregeln erhält man für die folgenden Meßpunkte jeweils die Zeitdauer,

[105] Bedenkt man die doppelte Übertragung der Nachrichten (vom Sender zur Packet Screen und von der Packet Screen zum Empfänger), wäre eine Halbierung der Nachrichtenumläufe im Vergleich zur direkten Host-zu-Host-Verbindung zu erwarten gewesen. Der gemessene Wert von 1.540 Umläufen pro Sekunde überschreitet die bei einer Halbierung zu erwartenden 1.296 Umläufe, da die Ergebnisse der Umlaufmessung im LAN durch die Verarbeitungszeiten in den Endsystemen bestimmt wird. Da bei der Packet Screen kein TCP "Overhead" anfällt, kann mit 1.540 Umläufen pro Sekunde ein vergleichsweise hoher Wert erreicht werden.

Konfiguration	Umläufe [Umläufe/s]	Nachrichtenlaufzeit [s]
IP-Filter 0 Regeln	1.540 ±4	$3,25 * 10^{-4}$
IP-Filter 10 Regeln	1.480 ±3	$3,38 * 10^{-4}$
IP-Filter 50 Regeln	1.363 ±5	$3,67 * 10^{-4}$
IP-Filter 100 Regeln	1.277 ±3	$3,92 * 10^{-4}$
IP-Filter 250 Regeln	1.006 ±1	$4,97 * 10^{-4}$
IP-Filter 500 Regeln	755 ±1	$6,62 * 10^{-4}$
IP-Filter 1.000 Regeln	495 ±1	$10,10 * 10^{-4}$
Host-zu-Host über Switch	2.593 ±2	$1,93 * 10^{-4}$

Tabelle 4.7: Zusammenfassung der Umlaufmessungen über eine Packet Screen

die für die Auswertung der Filterregeln in der Packet Screen benötigt werden. Dividiert man diese Zeitdauer durch die jeweilige Anzahl der Filterregeln, ergibt sich die Zeitdauer, die für die Auswertung einer Filterregel benötigt wird.[106] Im Mittel werden pro Filterregel $6,69 * 10^{-7}$ Sekunden (Standardabweichung $1,81 * 10^{-8}$) benötigt.

4.3.5 Verbindungsaufbauzeit über eine Packet Screen

Die Verbindungsaufbauzeiten hängen von der Verarbeitungszeit in den Endsystemen und von den Nachrichtenlaufzeiten ab. Da, wie im vorigen Abschnitt gezeigt, eine lineare Abhängigkeit zwischen der Anzahl der Filterregeln und der Nachrichtenlaufzeit besteht, ist damit zu rechnen, daß die Verbindungsaufbauzeiten im gleichen Maße wie die Nachrichtenlaufzeiten steigen.

Zur Ermittlung der Verbindungsaufbauzeiten wird die in Abschnitt 4.2.4 vorgestellte Transaktionsmessung über eine Packet Screen (s. Abb. 4.40) durchgeführt. Für die Messung werden die Socket-Größen von 8 und 256 KByte bei einer Meßdauer von zwei Minuten ohne Setzen der `NODELAY`-Option gewählt. Die Filterregeln der Packet Screen müssen für die Messungen geringfügig modifiziert werden. Im Gegensatz zur Messung des Datendurchsatzes, für die *eine* TCP-Verbindung eröffnet wird, deren Portnummern sich in festgelegten Grenzen[107] bewegen, werden bei einer Transaktionsmessung Verbindungen schnell nacheinander auf- und

[106] Da die Abbildung 4.54 im unteren Bereich bis 100 Filterregeln etwas größere Abweichungen zeigte, wurde diese Berechnung erst ab 100 Filterregeln durchgeführt.

[107] Die im DFN-FWL weiterentwickelte Version von "Netperf" verwendet bei der Messung des TCP-Datendurchsatzes ausschließlich Portnummern oberhalb von 32.768.

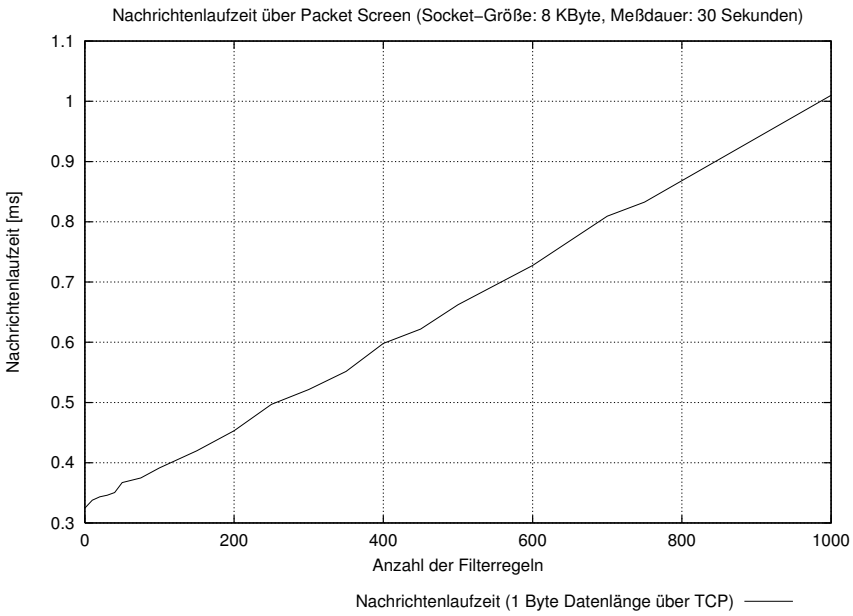


Abbildung 4.54: Nachrichtenlaufzeit über Packet Screen

wieder abgebaut. Dabei wird der gesamte Bereich der unprivilegierten Ports[108] verwendet. Dieser geänderte Portbereich muß sich in der jeweiligen Filterkonfiguration widerspiegeln, deren letzte Filterregel die Übertragung des zur Messung gehörenden Verkehrs erlauben soll. Von dieser geringen Konfigurationsänderung sind keine Auswirkungen auf die Performanz der Packet Screen zu erwarten.

Die Abbildung 4.55 zeigt den Verlauf der Transaktionen pro Sekunde im Vergleich zu den Nachrichtenumläufen pro Sekunde. Beiden Kurven liegen Messungen mit einer Socket-Größe von 8 KByte zugrunde. Die nicht abgebildete Transaktionsmessung mit 256 KByte Socket-Größe liefert Ergebnisse, die geringfügig (1%) unterhalb der Meßergebnisse mit 8 KByte Socket-Größe liegen. Durch die unterschiedliche Skalierung der Y-Achsen für die Nachrichtenumläufe (Skalierung auf der linken Seite) und die Transaktionsmessung (Skalierung auf der rechten Seite) wird eine Überdeckung der beiden Kurven erreicht. Daran erkennt man, daß sich die beiden Meßkurven ab 200 Filterregeln immer um einen konstanten Faktor voneinander unterscheiden. Unterhalb von 200 Filterregeln ist die Verzögerung durch die Packet Screen noch so gering, daß die beim Verbindungsaufbau notwendige Protokollverarbeitung in den beiden Endsystemen überwiegt. Erst oberhalb von 200 Filterregeln ist die Verzögerung durch die Packet Screen so groß, daß sie als wesentlicher Faktor die Anzahl der Transaktionen pro Sekunde begrenzt.

[108] Die Portnummern bis 1023 sind bei UNIX-Systemen als sogenannte “privilegierte Ports” nur von privilegierten Benutzern (`root`) zugreifbar.

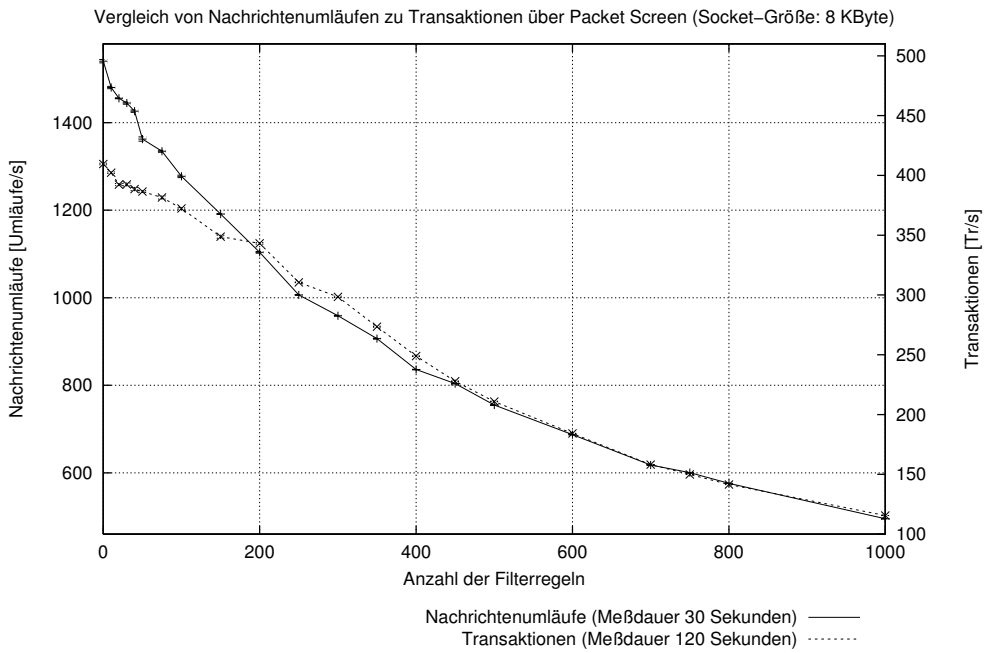


Abbildung 4.55: Nachrichtenumläufe vs. Transaktionen über Packet Screen

Aus dem in den Messungen ermittelten Transaktionsdurchsatz für die jeweilige Anzahl von Filterregeln kann die Dauer einer Transaktion berechnet werden. In Abschnitt 4.2.4 wurde für direkte Host-zu-Host-Verbindungen eine Transaktionslänge von $1,92 * 10^{-3}$ Sekunden ermittelt. Bei der Messung über die Packet Screen ergibt sich ohne konfigurierte Filterregeln eine durchschnittliche Transaktionslänge von $2,44 * 10^{-3}$ Sekunden. Bei 10 Filterregeln sind dies $2,49 * 10^{-3}$ Sekunden. Mit 200 Filterregeln wird eine Transaktionslänge von fast 3 ms erreicht ($2,91 * 10^{-3}$ Sekunden). Die längste gemessene Filterkonfiguration mit 1.000 Regeln führt zu einer Transaktionslänge von $8,65 * 10^{-3}$ Sekunden, was ungefähr der 4,5-fachen Transaktionslänge im Vergleich zu einer direkten Host-zu-Host-Messung entspricht.

4.3.6 Abschließende Bewertung der Performanz von Packet Screens

Die Messungen mit verschiedenen Packet Screen Konfigurationen haben gezeigt, daß in 155 MBit/s ATM-Netzen die Kopierleistung selbst einfacher Workstations ausreicht, um den theoretisch möglichen Datendurchsatz bei TCP-Verbindungen über CLIP von 134,80 MBit/s knapp zu erreichen. Sogar bei sehr langen Filterkonfigurationen von 1.000 Regeln konnten mit maximal 134 MBit/s an einzelnen

Meßpunkten immer noch Durchsätze erreicht werden, die sich praktisch nicht von den Vergleichsmessungen auf direkten Host-zu-Host-Verbindungen unterschieden.

Die zunächst erwarteten Leistungseinbußen durch die Segmentierung und Reassemblierung von IP-Datagrammen zu Zellen und zurück in IP-Datagramme hatten keinen Einfluß auf die Performanz der Packet Screen, da diese Aufgaben effizient in der Hardware der ATM-Schnittstellen erledigt werden, und somit nicht die CPU belasten.

Die Performanzeinbußen beim Einsatz von Packet Screens halten sich auch deshalb in Grenzen, weil die TCP-Implementationen der Endsysteme in der Lage sind, die durch die Packet Screen verursachten Leistungseinbußen teilweise zu kompensieren. Durch die Aggregation mehrerer Schreibzugriffe zu einem entsprechend größeren TCP-Segment in den Endsystemen brauchen von der Packet Screen weniger Dateneinheiten verarbeitet zu werden (s. Abschnitt 4.3.3). Die Aggregation ist dabei um so höher, je größer die Verzögerungen durch die Packet Screen sind. Eine höhere Aggregation reduziert wiederum die Paketlast auf der Packet Screen.

Die wichtigste Kenngröße bei Packet Screens ist der Datagrammdurchsatz (Datagramme pro Sekunde). Während die Einflüsse von Packet Screens auf den Datendurchsatz nur gering sind, haben die Messungen gezeigt, daß der begrenzende Faktor bei Packet Screens der Durchsatz an Datagrammen pro Sekunde ist. Der maximale gemessene Datagrammdurchsatz der Packet Screen ist mit knapp 8.000 Datagrammen pro Sekunde auch nur etwa halb so hoch, wie bei einer direkten Host-zu-Host-Verbindung. Durch das Hinzufügen von Filterregeln sinkt der Datagrammdurchsatz stark. Dieses Ergebnis gewinnt in der Praxis weitere Relevanz, da in realen Netzen kurze Datagrammlängen überwiegen, was bei gleichem (festgelegtem) Datendurchsatz einen höheren Datagrammdurchsatz des Netzes ermöglicht. Der Datagrammdurchsatz der Packet Screen ermöglicht jedoch nur bei Datagrammlängen von mindestens zwei KByte das Erreichen des theoretisch möglichen Datendurchsatzes des Netzes.

Zusätzlich zu dem Performanzproblem bei kleineren Datagrammen werden durch die Packet Screen die Laufzeiten der IP-Dateneinheiten verlängert, da die Überprüfung der Filterregeln und die zusätzliche Protokollverarbeitung in der Packet Screen die Übertragung an den Empfänger verzögert. Neben der konstanten Verzögerung durch die Packet Screen, wie sie bei einer Konfiguration ohne konfigurierte Filterregeln gemessen werden kann, besteht eine lineare Abhängigkeit von der Anzahl der Filterregeln (s. Abb. 4.54).

Obwohl eine Packet Screen nicht direkt am TCP-Verbindungsaufbau beteiligt ist, steigt mit der Zunahme der Nachrichtenlaufzeit auch die Verbindungsaufbauzeit. Bei Konfigurationen bis 200 Filterregeln sind jedoch immer noch primär

die Endsysteme für Engpässe verantwortlich. Erst oberhalb von 200 Filterregeln überwiegen die Einbußen durch die Packet Screen.

Trotz der deutlichen Reduzierung des Datagrammdurchsatzes und der linearen Abhängigkeit der Nachrichtenlaufzeit von der Anzahl der Filterregeln reicht die Performanz der Packet Screen aus, um innerhalb des ATM-Test-Netzes eine sichere Trennung durchzusetzen, ohne dabei einen Engpaß darzustellen. Dieses Ergebnis unterliegt jedoch zwei wichtigen Einschränkungen, die im folgenden diskutiert werden sollen. Zum einen können mit den Meßwerkzeugen nur unidirektionale Datendurchsatzmessungen durchgeführt werden. Auf der anderen Seite wird der maximale Durchsatz nur für große Datagrammlängen erreicht.

Die gemessenen Datendurchsätze durch eine Packet Screen beruhen auf Messungen über TCP-Verbindungen, bei denen Daten ausschließlich in einer Richtung übertragen werden. Da es bei ATM aber möglich ist, auf derselben Schnittstelle gleichzeitig zu senden und zu empfangen (voll-duplex), muß eine Packet Screen maximal 134 MBit/s in beiden Richtungen übertragen können. Das Verhalten einer Packet Screen unter dieser doppelten Last kann mit den vorhandenen Meßwerkzeugen nicht ausreichend untersucht werden,[109] so daß nur Abschätzungen durchgeführt werden können.[110] Das maximal zu erwartende Datenaufkommen wäre mit knapp 270 MBit/s ($2 * 134\ MBit/s = 268\ MBit/s$) noch unterhalb der bei den "Localhost"-Messungen ermittelten Durchsätze von 300-400 MBit/s (s. Abschnitt 4.2.2.10). Der ermittelte Datagrammdurchsatz muß beim Voll-Duplex-Betrieb der Packet Screen für die Datenübertragung in beide Richtungen ausreichen. Es verbleiben somit für jede Richtung etwa 4.000 Datagramme pro Sekunde, um einen Datendurchsatz von 134 MBit/s zu ermöglichen. Nach Abbildung 4.50 wäre eine Datagrammlänge von 4 KByte notwendig, um mit 4.000 Datagrammen pro Sekunde einen Datendurchsatz von 134 MBit/s zu erreichen. Für die unidirektionale Messung war hingegen nur ein Mindestdatagrammlänge von zwei KByte erforderlich. Aus diesen Ergebnissen läßt sich abschätzen, daß

[109] Zwar werden bei der TCP-Messung über die Bestätigung (ACK) auch Dateneinheiten in der Rückrichtung gesendet, jedoch transportieren diese Dateneinheiten keine Nutzdaten, so daß der Datendurchsatz nur in einer Richtung ausgenutzt wird.

[110] Zur Messung des Datendurchsatzes durch eine Packet Screen mit gleichzeitigen Datenströmen in beiden Richtungen ist es notwendig, synchronisiert mit dem Senden zu beginnen. Diese Synchronisation ist das Hauptproblem bei dieser Messung. Ohne eine Synchronisation wäre es möglich, daß Teile der Messung ohne die zusätzliche Belastung des zweiten Datenstroms in der Gegenrichtung ablaufen würden und dadurch bessere Durchsätze erzielen könnten. Ein Meßwerkzeug, das in beiden Richtungen Datenströme zur Durchsatzmessung erzeugen kann, hat bezüglich der Synchronisation ähnliche Anforderungen wie das bereits vorgestellte HTTP-Meßwerkzeug. Die Realisierung wird jedoch dadurch erschwert, daß die zu synchronisierenden Prozesse auf unterschiedlichen Endsystemen ausgeführt werden. Um zudem sicherzustellen, daß die an der Messung beteiligten Endsysteme keine Engpaß bilden, sollten für beide Datenströme unterschiedliche Rechnerpaare verwendet werden.

die Performanz einer Packet Screen auch für den Voll-Duplex-Betrieb knapp ausreichen kann, um den maximalen Datendurchsatz bei langen Datagrammen zu ermöglichen.

Der maximale Datendurchsatz wird über eine Packet Screen nur bei großen Datagrammlängen erreicht. Bei den Messungen ohne Reduzierung der MTU-Größe (s. Abschnitt 4.3.3) sorgt die Aggregation in den Endsystemen dafür, daß große Datagrammlängen überwiegen, so daß innerhalb des Testnetzes große Datenlängen genutzt werden können, um die Performanzeinbußen des Einsatzes der Packet Screen zu begrenzen. Die dadurch möglichen Datagrammlängen von über 2 KByte können jedoch außerhalb des Testnetzes meist nicht erreicht werden. Wird zum Beispiel ein ATM-Netz als Backbone für eine große Anzahl von kleineren (Fast)Ethernets verwendet, über die die Endsysteme angeschlossen sind, so ist die Datagrammlänge durch die MTU-Größe von Ethernet auf 1.500 Byte beschränkt. Bei Messungen der Paketlängen in Ethernets (s. Anhang C.1) werden durchschnittliche Datagrammlängen von weniger als 500 Byte ermittelt. Messungen der Paketlängenverteilung bei amerikanischen Internet-Providern[111] und im B-WiN [Treml 98] ergeben ebenfalls durchschnittliche Paketlängen, die zum Teil sogar noch deutlich unterhalb von 500 Byte liegen. Auch in ATM-Netzen ist die durchschnittliche Datagrammlänge nicht größer, wenn wie in dem beschriebenen ATM-Backbone-Netz die Datagrammlänge durch die MTU-Größe des Netzes der lokalen Endsysteme begrenzt ist oder aber geringere MTU-Größen auf dem Pfad zum Empfänger kürzere Datagrammlängen erforderlich machen.[112] Selbst in reinen ATM-Netzen können viele Protokolle die mögliche Datagrammlänge von über 9 KByte nicht erreichen, da nur sehr geringe Datenmengen ausgetauscht werden müssen.[113] Es ist daher realistisch anzunehmen, daß die durchschnittlichen Datagrammlängen auch in reinen ATM-Netzen kaum oberhalb von 500 Byte liegen werden.[114] In diesem Bereich kann durch die Packet Screen nur ein Durchsatz von maximal 30-40 MBit/s erreicht werden (s. Abb. 4.56). Auf einer direkten Host-

111 Siehe `http://www.nlanr.net/NA/learn/packetsizes.html`

112 Die Anbindung der Endsysteme über Ethernet reduziert nicht den erreichbaren Gesamtdurchsatz auf der Packet Screen, da dieser Gesamtdurchsatz aus der Summe vieler Einzeldurchsätze von Kommunikationsbeziehungen zwischen vielen über Ethernet angebundener Systeme bestehen kann.

113 Zum Ausliefern einer 2 KByte langen E-Mail müssen nach dem TCP-Verbindungsaufbau zeilenweise Adreßinformationen ausgetauscht werden, bevor die E-Mail übertragen werden kann. Lediglich bei der Übertragung der 2 KByte langen E-Mail kann in einem ATM-Netz ein Datagramm mit einer großen Datagrammlänge versendet werden. Alle vorher ausgetauschten Informationen sind durch den Aufbau von SMTP [Postel 82] auf wenige Byte begrenzt.

114 Unter der Annahme, daß weiterhin dieselben Anwendungsprotokolle mit denselben Verkehrscharakteristiken verwendet werden, können nur die Dateneinheiten von einer größeren MTU profitieren, die heute an der MTU liegen. Diese Dateneinheiten machen jedoch nur einen geringen Anteil an der Gesamtanzahl der Pakete aus.

zu-Host-Verbindung liegt der maximale Datendurchsatz bei 500 Byte langen Datagrammen unterhalb von 60 MBit/s, während theoretisch sogar ein Durchsatz von 120 MBit/s möglich wäre. Die Ursache dieses geringen Durchsatzes bei dieser Datagrammlänge ist der begrenzte Datagrammdurchsatz der Packet Screen. Um den theoretisch möglichen Durchsatz bei der Datagrammlänge von 500 Byte zu erreichen, muß der Durchsatz der Packet Screen auf etwa 30.000 Datagramme pro Sekunde vervierfacht werden. Bei der Absicherung eines Netzes mit durchschnittlichen Datagrammlängen unterhalb von 2 KByte über eine Workstation-basierte Packet Screen sind daher bereits bei 155 MBit/s ATM-Netzen sehr deutliche Performanzeinbußen zu erwarten.

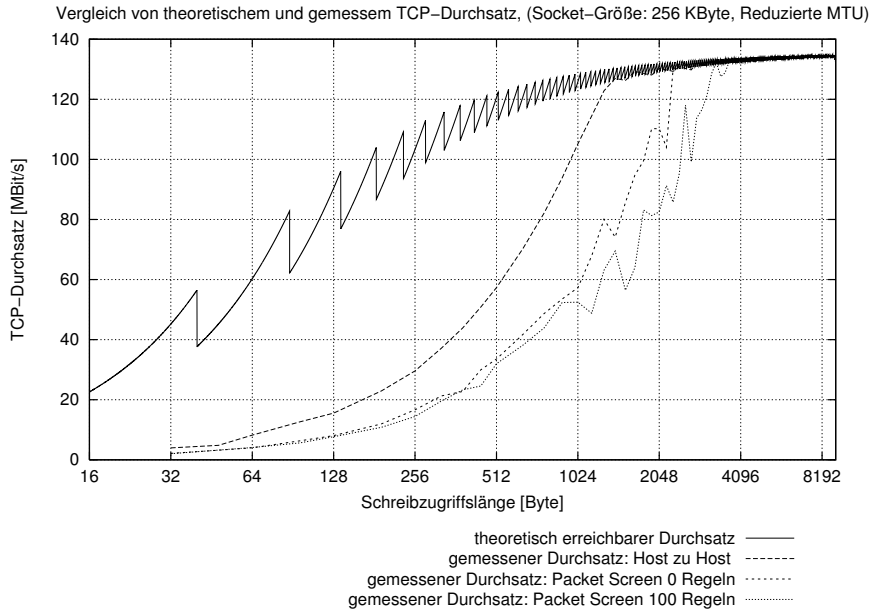


Abbildung 4.56: Vergleich des theoretischen Datendurchsatzes mit gemessenen Durchsätzen über eine Packet Screen

Bei höheren Skalierungsstufen von ATM, etwa OC-12c mit 622 MBit/s, verschärft sich das Durchsatzproblem bei Packet Screen zunehmend. Sind bei 155 MBit/s ATM "nur" etwa 200.000 Datagramme pro Sekunde notwendig (s. Abschnitt 4.2.2.6), um bei kurzen Datagrammen (maximal 40 Byte Datagrammlänge) den theoretisch möglichen Datendurchsatz zu erreichen, werden bei 622 MBit/s ATM bereits 800.000 Datagramme pro Sekunde benötigt. Die bei den Messungen ermittelten 8.000 Datagramme pro Sekunde, die eine Packet Screen maximal — d. h. ohne konfigurierte Filterregeln — übertragen kann, reichen bei 155 MBit/s ATM noch für Datagrammlängen ab 2 KByte zum Erreichen des theoretisch möglichen Datendurchsatzes. Bei 622 MBit/s ATM wird der theoretisch mögliche Datendurchsatz erst bei Datagrammlänge von 8 KByte knapp unter der MTU-Größe erreicht. In 622 MBit/s ATM-Netzen wird somit *jede* Reduzierung der

Performanz der Packet Screen — etwa durch die Konfiguration von Filterregeln — selbst bei großen Datagrammlängen Durchsatzeinbußen zur Folge haben. Eine Kompensation durch Aggregation in den Endsystemen ist wegen Erreichen der MTU-Größe nicht mehr möglich.

Fazit der Packet Screen Messungen ist, daß der *Daten*durchsatz von Packet Screens in 155 MBit/s ATM-Netzen noch kein Problem ist. In den üblichen Einsatzbereichen von Packet Screens führen jedoch durchschnittliche Datagrammlängen im Bereich von 500 Byte dazu, daß die Packet Screen durch den geringen Datagrammdurchsatz zum Engpaß wird. Daher ist bereits für den Einsatz von Packet Screens in 155 MBit/s ATM-Netzen eine Vervierfachung des Datagrammdurchsatzes auf etwa 30.000 Datagrammen pro Sekunde erforderlich. Für höhere Skalierungsstufen von ATM sind noch höhere Durchsätze erforderlich.

4.4 Performanz von Proxy-Servern

Proxy-Server finden in Firewall-Architekturen sowohl bei Gateway-Firewalls (s. Abschnitt 2.3.2.3) als auch bei Kombinationen von Packet Screen und Bastion (s. Abschnitt 2.3.2.2) Anwendung. Aus diesem Grund muß die Performanz von Proxy-Servern in zwei unterschiedlichen Testnetz-Konfigurationen gemessen werden. Die Konfiguration für Gateway-Firewall-Messungen (Abb. 4.57) entspricht von der Struktur dem Setup, das für die Messungen mit Packet Screens (s. Abschnitt 4.3) verwendet wird. Das Gateway "`mistral`" ist mit zwei Netzwerkschnittstellen ausgestattet, die mit zwei unterschiedlichen IP-Subnetzen (`192.168.14.0` und `192.168.15.0`) verbunden sind. Die beiden IP-Subnetze werden als virtuelle Netze über denselben ATM-Switch realisiert. Das Gateway unterscheidet sich von der Packet Screen im Mechanismus, über den die Daten weitergeleitet werden. Beim Gateway werden die Daten auf der Anwendungsebene zwischen den beiden Netzen weitergeleitet. Bei der Packet Screen ist das Weiterleiten hingegen auf IP-Ebene (Vermittlungsschicht) realisiert.

Bei der Kombination von Packet Screen und Bastion werden Proxy-Server auf der Bastion (bzw. mehreren Bastionen) eingesetzt. Hier verfügt das System, auf dem der Proxy-Server zum Weiterleiten von Daten verwendet wird, nur über *eine* Netzwerkschnittstelle, so daß die Daten zweimal über dieselbe Schnittstelle übertragen werden müssen (s. Abb. 4.58). Obwohl es bei ATM möglich ist, auf derselben Schnittstelle gleichzeitig zu senden als auch zu empfangen (voll-duplex), sind Durchsatzeinbußen gegenüber dem Gateway-Setup zu erwarten. Die Ursache hierfür ist, daß während einer Messung Daten in beiden Richtungen übertragen werden: die Nutzdaten und die Empfangsbestätigungen.

Die meisten Performanzmessungen mit der Firewall-Komponente Proxy-Server werden im Bastion-Setup durchgeführt. Dieses Setup ist in Abbildung 4.58 wie-

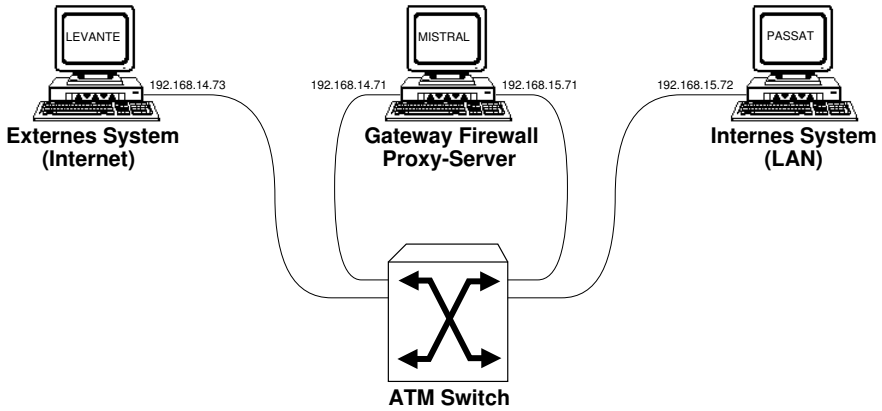


Abbildung 4.57: Testnetz-Konfiguration für Proxy-Server Einsatz bei einem Gateway-Firewall (Gateway-Setup)

dergegeben. Alle an der Messung beteiligten Systeme verwenden dasselbe Subnetz. Da Verbindungen auch an der Bastion vorbei direkt zum Empfänger eröffnet werden können, reicht der Einsatz einer Bastion allein nicht für den Aufbau eines Firewalls. In typischen Firewall-Architekturen wird die Bastion daher mit einer oder mehreren Packet Screens kombiniert. Die Packet Screens lassen nur Datenströme zu, die über die Bastion geführt werden. Zusätzlich zu den Performanzeinflüssen des Proxy-Servers bestimmen dann auch noch die Packet Screens das Performanzverhalten des Gesamtsystems. Bei komplexeren Firewall-Architekturen aus Packet Screens und Proxy-Servern werden Datendurchsatz, Transaktionsdurchsatz und die maximale Anzahl paralleler Verbindungen durch die schwächste Komponente begrenzt. Demgegenüber wird die Erhöhung der Nachrichtenlaufzeit mindestens so hoch sein, wie die Summe der Verzögerungen in den einzelnen Komponenten. In diesem Abschnitt sollen ausschließlich die Auswirkungen des Proxy-Servers auf die Performanz gemessen werden, um zunächst zu ermitteln, welche Effekte durch diese Firewall-Komponente bedingt sind. Für die Messungen der Performanz von Proxy-Servern auf Bastionen wird daher das in Abbildung 4.58 wiedergegebene Setup gewählt. Die Performanz von Kombinationen aus Packet Screen und Bastion werden im Abschnitt 4.5 untersucht.

Bei Proxies sind Datendurchsatz, Transaktionsdurchsatz, Verbindungsaufbauzeit, Nachrichtenlaufzeit und maximale Anzahl paralleler Verbindungen zu messen. Der Datagrammdurchsatz hat nur eine geringe Bedeutung für die Performanz, da die Daten vom Proxy auf Anwendungsebene verarbeitet werden und keine Verarbeitung auf der Basis einzelner Datagramme wie bei einer Packet Screen stattfindet. Während der Tests zeigte sich, daß der zunächst eingesetzte Proxy "Socks" keine ausreichende Experimentierplattform ist. Obwohl diese Standard-Software im Quellcode verfügbar ist und somit notwendige Ergänzungen und

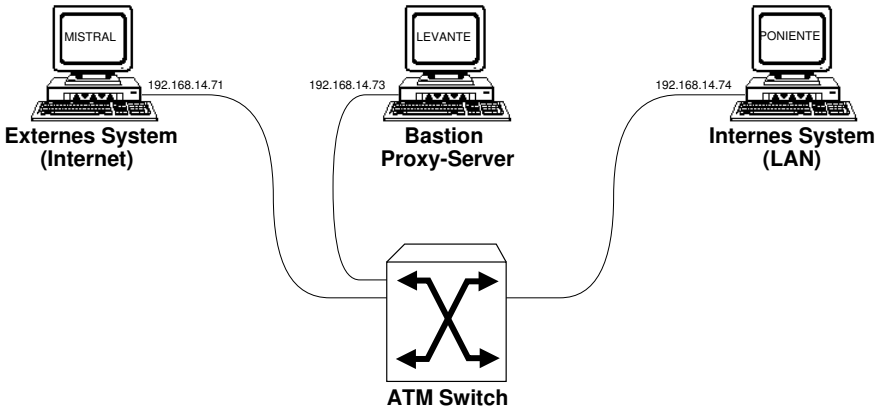


Abbildung 4.58: Testnetz-Konfiguration für Proxy-Server Einsatz auf einer Bastion (Bastion-Setup)

Anpassungen für Messungen möglich sind, führen Änderungen an der komplexen Implementation oft zu unerwünschten Nebenwirkungen. Daher wurde für weitergehende Tests ein eigener Proxy ("MGproxy") als äußerst flexible Experimentierplattform geschaffen. Zunächst werden die einfacheren Messungen des Datendurchsatzes, der Nachrichtenlaufzeit und der Verbindungsaufbauzeit anhand von "Socks" vorgestellt. Die komplexeren Messungen und zusätzlich alle bereits mit "Socks" durchgeführten Messungen werden anschließend bei "MGproxy" detailliert diskutiert.

4.4.1 Messungen mit "Socks"

Für die Messung der Performanz wird zunächst der Proxy-Server "Socks" (Version 1.0r4)[115] eingesetzt. "Socks" wurde ursprünglich als einfacher Proxy-Server zum Weiterleiten von TCP-Verbindungen konzipiert [Koblas et al. 92]. Da bei diesem Proxy-Server auf eine bei anderen Proxy-Servern durch Einmal-Paßworte oder kryptographische Verfahren übliche, sichere Authentisierung der Zugriffe verzichtet wird, ist der Einsatzbereich von "Socks" auf das Weiterleiten von Verbindungen vom internen Netz in das Internet beschränkt.[116]

Da "Socks" TCP-Verbindungen weiterleitet, ohne die über die TCP-Verbindung übertragenen Protokolle wie FTP oder Telnet zu kontrollieren, wird "Socks" auch als "Proxy auf Transportebene" bezeichnet. Eine Unterscheidung von "Proxy-

[115] Verfügbar unter: `http://www.socks.nec.com/cgi-bin/download.pl`

[116] "Socks v1.0r4" unterstützt inzwischen weitere Authentisierungsverfahren (z. B. Kerberos); der Haupteinsatzbereich bleibt jedoch das Weiterleiten von innen nach außen.

Servern auf Transportebene" und "Proxy-Servern auf Anwendungsebene" ist für die folgenden Performanzmessungen jedoch nicht relevant. Da sowohl bei "Proxy-Servern auf Transportebene" als auch "Proxy-Servern auf Anwendungsebene" die Daten von einem Anwendungsprozeß zwischen den beiden TCP-Verbindungen kopiert werden, unterscheiden sich "Proxy-Server auf Anwendungsebene" nur durch die zusätzlichen, anwendungsprotokollabhängigen Kontrollfunktionen. Vom Performanzgesichtspunkt stellt "Socks" als "Proxy-Server auf der Transportebene" gleichzeitig einen minimalen "Proxy-Server auf der Anwendungsebene" dar. Der Durchsatz des minimalen Proxy-Servers ist die Obergrenze von dem, was aufwendigere Proxy-Server erreichen können, da sich durch das Hinzufügen von protokollabhängigen Kontrollfunktionen die Performanz nur verschlechtern kann.

Den hier diskutierten Messungen liegt eine "Socks"-Standardinstallation ohne Optimierungen zugrunde. Um die Vergleichbarkeit der Meßergebnisse zu gewährleisten, werden dieselben Meßskripte verwendet, wie bei den Host-zu-Host-Messungen. Da bei Proxy-Servern die Performanz auf Anwendungsebene wichtig ist, kommen nur die Standard-Meßskripte für Anwendungsperformanz über TCP zum Einsatz. Die bei Messungen mit Packet Screens verwendete Reduzierung der MTU-Größe, die bestimmte Datagrammlängen auf Netzwerkebene garantiert, wird nicht genutzt. Da Proxy-Server auf Anwendungsebene wirken, ist auch nur die Anwendungsperformanz zu messen. Das Verhalten bei bestimmten Datagrammlängen ist im Gegensatz zu den Messungen bei Packet Screens nicht relevant.

Damit ein Client eine Verbindung über "Socks" aufbauen kann, muß der Client zunächst eine Verbindung zum Proxy-Server öffnen. Über diese Verbindung teilt er dem Proxy-Server über ein einfaches Protokoll [Leech et al. 96] die Zieladresse mit, zu der "Socks" dann eine Verbindung öffnet. Falls die Verbindung geöffnet werden konnte, wird dies dem Client mitgeteilt, und er kann mit dem Senden von Daten beginnen. Die Verbindung verhält sich für den Client genauso wie eine direkte Verbindung zum Zielrechner. Bei der vorliegenden "Socks"-Implementation ist eine Bibliotheksfunktion enthalten, die eine Verbindung über den "Socks"-Proxy-Server öffnet und dieselben Parameter wie "`connect`" verwendet. Durch die Verwendung derselben Parameter kann bei Systemen, die "Shared-Libraries" unterstützen, diese Bibliothek beim Programmstart hinzugebunden werden, und es ist keine Neuübersetzung der Client-Anwendungen erforderlich.[117]

[117] Die Möglichkeit, den Zugriff auf Proxy-Server ohne Neuübersetzung über "Shared-Libraries" zu realisieren, wurde bereits in [Ellermann 93] vorgestellt und ist inzwischen in der "Socks"-Software enthalten.

4.4.1.1 Datendurchsatz über "Socks"

Den folgenden Messungen des Datendurchsatzes, der Nachrichtenlaufzeit und der Verbindungsaufbauzeit bei "Socks" liegt ein Bastion-Setup (s. Abb. 4.58) zugrunde. Die Messungen des Datendurchsatzes zeigen eine deutliche Reduktion des Durchsatzes im Vergleich zu einer direkten Host-zu-Host-Messung (s. Abb. 4.59). Der maximale Datendurchsatz über eine von "Socks" verwaltete TCP-Verbindung beträgt 77 MBit/s und wird bei Schreibzugriffslängen im Bereich von 2 bis 8 KByte erreicht. Oberhalb von 8 KByte bricht der Durchsatz ab. Dies könnte mit der Länge der Kopieroperationen im Proxy zusammenhängen, die in 4 KByte langen Blöcken implementiert ist.[118] Die Auswirkungen der Länge der Kopieroperationen auf den Datendurchsatz von Proxy-Servern wird im Abschnitt 4.4.2 diskutiert.

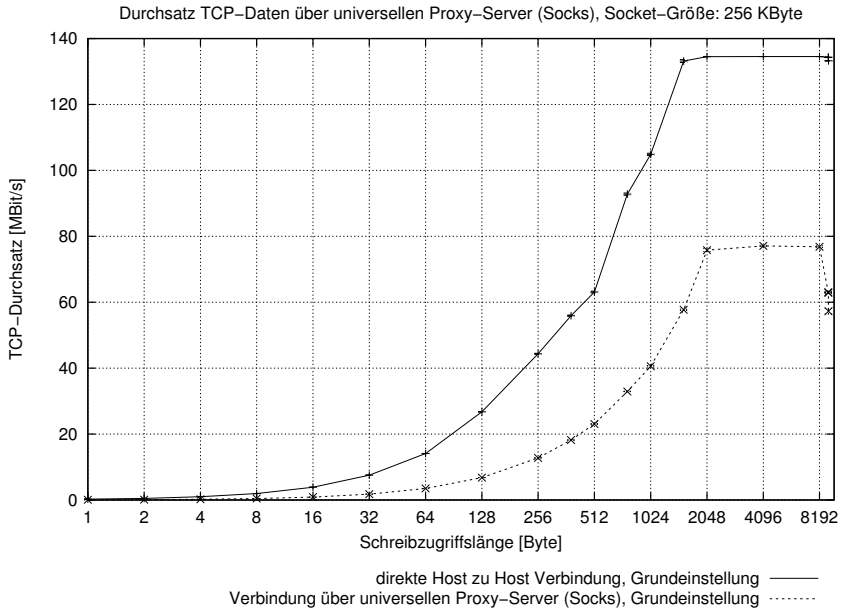


Abbildung 4.59: Datendurchsatz über Proxy-Server "Socks" im Vergleich zu einer direkten Host-zu-Host-Verbindung

4.4.1.2 Nachrichtenlaufzeit über "Socks"

Die Verlängerung der Nachrichtenlaufzeit über den Proxy-Server wird mit einer Umlaufzeitmessung (s. Abschnitt 4.2.3) ermittelt. Die drei in der Tabelle 4.8 ent-

[118] Die Schreibzugriffslängen von 2, 4 und 8 KByte lassen sich effizienter mit einer Kopierlänge von 4 KByte verarbeiten als die Schreibzugriffslängen von 9140 und 9150 Byte, bei denen ein "Verschnitt" entsteht.

haltenen, unterschiedlichen Konfigurationen des Proxy-Servers haben nur geringe Auswirkungen auf die Nachrichtenlaufzeiten. Die Multiprozeß- ("Prefork") und "Multithreaded"-Konfigurationen dienen zur Verringerung der Verbindungsaufbauzeiten. Ihre Funktionsweise wird im folgenden Abschnitt erläutert. Zum weiteren Vergleich sind in der Tabelle 4.8 auch die Anzahl der Umläufe pro Sekunde über eine Packet Screen (s. Abschnitt 4.3.4) und auf einer direkten Host-zu-Host-Verbindung (s. Abschnitt 4.2.3) wiedergegeben.

Umlaufmessung (30 Sek.) über:	Umläufe [Umläufe/s]	Nachrichten-laufzeit [s]
Socks Standard	823,31 $\pm 1,23$	$6,08 * 10^{-4}$
Socks Prefork	818,13 $\pm 5,47$	$6,11 * 10^{-4}$
Socks Multithreaded	805,02 $\pm 3,46$	$6,21 * 10^{-4}$
Host-zu-Host	2.593,47 $\pm 1,88$	$1,93 * 10^{-4}$
Packet Screen 0 Regeln	1.540,40 $\pm 3,85$	$3,25 * 10^{-4}$
Packet Screen 400 Regeln	835,96 $\pm 0,79$	$5,99 * 10^{-4}$

Tabelle 4.8: Nachrichtenumläufe über "Socks"

Durch den Proxy-Server ergibt sich eine zusätzliche Verzögerung von 0,4 Millisekunden gegenüber einer direkten Host-zu-Host-Verbindung. Dies entspricht ungefähr einer Verdreifachung der Nachrichtenlaufzeit. Der Proxy-Server bewirkt etwa die gleiche Verzögerung wie eine Packet Screen mit 400 Filterregeln. Der in der Messung verwendete Proxy-Server führt jedoch im Gegensatz zur Packet Screen noch keine Zugriffskontrolle des Verbindungsaufbaus durch, so daß alle Verbindungsaufbauwünsche von Clients gewährt werden. Da die Auswertung der Zugriffskontrollregeln bei einem Proxy-Server jedoch nur beim Aufbau einer neuen Verbindung erfolgt,[119] sind durch das Hinzufügen von Zugriffskontrollregeln keine Auswirkungen auf die Umlaufzeitmessung zu erwarten, da die Umläufe auf einer bereits geöffneten Verbindung gemessen werden. Eine anwendungsprotokollabhängige Kontrolle der Nachrichten*inhalte* wird dagegen eine weitere deutliche Verlängerung der Nachrichtenlaufzeiten zur Folge haben.

4.4.1.3 Verbindungsaufbauzeit über "Socks"

Stärker als auf den Datendurchsatz und die Nachrichtenlaufzeit wirken Proxy-Server noch auf die Verbindungsaufbauzeit. Während Packet Screens die Verbindungsaufbauzeiten nur indirekt über die Verlängerung der Nachrichtenlaufzeiten beeinflussen, greifen Proxy-Server direkt in den Verbindungsaufbau ein. Der

[119] Eine Packet Screen muß demgegenüber für jedes empfangene Datagramm die Filterregeln auswerten, bevor über die Weiterleitung des Datagramms entschieden werden kann.

Proxy-Server nimmt zunächst die Verbindung des Clients an, verifiziert gegebenenfalls die Identität des Clients (Authentisierung), überprüft die Berechtigung des Clients, die gewünschte Verbindung zu eröffnen (Autorisierung),[120] und eröffnet schließlich die Verbindung zum gewünschten Ziel.[121] Die vom Client gemessene Verbindungsaufbauzeit über den Proxy zum Server setzt sich somit zusammen aus den beiden Verbindungsaufbauzeiten (vom Client zum Proxy und vom Proxy zum Server) sowie der Verarbeitungszeit im Proxy. Die folgende Diskussion von Verbindungsaufbauzeiten über Proxy-Server bezieht sich immer auf das Eröffnen von beiden Verbindungen.

Der Nachrichtenfluß bei einer Transaktionsmessung über einen Proxy-Server ist in Abbildung 4.60 wiedergegeben. Im Vergleich zur Transaktionsmessung bei direkter Host-zu-Host-Kommunikation fallen die hintereinander stattfindenden Verbindungsaufbauphasen und die doppelte Länge der Datenübertragungsphase auf. Die Überschneidung von Datenübertragungsphase und Verbindungsabbauphase haben kaum Auswirkungen auf die Messung, da der Verbindungsabbau ohnehin asynchron erfolgt.

In der Standardkonfiguration von "Socks" wird nach dem Annehmen einer neuen Verbindung zuerst eine Kopie des Server-Prozesses angelegt (`fork`). Die Kopie des Prozesses ist anschließend für diese neue Verbindung zuständig, während der Server wieder für die Annahme weiterer Verbindungen zur Verfügung steht. Das Anlegen der Kopie des Server-Prozesses erhöht natürlich die Verbindungsaufbauzeiten erheblich, so daß nur noch knapp 35 Verbindungen pro Sekunde[122] verarbeitet werden können (s. Tab. 4.9). Die Meßergebnisse werden mit der in Abschnitt 4.2.4 vorgestellten Transaktionsmessung ermittelt.[123] [124] Aus der Anzahl

[120] Für Authentisierung, Autorisierung und Audit benötigt der Proxy die Hostnamen vom Client und vom Server, da zum Beispiel die erforderlichen Authentisierungsinformationen in der Regel nicht unter IP-Adressen, sondern in Form von logischen Hostnamen gespeichert sind. Die hierfür erforderlichen DNS-Anfragen des Proxies verzögern Transaktionen erheblich.

[121] Die Zieladresse ist entweder fest konfiguriert oder kann vom Client in einer weiteren Nachricht dem Proxy-Server mitgeteilt werden.

[122] Alle Transaktionsmessungen über "Socks" beziehen sich auf eine Testnetz-Konfiguration als Bastion (Abb. 4.58).

[123] Das `tcp_close_wait_intervall` wird bei der Proxy-Messung, wie auch bei den Messungen auf direkten Host-zu-Host-Verbindungen (s. Abschnitt 4.2.4), auf 30 Sekunden verringert, um zu verhindern, daß alle Ports gleichzeitig im Zustand "TIME-WAIT" blockiert sind.

[124] Da vom Proxy-Server für jede Verbindung drei Zeilen mit Verbindungsdaten (Adressen und Ports) aufgezeichnet werden, wächst die Log-Datei während der Messung um fast 1 MByte pro Minute. Längere Messungen können durch das Füllen der Platte Probleme verursachen. Für die Messungen wurde die Aufzeichnung der Log-Informationen abgeschaltet. Dadurch ist auch sichergestellt, daß Plattenzugriffszeiten und andere I/O-bedingte Wartezeiten, die nichts mit der Protokollverarbeitung zu tun haben, die Messung beeinflussen.

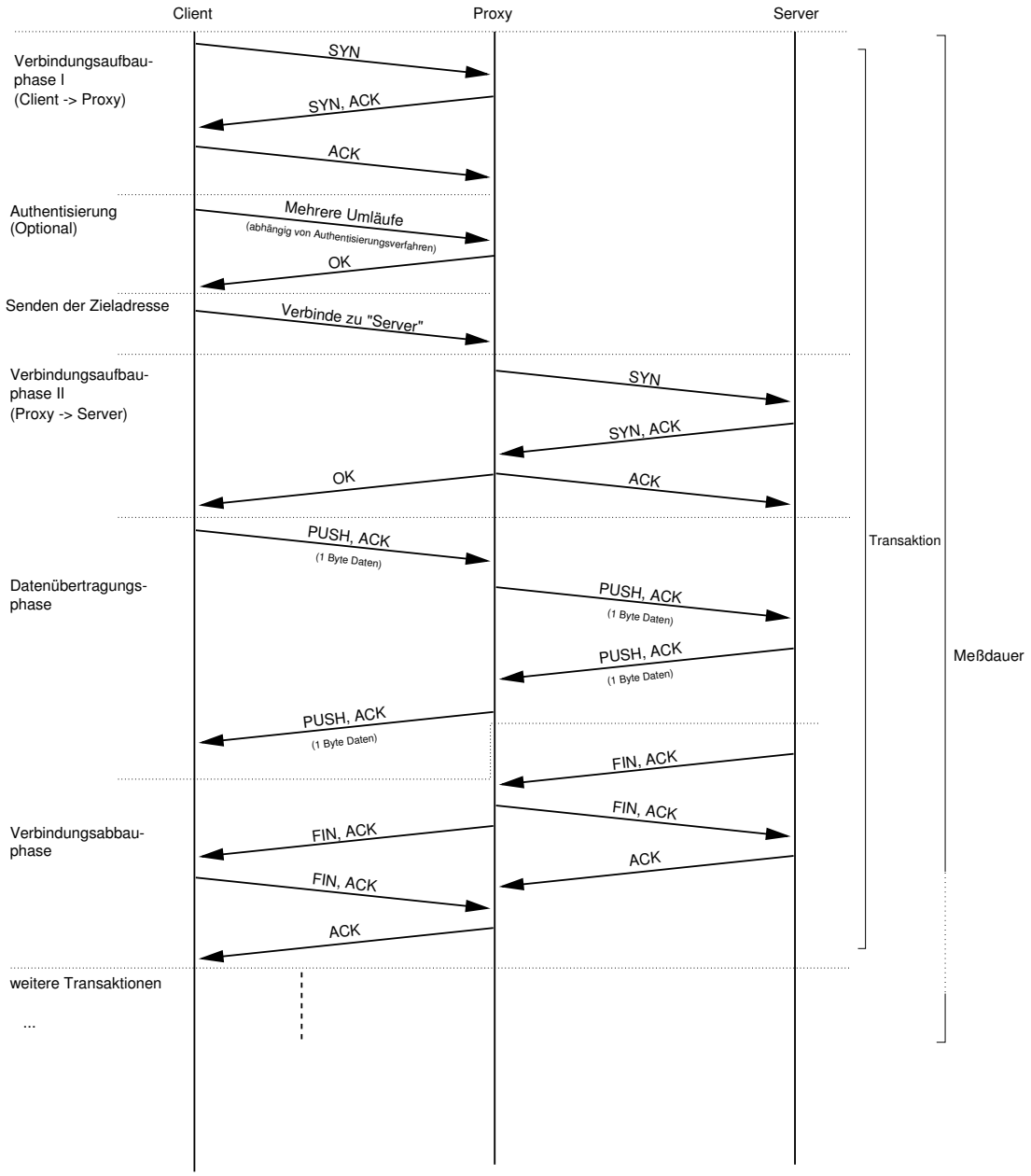


Abbildung 4.60: Transaktionsmessung über Proxy-Server

der sequentiell durchgeführten Transaktionen pro Sekunde kann die Dauer einer einzelnen Transaktion (Transaktionslänge) mit $2,86 * 10^{-2}$ Sekunden berechnet werden. Da bei der Transaktionsmessung nach dem Verbindungsaufbau ein Umlauf durchgeführt wird, muß zur Ermittlung der Verbindungsauf- und -abbauzeit die Nachrichtenumlaufzeit (s. Abschnitt 4.4.1.2) von der Transaktionslänge abgezogen werden.

Transaktionsmessung (180 Sek.) über:	Transaktionen [Tr/s]	Transaktionslänge [s]	Verbindungsaufbauzeit [s]
Socks Standard	34,89 $\pm 0,05$	$2.86 * 10^{-2}$	$2,74 * 10^{-2}$
Socks Inetd	3,34 $\pm 0,30$	$3,03 * 10^{-1}$	$(3,02 * 10^{-1})^*$
Socks Prefork	88,69 $\pm 0,35$	$1,12 * 10^{-2}$	$1,00 * 10^{-2}$
Socks Multithreaded	93,75 $\pm 0,48$	$1,06 * 10^{-2}$	$0,94 * 10^{-2}$
Host-zu-Host	517,41 $\pm 1,45$	$1,93 * 10^{-3}$	$1,55 * 10^{-3}$
Packet Screen 0 Regeln	409,49 $\pm 0,65$	$2,44 * 10^{-3}$	$1,80 * 10^{-3}$
Packet Screen 400 Regeln	248,88 $\pm 0,06$	$4,02 * 10^{-3}$	$2,82 * 10^{-3}$
Packet Screen 1.000 Regeln	115,56 $\pm 0,04$	$8,62 * 10^{-3}$	$6,60 * 10^{-3}$

Tabelle 4.9: Transaktionen über "Socks" (Bastion-Setup)

Beim Start des Proxy-Servers über "Inetd"[125] sind die Verbindungsaufbauzeiten noch um den Faktor 10 länger.[126] Bei jeder neuen Verbindung muß "Inetd" den Proxy-Server laden und ausführen. Der Proxy-Server muß vor der Bearbeitung der Verbindung zunächst die notwendigen Initialisierungen durchführen. Das Starten des Proxy-Servers über "Inetd" kann daher nicht empfohlen werden, wenn ein hoher Durchsatz an Transaktionen pro Sekunde erreicht werden muß.

Bei der Standardkonfiguration führt das Kopieren des Server-Prozesses (`fork`) zur Erhöhung der Verbindungsaufbauzeit. Die Verbindungsaufbauzeit kann verringert werden, wenn bereits vor dem Eintreffen einer Verbindung eine Kopie des Server-Prozesses erzeugt wird. In der Multiprozeß-Konfiguration ("Prefork"-Konfiguration) werden bereits unmittelbar nach dem Start des Proxy-Servers ausreichend viele (64) Server-Prozesse gestartet, so daß die Bearbeitung einer neuen Verbindung ohne Verzögerung beginnen kann. Hierdurch wird fast eine Verdreifachung auf 89 Transaktionen pro Sekunde erreicht.

Der Nachteil bei diesem Vorgehen ist, daß die Bastion auch dann alle Proxy-Server-Prozesse verwalten muß, wenn der Proxy-Server zur Zeit nicht benutzt wird. Diese Verschwendung von Betriebsmitteln wird bei der "Multithreaded"-

[125] "Inetd" ist ein Server, der an mehreren Ports auf Anfragen von Clients wartet und erst beim Eintreffen einer Client-Anfrage den für den jeweiligen Dienst zuständigen Server startet. Dadurch wird insbesondere bei selten genutzten Diensten vermieden, daß eine Vielzahl wartender Prozesse das System unnötig belasten.

[126] Die Umlaufzeiten über "Inetd" wurden nicht gemessen. Für die Berechnung der Verbindungsaufbauzeit wurde ersatzweise die Umlaufzeit bei der Standardkonfiguration von "Socks" verwendet. Die Umlaufzeiten der beiden Konfigurationen werden sich kaum voneinander unterscheiden, da der Verarbeitungsaufwand etwa gleich ist. Im Gegensatz dazu belasten die Multiprozeß- und die "Multithreaded"-Konfiguration das System deutlich stärker.

Konfiguration von "Socks" verringert. "Threads"[127] laufen innerhalb eines Prozesses ab, so daß auch nur eine Kopie des Proxy-Servers im Speicher abgelegt werden muß. Für jeden "Thread" müssen nur die exklusiv vom "Thread" benötigten Daten gespeichert werden. Insgesamt ist der Einsatz von mehreren Threads sehr viel "leichtgewichtiger" als für jede Verbindung einen zusätzlichen Prozeß zu verwenden. Daher können in der "Multithreaded"-Konfiguration von "Socks" auch noch geringfügig mehr Transaktionen pro Sekunde erreicht werden als bei der Multiprozeß-Konfiguration.

4.4.2 Messungen mit "MGproxy"

Seit der Erstellung des ersten Konzeptes für den im vorigen Abschnitt für die Messungen verwendeten Proxy-Server "Socks" [Koblas et al. 92] sind diverse Ergänzungen und Verbesserungen an diesem Proxy-Server erfolgt. Inzwischen umfaßt der Quellcode über 5.000 Zeilen (einschließlich Kommentaren) für den Proxy-Server und weitere 9.000 Zeilen für diverse Bibliotheksfunktionen. Es kann also bei "Socks" nicht mehr von einem *minimalen* Proxy-Server gesprochen werden. Ein weiterer Nachteil dieser Komplexität ist, daß nachträgliche Ergänzungen schwierig durchzuführen sind und unerwartete Effekte auftreten können. Aus diesem Grund wurde mit der Entwicklung eines minimalen, generischen Proxy-Servers ("MGproxy") begonnen.

Die wichtigsten Komponenten, die für einen Proxy-Server benötigt werden, sind:

- ein Server, der Verbindungen annimmt und für die Bearbeitung jeder neuen Verbindung einen parallelen Prozeß initialisiert,
- eine "Socks"-Protokoll-Implementation, die die gewünschte Zieladresse beim Client erfragt,
- eine Funktion zum Öffnen einer Verbindung zur gewünschten Zieladresse,
- eine Kopierfunktion zum Kopieren der Daten zwischen den beiden geöffneten Verbindungen.

Diese elementaren Funktionen eines Proxy-Servers sind bei "MGproxy" in weniger als 800 Zeilen Quellcode implementiert worden. Darin sind bereits verschiedene Ergänzungen enthalten, um bei späteren Messungen wesentliche Parameter,

[127] Das Wort "Thread" kann nur unzureichend mit "Gedankengang" oder "Ablauffaden" übersetzt werden. Ein "Multithreaded"-Prozeß verwaltet die Kontextinformationen für mehrere Abläufe, die parallel innerhalb desselben Prozesses ablaufen können. [Coulouris et al. 94]

wie Socket-Größen und Kopierlängen des Proxy-Servers über Kommandozeilen-Optionen steuern zu können. Diese Implementation bildet gleichzeitig eine flexible Plattform für die Erprobung von Design-Alternativen und für die Entwicklung von Prototypen für parallele Proxy-Server.

4.4.2.1 Datendurchsatz über "MGproxy"

Durch die Unterstützung des "Socks"-Protokolls kann "MGproxy" in den Testkonfigurationen "Socks" direkt ersetzen, weitere Veränderungen sind nicht notwendig. Auch der Zugriff des Meßwerkzeuges "Netperf" auf den Proxy-Server über "Shared-Libraries" kann übernommen werden.

Bei den Datendurchsatzmessungen mit "MGproxy" wird analog zur Variation der Schreibzugriffslänge des Meßwerkzeuges zusätzlich auch die Kopierlänge des Proxy-Servers variiert. Die Kopierlänge gibt an, wieviele Daten maximal mit einem Zugriff vom Proxy-Server zwischen den beiden Verbindungen kopiert werden können. Das Einlesen von Daten von einer Verbindung ist bei "MGproxy" über den Systemaufruf "`recv`" implementiert.[128] Über einen Parameter wird beim `recv` angegeben, wieviele Daten maximal von einer Verbindung gelesen werden dürfen (Kopierlänge). Wenn weniger Daten anliegen, liest `recv` nur die zur Verfügung stehenden Daten ein. Die eingelesenen Daten werden anschließend sofort auf der zweiten Verbindung wieder herausgesendet. Da keine Daten über den `send`-Aufruf hinaus zwischengespeichert werden, begrenzt die Kopierlänge auch die Schreibzugriffslänge beim Heraussenden auf der zweiten Verbindung. Im Idealfall sind Kopierlänge, die tatsächlich gelesene Menge von Daten und die Schreibzugriffslänge gleich. Wenn jedoch nur eine kleinere Dateneinheit angekommen ist, kann die tatsächlich gelesene Menge ($Länge_{gelesen}$) von Daten geringer sein als die Kopierlänge. Die eingelesene Datenmenge wird anschließend in einem Schreibzugriff an TCP übergeben. (Es gilt Gleichung 4.19.) TCP kann aufgrund verschiedener Beschränkungen gezwungen sein, die übergebenen Daten in mehrere TCP-Segmente aufzuteilen; es können jedoch auch über Aggregation ("Nagle") größere Segmente erzeugt werden. Wie bereits die Diskussion bei den Host-zu-Host-Messungen (s. Abschnitt 4.2.2) gezeigt hat, ist die Länge des Schreibzugriffs der wichtigste Parameter bei Messungen des maximalen Datendurchsatzes.

[128] Alternativ könnte auch der Systemaufruf "`read`" verwendet werden, der nicht nur für Sockets sondern für unterschiedliche Arten von Datenströmen (Dateien, Devices, Sockets, etc.) verwendet werden kann. Bei den Systemaufrufen "`read`" und "`write`" statt "`recv`" und "`send`" entsteht ein geringfügig höherer "Overhead", der den maximalen Durchsatz aber nur um ≤ 1 MBit/s verringert. Ab einer Schreibzugriffslänge von 8 KByte werden mit beiden Verfahren dieselben Durchsätze erreicht. Hier macht sich der größere "Overhead" von "`read`"-"`write`" nicht so stark bemerkbar, da wenige Aufrufe der Funktion ausreichen, um die Daten umzukopieren.

$$Kopierlänge \geq Länge_{gelesen} = Schreibzugriffslänge \tag{4.19}$$

Bei den Datendurchsatzmessungen mit "MGproxy" in einem Gateway-Setup (s. Abb. 4.57) werden folgende Kopierlängen gemessen:

- Mit einer Kopierlänge von einem Byte kann natürlich nur ein sehr geringer Durchsatz erreicht werden. Trotzdem implementieren manche Proxy-Server ein byteweises Umkopieren.[129] Mit "MGproxy" kann ein Datendurchsatz von 0,1 MBit/s bei byteweisem Umkopieren erreicht werden. Dieser Wert ist unabhängig von der beim Meßwerkzeug verwendeten Schreibzugriffslänge. Zum Vergleich wird auf einer direkten Host-zu-Host-Verbindung mit einer Schreibzugriffslänge von einem Byte ein Durchsatz von 0,26 MBit/s erzielt.

- Die Kopierlänge von 70 Byte simuliert ein zeilenweises Kopieren der Daten. Ab einer Schreibzugriffslänge des Meßwerkzeuges von 128 Byte wird bei dieser Kopierlänge ein konstanter Durchsatz von etwa 7 MBit/s erreicht. Auf einer direkten Host-zu-Host-Verbindung kann mit der Schreibzugriffslänge von 64 Byte[130] ein Durchsatz von knapp 15 MBit/s erreicht werden. Die Verbindung durch den Proxy-Server erreicht also nur knapp die Hälfte des Durchsatzes einer direkten Host-zu-Host-Verbindung. Dies entspricht der doppelten Belastung des Systems durch die zwei Verbindungen.

- Ab einer Kopierlänge von 512 Byte ist erkennbar, daß alle Messungen unabhängig von der Kopierlänge zunächst denselben Kurvenverlauf folgen, bis die Schreibzugriffslänge des Meßwerkzeuges die Kopierlänge des Proxy-Servers überschreitet und der Datendurchsatz ab diesem Punkt durch die Kopierlänge des Proxy-Servers bestimmt wird (s. Abb. 4.61).[131] Bei den Kopierlängen von 512 Byte und 1 KByte können die Durchsätze von Schreibzugriffslängen oberhalb der Kopierlänge noch etwas höher ansteigen, weil durch die Verzögerungen des Proxy-Servers die Aggregation von Schreibzugriffen zu größeren TCP-Segmenten begünstigt wird. Da in dem Bereich von Schreibzugriffslängen bis 1.500 Byte auch noch der Durchsatz auf den

[129] Ein byteweises Kopieren ist zum Beispiel beim "http-gw" des TIS-FWTK (V 2.0) implementiert.

[130] Der Datendurchsatz auf einer direkten Host-zu-Host-Verbindung wurde für eine Schreibzugriffslänge von 70 Byte nicht ermittelt.

[131] Die Abbildung 4.61 beginnt bei einer Schreibzugriffslänge von 512 Byte. Der Bereich unterhalb von 512 Byte zeigt ein weiteres Abfallen der Kurven. Bei 128 Byte Schreibzugriffslänge fällt die 70 Byte Kurve mit den anderen Kurven zusammen. Zur Verbesserung der Übersichtlichkeit sind in der Abbildung 4.61 keine Konfidenzintervalle angegeben. Die Meßergebnisse mit angegebenen Konfidenzintervallen finden sich auf Seite 469 im Anhang B.

direkten Host-zu-Host-Verbindungen ansteigt, sind hier die Aggregationseffekte besonders deutlich zu erkennen. Für die Kopierlänge von 512 Byte wird ab einer Schreibzugriffslänge von 768 Byte der maximale Durchsatz von etwa 32 MBit/s erreicht. Zum Vergleich sind bei 512 Byte Schreibzugriffen auf einer direkten Host-zu-Host-Verbindung bereits 65 MBit/s erreichbar.

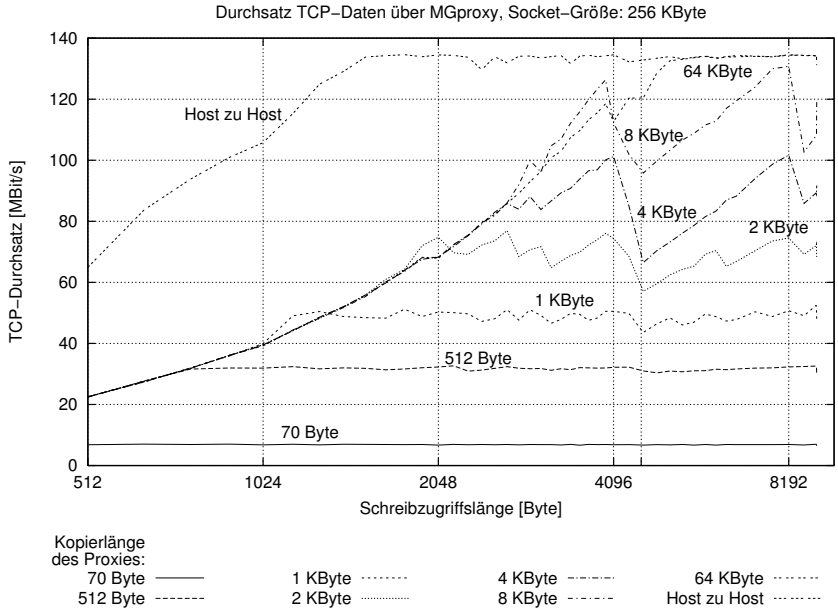


Abbildung 4.61: Durchsatz über "Minimalen Generischen Proxy" ("MGproxy")

- Auch bei der Kopierlänge von 1 KByte wird mit 50 MBit/s etwa die Hälfte des auf einer direkten Verbindung möglichen Datendurchsatzes von 106 MBit/s erreicht.
- Bei den bisher diskutierten Messungen mit Kopierlängen unterhalb von 2 KByte wird nach Erreichen des maximalen Durchsatzes dieser Wert konstant auch bei längeren Schreibzugriffen erreicht. Die Messungen mit Kopierlängen von 2, 4 und 8 KByte erreichen diesen konstanten Durchsatz jedoch nicht mehr; hier weichen bei benachbarten Meßpunkten die Datendurchsätze mitunter stark voneinander ab (s. Abb. 4.61). Diese Abweichungen sind durch mehrere Effekte bedingt:
  - Zum einen können Schreibzugriffslängen, die einem ganzzahligen Vielfachen der Kopierlänge entsprechen, ohne Rest in Blöcken mit Kopierlänge vom Proxy-Server weitergeschickt werden. So können 6.144 Byte bei eine Kopierlänge von 2 KByte in drei Schreibzugriffen übertragen werden. Dabei wird ein Durchsatz von 70 MBit/s erreicht.

Bleibt jedoch ein kleiner Rest, wie zum Beispiel bei der Schreibzugriffslänge von 6.400 Byte, so werden bei einer Kopierlänge von 2 KByte vier Lese- und Schreibzugriffe vom Proxy-Server durchgeführt. Da beim vierten Lesezugriff, sofern noch keine weiteren Daten vom Meßwerkzeug eingetroffen sind, nur noch 256 Byte gelesen werden können, ist der Datendurchsatz hier mit nur 65 MBit/s geringer (s. Abb. 4.61, Schreibzugriffslänge 6.400, Kurve 2 KByte).

- Ein weiterer Grund für die Abweichungen bei benachbarten Schreibzugriffslängen liegt in der Möglichkeit, daß die an der Messung beteiligten Systeme mehrere Schreibzugriffe des Meßwerkzeuges zu einer größeren Dateneinheit aggregieren (s. Abschnitt 4.2.2.2). So können zum Beispiel zwei Schreibzugriffe einer Länge bis 4 KByte in einem Datagramm aggregiert werden, das der Proxy-Server bei einer Kopierlänge von 8 KByte mit einer Kopieroperation übertragen kann. Werden bei Schreibzugriffen über 4 KByte mehr Daten in einer Dateneinheit übertragen, muß der Proxy-Server die Daten in mehreren Kopieroperationen übertragen. Sind bis zum zweiten Lesezugriff des Proxy-Servers weitere Daten eingetroffen, so können diese zusammen mit dem Rest der vorigen Dateneinheit in einer Operation mit der durch die Kopierlänge angegebenen Anzahl an Byte übertragen werden.
- Die Aggregation von zwei Schreibzugriffen des Meßwerkzeuges zu einer Dateneinheit kann jedoch nur dann stattfinden, wenn dadurch nicht die MTU-Größe von 9.180 Byte überschritten wird. Über eine TCP-Verbindung können also maximal zwei 4.570 Byte lange Schreibzugriffe zum Erreichen der MTU-Größe aggregiert werden. Oberhalb dieses Wertes, der in Abbildung 4.61 als zusätzliche vertikale Gitternetzlinie eingezeichnet ist, wird für jeden Schreibzugriff des Meßwerkzeuges genau ein Datagramm erzeugt; eine Aggregation findet nicht statt. Dies führt zu deutlich erkennbaren Einbrüchen bei Kopierlängen von 2, 4 und 8 KByte, sowie zu einem geringeren Einbruch auch bei 1 KByte Kopierlänge.

- Bei der maximalen Kopierlänge von 64 KByte sind die bei geringeren Kopierlängen beobachteten Effekte nicht mehr erkennbar. Ab einer Schreibzugriffslänge von 5.376 Byte wird ein Durchsatz von 133 MBit/s erreicht, was dem Datendurchsatz auf einer direkten Host-zu-Host-Verbindung entspricht. Der maximale Durchsatz wird bei 8 KByte langen Schreibzugriffen mit 134,44 MBit/s erreicht. Dieser Wert liegt nur knapp unter den 134,55 MBit/s, die bei der Vergleichsmessung über eine direkte Host-zu-Host-Verbindung gemessen werden. Mit der Kopierlänge von 8 KByte werden zum Vergleich nur maximal 130,75 MBit/s erreicht.

Die Kopierlänge von 64 KByte ermöglicht die beste Entkoppelung der Verarbeitung von Datagrammen im Kernel von den Kopieroperationen im Proxy-Server. Nachdem der Kernel einen bis zu 64 KByte großen Datenblock erhalten hat, kann er daraus Dateneinheiten (TCP-Segmenten) erzeugen, ohne daß der Proxy-Server als Applikation davon betroffen ist. Im günstigsten Fall sind beim nächsten Lesezugriff des Proxy-Servers bereits mehrere Dateneinheiten vom Meßwerkzeug angekommen, so daß erneut ein großer Datenblock kopiert werden kann. Dies hat zur Folge, daß die vom Proxy-Server weitergesendeten Daten in größeren Datagrammen transportiert werden als die vom Proxy-Server empfangenen Daten, deren Datagrammlänge durch die Schreibzugriffslänge des Meßwerkzeuges bestimmt wird.

Der Durchsatz bei Schreibzugriffen mit 4 KByte Länge ist jedoch bei 8 KByte Kopierlänge höher, als bei 64 KByte Kopierlänge (s. Ausschnitt Abb. 4.62). Dieses Phänomen ist noch nicht endgültig geklärt. Es ist anzunehmen, daß bei 8 KByte Kopierlänge ein höherer Grad der Nebenläufigkeit zwischen Kernel und Anwendungsprozeß erreicht werden kann, da bei der Kopierlänge von 64 KByte häufiger mit Unterbrechungen gerechnet werden muß.

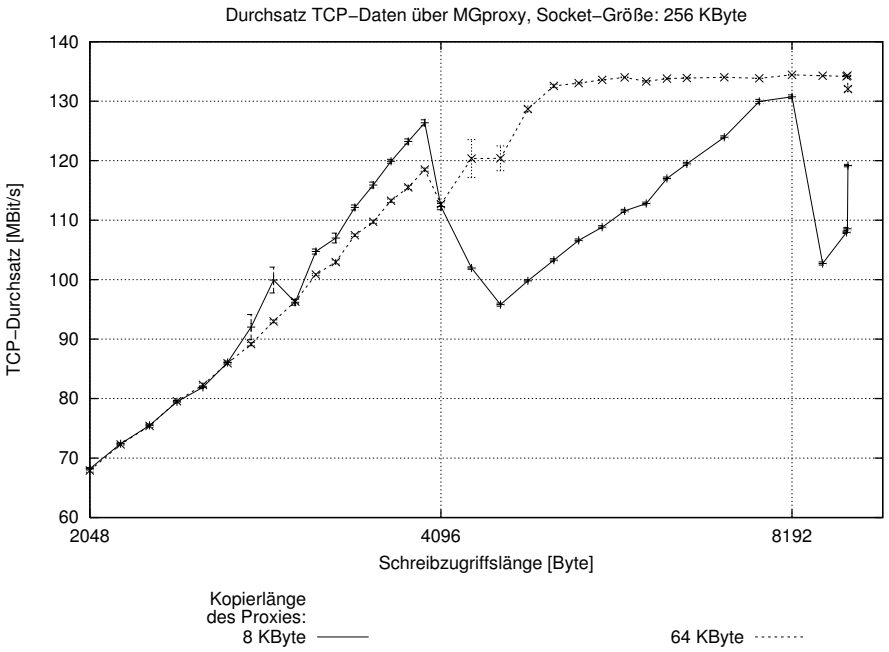


Abbildung 4.62: Durchsatz über "MGproxy" (Ausschnitt)

4.4.2.1.1 Auswirkungen von Sende- und Empfangspuffergrößen

Die Variation von Socket-Größen des Proxy-Servers gegenüber den Socket-Größen des Meßwerkzeuges zeigte bei den Messungen im Testnetz keine erkennbaren Wirkungen. Dies entspricht den Erfahrungen bei direkten Host-zu-Host-Verbindungen (s. Abschnitt 4.2.2.4).

Wichtiger sind die Auswirkungen, wenn die Kopierlänge des Proxy-Servers die Socket-Größe des Meßwerkzeuges überschreitet. Dieser Fall ist in Abbildung 4.63 wiedergegeben. Die Ursache liegt vermutlich in dem Anfüllen und Leeren der Socket-Puffer, die entsprechend der Socket-Größe reserviert werden. Bei gefülltem Puffer werden weitere Schreibzugriffe des Meßwerkzeuges blockiert, was zu den beobachteten Einbrüchen führt, die zum Teil bis auf 25 MBit/s heruntergehen.[132] Der Effekt verschwindet, wenn die Socket-Größe des Proxy-Servers auf den selben Wert von 8 KByte reduziert wird, der auch beim Meßwerkzeug verwendet wird. Dies verhindert, daß sich beim Proxy-Server mehr als 8 KByte Daten ansammeln, die von einem 64 KByte langem Zugriff gelesen werden. Durch die Reduktion der Socket-Größe wird indirekt also auch die Kopierlänge auf 8 KByte reduziert.

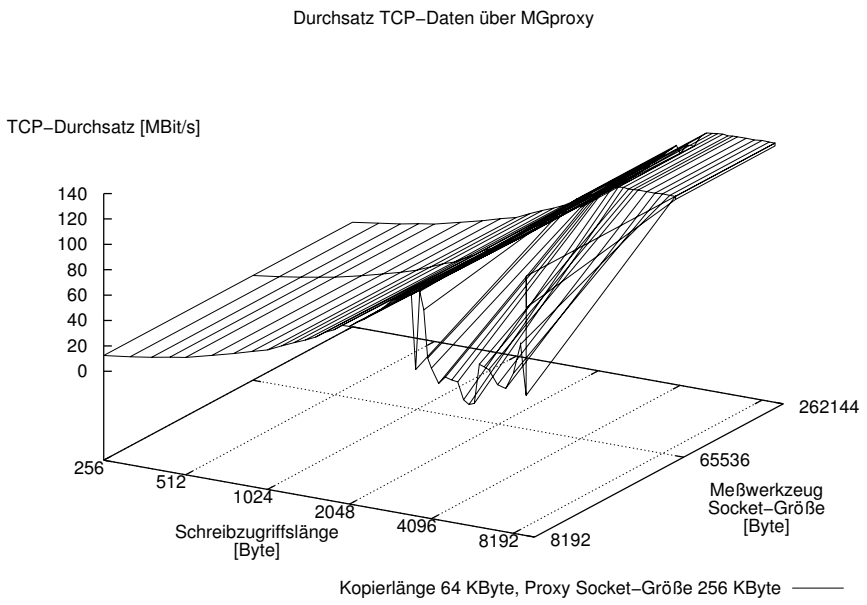


Abbildung 4.63: Durchsatz bei "MGproxy": Kopierlänge überschreitet Socket-Größe des Meßwerkzeuges

[132] Diese Messung ähnelt den Messungen mit asymmetrischen Sende- und Empfangspuffern in Abschnitt 4.2.2.4.

Ein wichtiger Aspekt bei der Auswahl einer geeigneten Puffergröße stellt der erforderliche Speicherplatz dar. Für jede Verbindung, die über den Proxy aufgebaut wird, muß der Proxy den Empfangs- und den Sendepuffer für die Verbindung zum Client und dieselbe Speichermenge ebenfalls für die Verbindung zum Server reservieren. Insgesamt wird somit die vierfache Speichergröße der eingestellten Puffergröße benötigt. Zusätzlich wird für den Kopierpuffer und für den Proxy-Prozeß selbst weiterer Speicher benötigt. Wenn Sende- und Empfangspuffer zu groß gewählt werden, so reicht der Speicher der Bastion nur für eine geringe Anzahl paralleler Verbindungen aus, die gleichzeitig vom Proxy kontrolliert werden können. Wenn der Speicher der Bastion nicht mehr ausreicht und Teile des Speichers daher auf eine Platte ausgelagert werden müssen, wird dies starke Auswirkungen auf die Performanz haben.

Aufgrund der geringen Nachrichtenlaufzeiten im LAN zwischen Client und Proxy sowie zwischen Proxy und Server reichen auch geringe Socket-Größen aus. Da der Haupteinsatzbereich von Firewalls jedoch am Übergang zwischen LAN und WAN (Internet) liegt, sind wegen der höheren Verzögerungen im WAN größere Sende- und Empfangspuffer notwendig. Um den Speicherbedarf des Proxies gering zu halten, erscheint es daher empfehlenswert, nur für Verbindungen im WAN größere Sende- und Empfangspuffer zu reservieren. Da Clients im LAN nur über den Proxy mit Systemen im WAN kommunizieren, müssen keine Einstellungen der Puffergrößen der Clients vorgenommen werden, um auch bei der Kommunikation mit Systemen im WAN hohe Durchsätze zu erzielen. Da keine Messungen im WAN durchgeführt werden konnten, können die Auswirkungen unterschiedlich großer Puffer für Verbindungen im LAN oder im WAN nicht endgültig bewertet werden.

4.4.2.1.2 Datendurchsatz beim Bastion-Setup

Das Bastion-Setup (s. Abb. 4.58) unterscheidet sich vom Gateway-Setup (s. Abb. 4.57) durch die fehlende zweite Netzwerkschnittstelle in der Bastion. Die Daten werden zwar vom Proxy-Server wie beim Betrieb im Gateway-Firewall auf einer Verbindung angenommen und auf einer zweiten Verbindung weitergeleitet, die Daten beider Verbindungen werden dabei jedoch über dieselbe ATM-Schnittstelle übertragen. Dadurch ist jedoch nicht mit einer Halbierung des Durchsatzes zu rechnen, da die ATM-Schnittstelle den angegebenen Durchsatz im Voll-Duplex-Betrieb ermöglicht, d. h. über dieselbe 155 MBit/s ATM-Schnittstelle können gleichzeitig 155 MBit/s empfangen *und* gesendet werden.

Trotz der Möglichkeit, im Voll-Duplex-Betrieb 155 MBit/s gleichzeitig zu empfangen und zu senden, ist der Datendurchsatz beim Bastion-Setup geringer (s. Abb. 4.64). Statt 134,44 MBit/s ist nur noch ein maximaler Durchsatz von 133,33 MBit/s möglich. Da bei Messungen über TCP in der Gegenrichtung Empfangs-

bestätigungen gesendet werden müssen, kann nicht der gesamte zur Verfügung stehende Durchsatz der Schnittstelle für die Übertragung der Nutzdaten verwendet werden. Der Verlust von unter einem Prozent gegenüber einer Gateway-Messung ist so gering, daß der Einsatz einer Bastion als kostengünstige Alternative zum Gateway-Firewall und auch wegen der größeren Flexibilität trotzdem empfohlen werden kann.

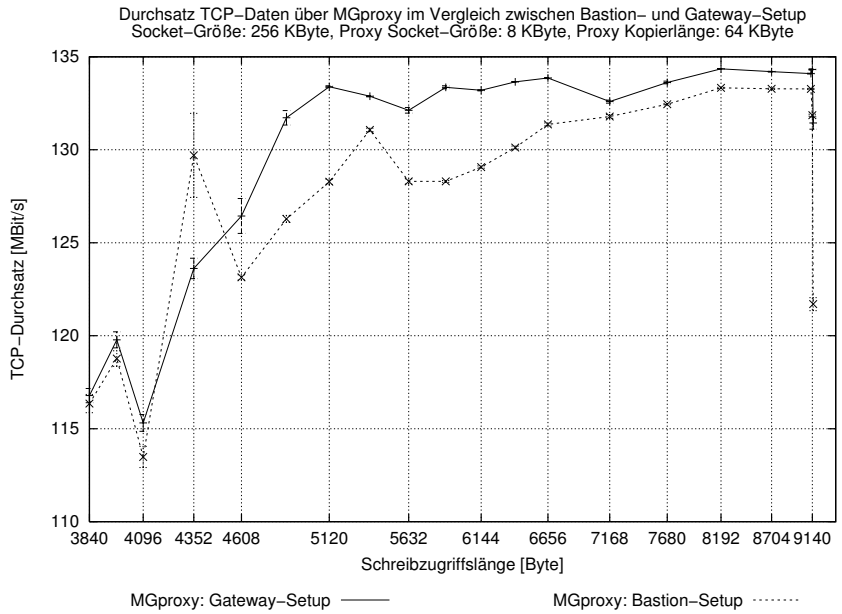


Abbildung 4.64: Durchsatz über "MGproxy": Vergleich zwischen Bastion und Gateway Setup

Interessant ist die Schreibzugriffslänge von 4.352 Byte. Bei dieser Schreibzugriffslänge kann mit fast 130 MBit/s der beim Gateway-Setup ermittelte Durchsatz von 124 MBit/s übertroffen werden. Zu vermuten ist auch hier wieder ein Effekt, der mit der Aggregation mehrerer Schreibzugriffe und den Auswirkungen von Pufferspeichergrößen zusammenhängt. Diese Annahme wird dadurch bestätigt, daß durch die Reduktion der Socket-Größe beim Meßwerkzeug der Durchsatz über die Bastion an diesem Punkt sinkt.

4.4.2.2 Nachrichtenlaufzeit über "MGproxy"

Der Proxy-Server erreicht beim Gateway-Setup mit 1.002,03 $\pm 1,75$ Umläufen pro Sekunde — dies entspricht einer Nachrichtenlaufzeit von $4,99 * 10^{-4}$ Sekunden — eine um knapp 20% bessere Nachrichtenlaufzeit als "Socks", mit dem maximal 823 Umläufe pro Sekunde erreicht werden können. Dieses Ergebnis ist

auf die kompaktere und effizientere Implementierung der Kopierroutine zurückzuführen. Beim Einsatz von "MGproxy" auf einer Bastion wurde mit 998,80 $\pm 0,36$ Umläufen pro Sekunde ein nur geringfügig schlechteres Ergebnis ermittelt.

Umlaufmessung (30 Sek.) über:	Umläufe [Umläufe/s]	Nachrichten-laufzeit [s]
MGproxy Gateway	1.002,03 $\pm 1,75$	$4,99 * 10^{-4}$
MGproxy Bastion	998,80 $\pm 0,36$	$5,01 * 10^{-4}$
Host-zu-Host	2.593,47 $\pm 1,88$	$1,93 * 10^{-4}$
Socks Standard	823,31 $\pm 1,23$	$6,08 * 10^{-4}$

Tabelle 4.10: Nachrichtenumläufe über "MGproxy"

Trotz dieses guten Ergebnisses bleiben beide Werte weit hinter den 2.593 Umläufen pro Sekunde zurück, die auf einer direkten Host-zu-Host-Verbindung erzielt werden können. Bei den beiden Konfigurationen (Gateway- und Bastion-Setup) sind zwei Übertragungen notwendig — vom Sender zum Proxy-Server und vom Proxy-Server zum Empfänger. Jede der beiden Übertragungen führt über ein gleich langes (kurzes) Kabel im LAN und wird auch auf beiden Seiten der Verbindung jeweils bis auf die Anwendungsebene "hochgereicht". Da beide Übertragungen sequentiell stattfinden müssen,[133] können über die beiden Verbindungen, die über den Proxy-Server führen, maximal die Hälfte der Umläufe pro Sekunde erreicht werden. Der maximal erreichbare Wert läge dann bei 1.296 Umläufen pro Sekunde. Die über den Proxy-Server gemessenen 1.000 Umläufe pro Sekunde sind somit immer noch etwa 23% schlechter als dieser theoretisch erreichbare Wert von 1.296 Umläufen pro Sekunde.

4.4.2.3 Verbindungsaufbauzeit über "MGproxy"

Die Auswirkungen von Proxy-Servern auf die Verbindungsaufbauzeiten (s. Abschnitt 4.4.1.3) machen den Bedarf für eine Optimierung deutlich. "MGproxy" wird daher um eine zweite, auf geringe Verbindungsaufbauzeiten optimierte Variante ergänzt.

Bei den Verbindungsaufbauzeiten über die bisher beschriebene "MGproxy"-Variante ist nicht zu erwarten, daß ein besseres Ergebnis als beim "Socks"-Proxy-Server erreicht wird, da "MGproxy" nur einen Standard-Server implementiert. Die durch die Multiprozeß- oder "Multithreaded"-Varianten bei "Socks" möglichen Verbindungsaufbauzeiten können somit von dieser Version nicht erreicht

[133] Das Meßwerkzeug beginnt erst dann mit dem Senden der nächsten Dateneinheit, wenn ein Byte vom Empfänger zurückgesendet wurde. (s. Abb. 4.20)

werden. Der beim Bastion-Setup über "MGproxy" gemessene Wert von 60,38 $\pm 0,11$ Transaktionen pro Sekunde muß daher mit den knapp 35 Transaktionen pro Sekunde des "Socks"-Proxy-Servers in der Konfiguration als Standard-Server verglichen werden. Durch den geringen Speicherbedarf des Prozesses und die einfachere Implementation des "Socks"-Protokolls[134] können die Verbindungsaufbauzeiten bei der Standard-Version von "MGproxy" gegenüber "Socks" annähernd halbiert werden. Die hieraus weiterentwickelte "Multithreaded"-Variante von "MGproxy" erreicht je nach Grad des "Loggings"[135] bis zu 190,90 $\pm 0,14$ Transaktionen pro Sekunde.

Die Ergebnisse der Transaktionsmessungen über "MGproxy" sind in Tabelle 4.11 zusammengefaßt. Die letzte Spalte der Tabelle mit den Zeiten für den Verbindungsauf- und -abbau berechnet sich aus der Transaktionslänge abzüglich der Zeit für einen Umlauf. Für die Berechnung der Dauer eines Umlaufs werden die Ergebnisse der Umlaufmessungen (s. Abschnitt 4.4.2.2) herangezogen.

Transaktionsmessung (180 Sek.) über:	Transaktionen [Tr/s]	Transaktionslänge [s]	Verbindungsaufbauzeit [s]
MGproxy Standard			
Bastion abgeschaltet Syslogd	60,38 $\pm 0,11$	$1,66 * 10^{-2}$	$1,56 * 10^{-2}$
Gateway abgeschaltet Syslogd	69,29 $\pm 0,22$	$1,44 * 10^{-2}$	$1,34 * 10^{-2}$
Gateway mit Syslog	66,15 $\pm 0,55$	$1,51 * 10^{-2}$	$1,41 * 10^{-2}$
MGproxy Multithreaded			
Gateway mit Syslog	155,79 $\pm 0,26$	$6,42 * 10^{-3}$	$5,42 * 10^{-3}$
Gateway abgeschaltet Syslogd	163,85 $\pm 0,27$	$6,10 * 10^{-3}$	$5,10 * 10^{-3}$
Gateway kein Syslog	190,90 $\pm 0,14$	$5,24 * 10^{-3}$	$4,24 * 10^{-3}$
Host-zu-Host	517,41 $\pm 1,45$	$1,93 * 10^{-3}$	$1,55 * 10^{-3}$

Tabelle 4.11: Transaktionen über "MGproxy"

Die Transaktionsmessung über "MGproxy" bei einem Gateway-Setup erzielt mit 69,29 Transaktionen pro Sekunde die geringsten Verbindungsaufbauzeiten der Standard-Version von "MGproxy". Warum beim Gateway-Setup ein 15% besseres Ergebnis erreicht wird als beim Bastion-Setup, ist nicht erklärbar. Entsprechend des geringen Unterschieds bei den Umlaufmessungen, wäre bei den Transaktionsmessungen eine ebenso geringe Differenz von unter 1% zu erwarten gewesen. Da nur relativ geringe Datenmengen gesendet werden, kann das Erreichen des maximalen Durchsatzes der ATM-Schnittstelle bei der Bastion nicht die

[134] "MGproxy" implementiert ausschließlich die wichtigsten Funktionen des Protokolls. Dies ist der "`CONNECT`"-Request mit dem Authentisierungsverfahren "`AUTH_NONE`" (d. h. ohne Authentisierung), sowie die Signalisierung aller Fehler, die auftreten können.

[135] Siehe Diskussion am Ende dieses Abschnitts.

Ursache für die längeren Verbindungsaufbauzeiten sein. Auch Kollisionen, die häufig für Performanzverluste in einem Ethernet verantwortlich sind, kommen bei dem ATM-Testnetz nicht in Frage.

Die Ursache für diesen Performanzverlust muß also mit Begrenzungen der Protokollrealisierung auf der Bastion zusammenhängen. Möglicherweise werden Ressourcen aufgebraucht, die für jede Schnittstelle zur Verfügung stehen. Die genauere Analyse dieses Effektes erfordert eine Parametrisierung des Protokollstapels, was jedoch mit erheblichem Aufwand verbunden ist. Warum zwischen Bastion- und Gateway-Setup ein Unterschied von 15% liegt, kann daher bisher noch nicht abschließend erklärt werden.

In der Tabelle 4.11 sind auch die Folgen des "Loggings" erkennbar. Da bei eingeschalteten "Logging" für jede Verbindung vom Standard-Server drei Zeilen und von der Multithreaded-Variante eine Zeile protokolliert werden, führt beim ständige Auf- und Abbauen von Verbindungen das Protokollieren zu einer zusätzlichen Belastung, die eine Verringerung des Transaktionsdurchsatzes verursacht. Die Multithreaded-Variante von "MGproxy" erreicht 155,79 Transaktionen pro Sekunde, wenn für jede Verbindung eine Zeile Logdaten aufgezeichnet werden. Bei dieser Last werden über 500.000 Zeilen bzw. über 40 MB Logdaten pro Stunde aufgezeichnet. Wird die Aufzeichnung beim Syslogd abgeschaltet, so entfällt der Aufwand für die Speicherung der Daten auf Platte und der Durchsatz steigt auf 163,85 Transaktionen pro Sekunde. Eine weitere Steigerung auf 190,90 Transaktionen pro Sekunde läßt sich erreichen, wenn bereits die Funktionen zur Erzeugung der Logdaten im Proxy-Server deaktiviert werden. Während das Protokollieren in der Testumgebung abgeschaltet werden kann, ist die Aufzeichnung von sicherheitsrelevanten Ereignissen in realen Firewall-Umgebungen eine zwingende Voraussetzung für den sicheren Betrieb. Die Performanzverluste durch das Protokollieren können daher nicht gänzlich vermieden werden. Um so wichtiger ist es, durch die Wahl effizienter "Logging"-Verfahren und durch frühzeitige Datenreduktion den "Overhead" so gering wie möglich zu halten.[136]

4.4.2.4 Anwendungsperformanzmessungen mit "MGproxy"

Zur Ermittlung der Anwendungsperformanz des Proxy-Servers "MGproxy" werden die beiden "Ultra-2"-Systeme und ein "Ultra-1"-System in einem Bastion-Setup konfiguriert. Dies bedeutet, daß sich alle drei Systeme im selben IP-Subnetz befinden und daher keine Auswirkungen durch zusätzliche Packet Screens oder Router auftreten. Die "Ultra-2"-Systeme werden mit dem im Abschnitt 4.2.5

[136] Handelt es sich hier um die Protokollierung der berechtigten Benutzung des Proxy-Servers, so ist die effiziente Verarbeitung insbesondere auch während eines Angriffs wichtig, da sonst über das "Logging"-System "Denial of Service"-Angriffe durchgeführt werden können.

beschriebenen Meßwerkzeug als Quelle und Senke für HTTP-Anfragen eingesetzt. Die dort gemessenen Ergebnisse für die direkte Host-zu-Host-Kommunikation werden in diesem Abschnitt als Referenzwerte zum Vergleich der Performanz des Proxy-Servers herangezogen. Auf dem "Ultra-1"-System ist der Proxy-Server "MGproxy" in der "Multithreaded"-Variante installiert. Als Kopierlänge wird der in Abschnitt 4.4.2.1 als optimal ermittelte Wert von 64 KByte verwendet.

Leider kann der Zugriff des Clients auf den Proxy-Server nicht wie bei den Messungen mit "Netperf" über das als "Shared-Library" implementierte "Socks"-Protokoll erfolgen, da diese "Shared-Library" mit "multithreaded" Clients nicht korrekt funktioniert.[137] Dieses Problem kann durch die Implementation einer "multithread"-fähigen "Shared-Library" oder durch die Ergänzung der sogenannten "Plug-Gateway"-Funktionalität beim Proxy-Server gelöst werden. Ein "Plug-Gateway"-Proxy-Server leitet Verbindungen immer zu einem vorher fest konfigurierten Port weiter. Der Client hat keine Möglichkeit, andere Ports oder andere Hosts über den Proxy anzusprechen. Da bei der "Plug-Gateway"-Lösung der Client dem Proxy die gewünschte Zieladresse nicht mitzuteilen braucht, kann auf das "Socks"-Protokoll verzichtet werden. Weil stattdessen nur die benötigte IP-Adresse und Portnummer beim Start des Proxy-Servers als Parameter übergeben werden müssen, ist diese Lösung einfacher zu implementieren als eine "multithread"-fähige "Shared-Library" für das "Socks"-Protokoll. Bei den folgenden Messungen in diesem Abschnitt wird "MGproxy" in einer "Plug-Gateway"-Konfiguration betrieben.

4.4.2.5 Maximale Anzahl paralleler Verbindungen

Bei direkter Host-zu-Host-Kommunikation konnte das Meßwerkzeug bis zu 800 parallele HTTP-Verbindungen gleichzeitig betreiben (s. Abschnitt 4.2.6). Demgegenüber wird bei der Kommunikation über den Proxy-Server "MGproxy" bereits bei 250 Verbindungen die maximale Anzahl paralleler Verbindungen erreicht (s. Abb. 4.65).[138]

Wie auch bei der Messung bei direkter Host-zu-Host-Kommunikation wird bei Konfigurationen mit 1 bis 30 Threads ein Datendurchsatz erreicht, der nur knapp unterhalb des möglichen Schnittstellendurchsatzes liegt. Erst bei Messungen mit mehr als 40 Threads fällt der Durchsatz unter 130 MBit/s; gleichzeitig steigt

[137] Es ist hingegen unerheblich, ob der Proxy-Server "multithreaded" implementiert ist. Die Unverträglichkeit tritt nur bei "multithreaded" Clients auf.

[138] Bei der diesem Abschnitt zugrundeliegenden Messung wurde die Socket-Größe von 64 KByte noch über "`setsockopt`"-Aufrufe eingestellt. Der beim Transaktionsdurchsatz effizientere Ansatz, die Socket-Größe systemweit auf den gewünschten ("Default") Wert einzustellen, bringt bei den sehr langen Transaktionen dieser Messung keinen meßbaren Vorteil. Auf eine Wiederholung dieser Messung wurde daher verzichtet.

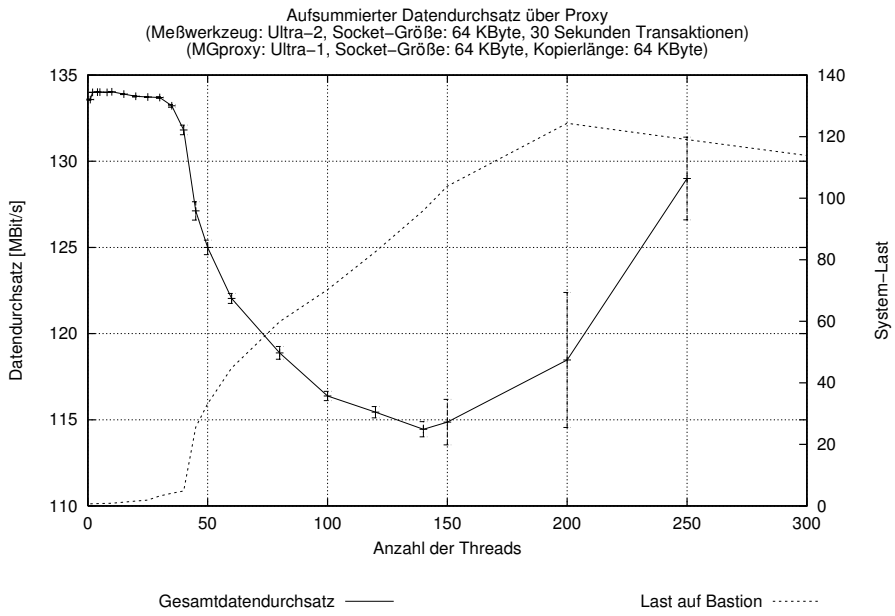


Abbildung 4.65: Gesamtdatendurchsatz über "MGproxy"

die CPU-Last[139] auf der Bastion rapide an. Mit einer CPU-Last von über 100 gleichzeitig die CPU anfordernden Threads bzw. Prozessen werden bei dieser Messung Werte erreicht, die die bei gebräuchlichen Servern nur kurzzeitig erreichten Spitzenwerte um mehr als das 10-fache überschreiten. Eine CPU-Last von über 100 wird auch bei den mit Fehlern abgebrochenen Messungen oberhalb von 250 Threads erreicht. Demgegenüber bleibt die CPU-Last auf den Meßsystemen während der gesamten Messung unter einem Wert von 0,3; was bedeutet, daß die CPU nur etwa zu einem Drittel ausgelastet ist.

Die Ursache für den Datendurchsatzrückgang im Bereich oberhalb von 30 Threads ist auf die Wirkung des Proxy-Servers zurückzuführen. Dies wird deutlich aus dem Vergleich der Messung über den Proxy-Server (Abb. 4.65) mit der Messung bei direkter Host-zu-Host-Kommunikation (Abb. 4.28). Zudem fällt der Beginn des Datendurchsatzrückganges zusammen mit dem Anstieg der CPU-Last auf der Bastion, während die CPU-Last auf den beiden Meßsystemen gering bleibt. Bei einer Kopierlänge von 64 KByte sowie einer Sende- und Empfangspuffergröße von ebenfalls 64 KByte[140] entsteht für jede über den Proxy aufgebaute Verbindung ein

[139] Die CPU-Last gibt die über einen Zeitraum (hier 15 Minuten) gemittelte Anzahl von Prozessen bzw. Threads an, die gleichzeitig die CPU beanspruchen bzw. beanspruchen könnten. Eine CPU-Last von 1 gibt an, daß über den Meßzeitraum ständig ein Prozeß in Bearbeitung war.

[140] Sende- und Empfangspuffer müssen vom Proxy jeweils für die Verbindung zum Client und die Verbindung zum Server bereitgestellt werden. Es werden somit vier Puffer benötigt.

Speicherbedarf von 320 KByte. Der vor einer Messung im Leerlauf zur Verfügung stehende freie Speicher beträgt bei dem als Bastion eingesetzten "Ultra-1"-System etwa 9 bis 10 MByte. Bei einem Speicherbedarf von 320 KByte für jede parallele Verbindung reicht der Speicher für etwa 30 parallele Verbindungen aus, bevor Teile des Speichers auf eine Platte ausgelagert werden müssen und dies zu der beobachteten starken Zunahme der CPU-Last führt.

Nachdem der Datendurchsatz oberhalb von 30 Threads mit steigender Anzahl von Threads absinkt, steigt er ab 140 Threads wieder an (s. Abb. 4.65). Der Anstieg wurde bereits bei der direkten Host-zu-Host-Kommunikation beobachtet (s. Abb. 4.28). Im Abschnitt 4.2.6 wurde bereits darauf hingewiesen, daß die Synchronisationsmechanismen des Meßwerkzeuges bei Messungen mit mehr als 100 Threads nicht mehr ausreichen, um den Gesamtdatendurchsatz korrekt zu bestimmen.[141] In diesem Bereich oberhalb von 140 Threads kann auch nicht mehr von der Annahme ausgegangen werden, daß die Anzahl der parallelen Verbindungen während der gesamten Messung der Anzahl der Threads entspricht, da viele Verbindungen wegen längerer Inaktivität von der Überwachungsfunktion des Meßwerkzeuges abgebrochen werden.[142] Dieses Abbrechen von Verbindungen stellt sicher, daß die Messungen in akzeptabler Zeit abgeschlossen werden können, darüber hinaus wird durch das Abbrechen das Verhalten von Benutzern emuliert, die in der Praxis relativ schnell Verbindungen abbrechen, wenn sie keinen Fortschritt mehr beobachten.

Abbildung 4.67 zeigt die Kurven für den Einzel- und Gesamttransaktionsdurchsatz im Vergleich zwischen direkter Host-zu-Host-Kommunikation und für Verbindungen über "MGproxy". Der Verlauf der Kurven ist für Messungen mit bis zu 80 Threads identisch. Danach kann über den Proxy-Server ein höherer Transaktionsdurchsatz erzielt werden als bei direkter Host-zu-Host-Kommunikation. Dieses überraschende Ergebnis wird erklärbar, wenn die Datendurchsatzkurve betrachtet wird. Da der Datendurchsatz durch den Proxy-Server begrenzt wird, sind auf den Meßsystemen weniger Ressourcen notwendig und die CPU-Last auf den Meßsystemen bleibt während der gesamten Messung unterhalb von 0,3. Bei der geringen CPU-Last auf den Meßsystemen kann die geforderte Transaktionslänge von 30 Sekunden genauer eingehalten werden als bei der direkten Host-zu-Host-Kommunikation. Die Anzahl der Transaktionen, die in der Meßphase durch-

[141] Durch die geringere CPU-Last auf den Meßsystemen bei dieser Messung reichen die Synchronisationsmechanismen auch für eine größere Anzahl von Threads aus als bei direkter Host-zu-Host-Kommunikation. Die genaue Anzahl von Threads, bis zu der der gemessene Datendurchsatz noch nicht durch mangelnde Synchronisation verfälscht wird, kann an dieser Stelle leider nicht exakt ermittelt werden. Die durchschnittliche Anzahl der Transaktionen, die während der Meßphase durchgeführt werden mußten (s. Abb. 4.66), sprechen dafür, daß die Synchronisationsmechanismen bei dieser Messung für den gesamten Meßbereich ausreichen.

[142] Das Abbrechen von Verbindungen tritt erstmals bei der Messung mit 140 Threads auf.

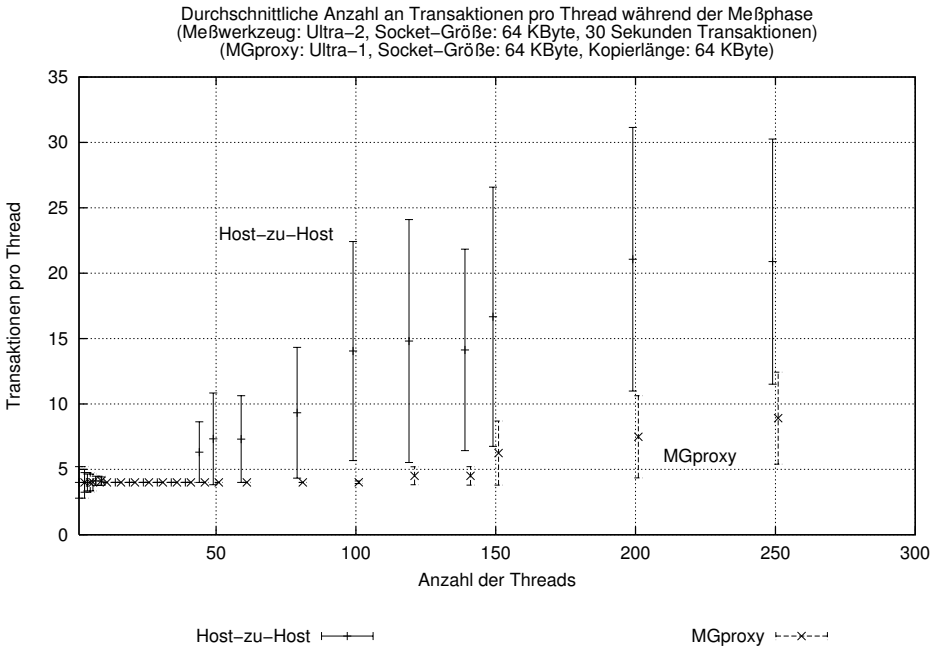


Abbildung 4.66: Anzahl von Transaktionen während der Meßphase

geführt werden müssen, — eine Maßzahl für die gleichmäßige Verteilung der Betriebsmittel auf die einzelnen Threads des Meßwerkzeuges — bleibt bei der Messung über den Proxy-Server aus diesem Grunde auch geringer (s. Abb. 4.66). Erst bei 250 Threads ist der Transaktionsdurchsatz über "MGproxy" geringer, da Verbindungen von der Überwachungsfunktion aufgrund längerer Inaktivität vorzeitig abgebrochen werden. Diese abgebrochenen Transaktionen tragen nicht zum Transaktionsdurchsatz bei.

Implementationsdetails: Bei der ersten Messung wurde bereits ein wichtiger Fehler im Proxy-Server entdeckt. Die maximale Anzahl der Datei-Deskriptoren, die ein Prozeß verwenden darf, wird durch das Betriebssystem begrenzt. Die voreingestellte Grenze ("Ressource-Limit") ist so niedrig, daß nur maximal 30 Verbindungen über den Proxy geführt werden können. Um mehr Datei-Deskriptoren verwenden zu können, muß vom Prozeß ein höheres "Ressource-Limit" angefordert werden. Nach dem Heraufsetzen des "Ressource-Limits" darf der Prozeß bis zu 1.024 Datei-Deskriptoren verwenden, was prinzipiell für über 500 Threads ausreicht.[143]

[143] Jeder Thread des Proxies benötigt zwei Datei-Deskriptoren – einen für die Verbindung zum Client und einen weiteren für die Verbindung zum Server.

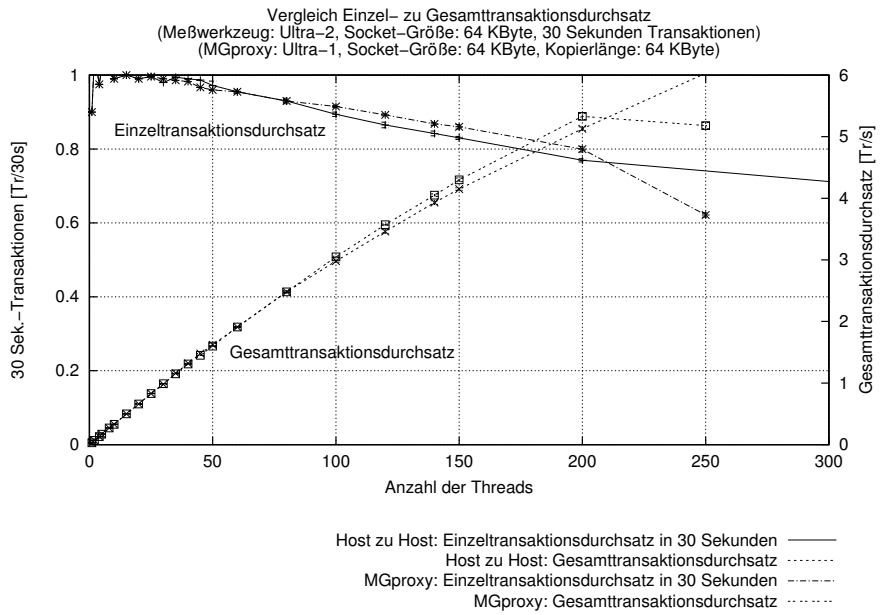


Abbildung 4.67: Einzel- und Gesamttransaktionsdurchsatz über "MGproxy"

Fazit: Der Proxy-Server kann maximal 250 parallele Verbindungen verarbeiten. Bei mehr als 30 Verbindungen, die gleichzeitig aktiv genutzt werden, begrenzt der Proxy-Server zudem den Gesamtdatendurchsatz. Gleichzeitig steigt oberhalb von 30 Threads die CPU-Last stark an. Bei zusätzlicher Belastung des Proxy-Servers durch aufwendige Kontrollmechanismen sind noch sehr viel schlechtere Ergebnisse zu erwarten (s. Abschnitt 4.4.2.8)

4.4.2.6 Transaktionsdurchsatz bei parallelen Transaktionen

Die Verlängerung der Verbindungsaufbauzeiten durch einen Proxy-Server (s. Abschnitt 4.4.2.3) wirkt sich besonders stark auf Protokolle wie HTTP aus, bei denen kurze Transaktionen überwiegen. Messungen bei direkter Host-zu-Host-Kommunikation ergeben einen Maximalwert von 1.796 Transaktionen pro Sekunde (Tr/s) bei 12 parallelen Threads (s. Abschnitt 4.2.7). Über den Proxy-Server "MGproxy" werden im Vergleich dazu nur 392 ± 3 Transaktionen pro Sekunde erreicht (s. Abb. 4.68). Dieser Durchsatz wird bereits bei 2 parallelen Threads des Meßwerkzeuges erzielt.[144]

[144] Dabei muß jedoch berücksichtigt werden, daß der Proxy-Server auf einer "Ultra-1" eingesetzt wird. Messungen zwischen zwei "Ultra-1"-Systemen ergaben eine Transaktionsdurchsatz mit 625 Tr/s bei 4 bis 6 Threads (s. Abschnitt 4.2.7). Selbst im Vergleich zu dem Transaktionsdurchsatz von 625 Tr/s führt der Einsatz des Proxy-Servers zu einer Reduzierung um 37%.

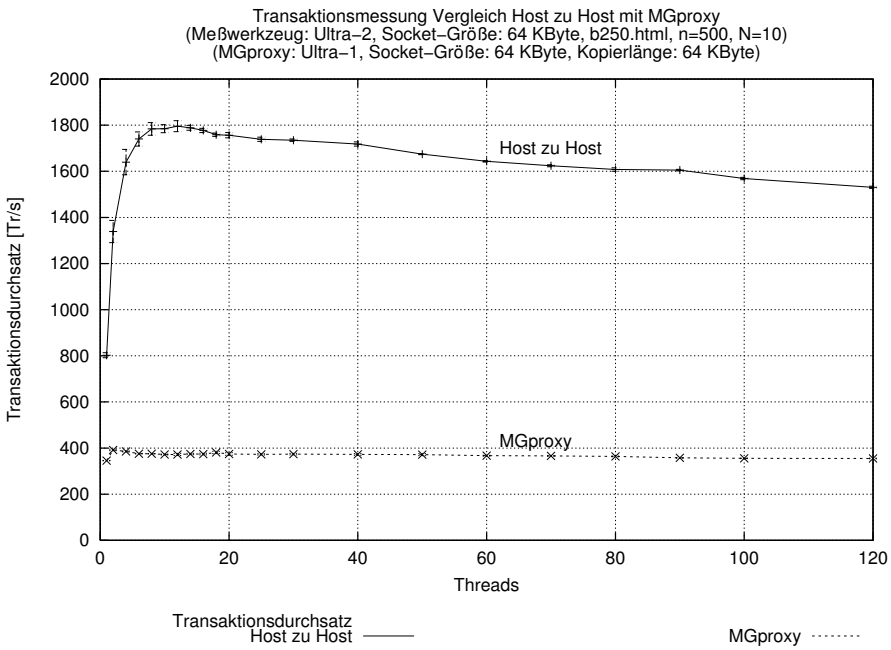


Abbildung 4.68: Transaktionsdurchsatz über "MGproxy"

Ressourcen des Netzes und auch Ressourcen auf der Bastion bleiben bei der seriellen Transaktionsmessung (s. Abschnitt 4.4.2.3) während der Wartezeit beim Verbindungsaufbau ungenutzt. Bei der direkten Host-zu-Host-Kommunikation können diese Ressourcen durch parallel laufende Threads ausgenutzt werden, so daß sich der Transaktionsdurchsatz von 803 Transaktionen pro Sekunde bei einem Thread auf 1.796 Transaktionen pro Sekunde bei 12 Threads mehr als verdoppeln kann. Bei der Transaktionsmessung über den Proxy sind diese Steigerungen nicht möglich. Hier wird bei einem Thread des Meßwerkzeuges ein Transaktionsdurchsatz von 346 $\pm$ 0,2 Transaktionen pro Sekunde erreicht. Wenn 2 Threads eingesetzt werden, steigert sich der Gesamttransaktionsdurchsatz nur geringfügig auf 392 Transaktionen pro Sekunde. Der Einsatz weiterer paralleler Threads zur Messung bringt statt einer weiteren Steigerung einen leichten Abfall des Transaktionsdurchsatzes.

Die 392 Tr/s zeigen im Vergleich zu den bei der direkten Host-zu-Host-Kommunikation erreichten 1.796 Tr/s, daß der Proxy-Server bei der Transaktionsmessung den Engpaß darstellt. Die wichtigste Ursache für diesen Engpaß ist die Verbindungsannahmen durch den Hauptthread. Bei kurzen Transaktionen muß der Hauptthread den größten Teil der Bearbeitung im Proxy-Server durchführen. Der Hauptthread nimmt eine neue Verbindung zunächst an und muß dann einen Thread suchen, der die weitere Bearbeitung der Anfrage übernimmt. Da der ausgewählte Thread aber nur noch eine Verbindung zum Server aufbauen und eine

kurze Transaktion durchführen muß, steht der Thread schnell wieder für weitere Transaktionen zur Verfügung. Die Aufgabe des Hauptthreads kann jedoch nicht auf mehrere Threads verteilt werden, so daß dieser Hauptthread bei Messungen mit kurzen Transaktionen zum Engpaß wird.

Beobachtet man die Anzahl der offenen Verbindungen zwischen Client und Proxy sowie zwischen Proxy und Server, zeigt sich, daß während einer Messung unabhängig von der Anzahl der vom Meßwerkzeug verwendeten Threads immer nur 2 bis 4 Verbindungen zwischen Proxy und Server gleichzeitig geöffnet sind. Auf der Seite zum Client hin hat der Proxy immer etwa 10 Verbindungen gleichzeitig geöffnet. Bei der gewählten kurzen Transaktionslänge von nur 250 Byte ist es verständlich, daß keine große Menge offener Verbindungen erreicht wird, da die Transaktionen schneller abgeschlossen werden als das neue Verbindungen eröffnet werden.

Die größere Anzahl offener Verbindungen auf Client-Seite ergibt sich, weil die Verbindungen zwischen Client und Proxy viel länger geöffnet bleiben als die Verbindungen zwischen Proxy und Server (siehe Abbildung 4.60, Seite 319).

Unterschiede zur "Netperf" Verbindungsaufbauzeitmessung: Eine Messung mit einem Thread sollte ähnliche Ergebnisse liefern wie die Verbindungsaufbauzeitmessung mit "Netperf". Trotz längerer Transaktionen wird bei der HTTP-Messung mit 346 Tr/s ein um etwa 82% höheres Ergebnis erreicht als bei der "Netperf"-Messung mit 190 Tr/s. Der höhere Transaktionsdurchsatz ist auf das Entfallen des "Socks"-Protokolls bei der HTTP-Messung zurückzuführen. In der Konfiguration als "Plug-Gateway" braucht der Client dem Proxy-Server nicht zu Beginn jeder Verbindung die gewünschte Zieladresse mitzuteilen, da die Verbindung immer nur zur selben Adresse hin geöffnet werden kann. Beim "Socks"-Protokoll sind mindestens zwei Umläufe zwischen Client und Proxy notwendig, bevor der Proxy-Server die Adresse des gewünschten Zielrechners erhalten hat. Bei Transaktionen, die bereits mit einem Umlauf zwischen Client und Server abgeschlossen sind, machen sich die zusätzlichen Umläufe des "Socks"-Protokolls stark bemerkbar.

Implementationsalternative: Dezentrale Verbindungsannahme: Da der Hauptthread beim Transaktionsdurchsatz mit kurzen Transaktionen zum Engpaß wird, soll hier eine alternative Realisation vorgestellt werden. Bei dem Proxy-Server, der bei den Messungen (Abbildung 4.68) verwendet wird, führt der Hauptthread die Annahme von neuen Verbindungen und die Auswahl eines freien Arbeitsthreads aus. Die weitere Bearbeitung der Verbindung des Clients wird dann vom Arbeitsthread übernommen. Da diese zentrale Verbindungsannahme bei der Transaktionsmessung zum Engpaß wird, wäre eine dezentrale An-

nahme von neuen Verbindungen vorzuziehen. Dazu muß jeder Arbeitsthread selbst eine neue Verbindung annehmen können. Der Systemaufruf zum Annehmen einer Verbindung ("`accept`") muß dafür mit einer Semaphore abgesichert werden, so daß zur Zeit immer nur ein Arbeitsthread eine neue Verbindung annehmen kann. Nach dem Annehmen einer Verbindung gibt der Arbeitsthread die Semaphore frei und beginnt mit der Bearbeitung der Client-Anfrage. Die freigegebene Semaphore wird von einem anderen Arbeitsthread sofort aufgenommen und ermöglicht diesem, auf neue Verbindungen zu warten (d.h. einen "`accept`"-Aufruf durchzuführen). Alle weiteren Arbeitsthreads warten, sofern sie nicht gerade Client-Anfragen bearbeiten, auf eine erneute Freigabe der Semaphore.

Das beschriebene Konzept der dezentralen Verbindungsannahme wird als eine zusätzliche Variante von "MGproxy" implementiert. Die erzielte Steigerung des Transaktionsdurchsatzes ist in Abbildung 4.69 wiedergegeben. Während bei der zentralen Verbindungsannahme der Transaktionsdurchsatz oberhalb von 2 Threads bereits wieder abfällt, sind bei der dezentralen Verbindungsannahme noch leichte Steigerungen möglich. Der maximale Transaktionsdurchsatz von 409 ± 5 Tr/s wird hier mit 14 Threads im gleichen Bereich erreicht, bei dem auch auf der direkten Host-zu-Host-Messung das Maximum gemessen wurde — der erzielte Wert von 409 Tr/s liegt jedoch noch weit entfernt von den 1.796 Tr/s bei direkter Host-zu-Host-Kommunikation.

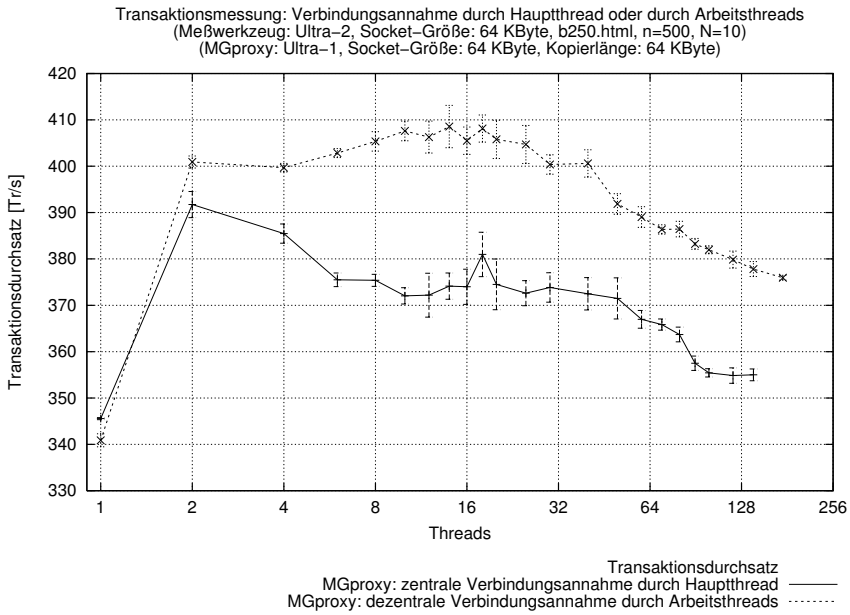


Abbildung 4.69: Transaktionsdurchsatz: Verbindungsannahme-Alternativen

Durch das Konzept der dezentralen Verbindungsannahme wird zwar verhindert, daß der Hauptthread durch seine Verteilaufgabe zum Engpaß wird, jedoch ist die-

se Lösung in der Praxis nicht einsetzbar. Gibt ein Arbeitsthread nach der Verbindungsannahme die Semaphore frei, so aktiviert das Betriebssystem alle Threads, die an der Semaphore warten. Durch diese Aktivierung steigt abhängig von der Anzahl der wartenden Threads die CPU-Last auf dem System (s. Abb. 4.70). Bei einer geringen Anzahl von Threads (10-15)[145] bleibt der Lastanstieg immer noch in einem akzeptablen Rahmen. Darüber steigt die CPU-Last auf der Bastion jedoch schnell auf Werte über 30. Da die Arbeitsthreads aufgrund der kurzen Transaktionen kaum durch die Bearbeitung von Verbindungen der Clients zur CPU-Last beitragen und diese Bearbeitung aufgrund des gleichbleibenden, gemessenen Transaktionsdurchsatzes unabhängig von der Anzahl der Threads etwa gleichviel CPU-Last erzeugt, ist dieser Lastanstieg ausschließlich auf die Synchronisation der Verbindungsannahme zurückzuführen. Wenn vom Proxy größere Datenmengen transportiert werden müssen oder z. B. eine Verschlüsselung der übertragenen Daten durchgeführt werden soll, führt die synchronisationsbedingte CPU-Last zu einer Verschlechterung der Performanz im Vergleich zu einem Proxy mit zentraler Verbindungsannahme.

Da Proxy-Server auch eine größere Anzahl paralleler Verbindungen unterstützen müssen, ist das vorgestellte Konzept mit dezentraler Verbindungsannahme für den Einsatz bei Proxy-Servern nicht geeignet. Die zentrale Verbindungsannahmen wird zwar bei Transaktionsmessungen zum Engpaß, führt aber unabhängig von der Anzahl der Threads zu einer gleichbleibenden CPU-Last. Die dezentrale Verbindungsannahme erreicht bei einer sehr viel höheren CPU-Last nur einen geringfügig höheren Transaktionsdurchsatz. Beim gleichzeitigen Einsatz CPU-intensiver Verschlüsselung der übertragenen Daten muß das Verfahren zur Annahme von Verbindungen zudem um die CPU konkurrieren.

Auswirkungen von Implementationsentscheidungen: Setzen der Socket-Größe: Im Abschnitt 4.2.7 wurde bereits auf eine Verringerung des Transaktionsdurchsatzes beim Meßwerkzeug hingewiesen, der durch das Setzen der Socket-Größe über "`setsockopt`"-Funktionen bedingt war (siehe Seite 271). Nach der systemweiten Einstellung der "Default"-Socket-Größe kann auf das individuelle Setzen der Socket-Größe verzichtet werden.

Beim Proxy-Server sind dieselben Auswirkungen auf den Transaktionsdurchsatz zu beobachten. Bei den in diesem Abschnitt vorgestellten Messungen wird die Socket-Größe sowohl auf den Meßsystemen als auch auf der Bastion systemweit

[145] Die Anzahl der Threads ist für jeden Meßpunkt beim Proxy-Server um 5 Threads höher als beim Meßwerkzeug. Da die Anzahl der benötigten Threads bei jedem Meßpunkt bereits vor Meßbeginn feststeht, kann die Anzahl der Threads beim Start des Proxies angegeben werden. In einer realen Firewall-Umgebung variiert die Anzahl der benötigten Threads ständig. Hier ist ein Verfahren notwendig, daß durch Erzeugen bzw. Beenden von Threads die Anzahl der Threads ständig anpaßt.

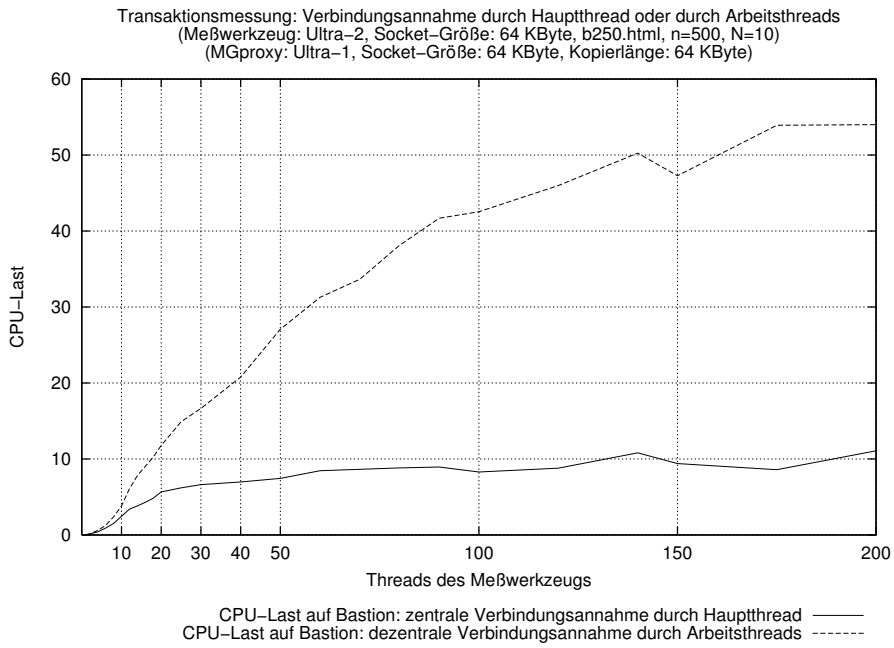


Abbildung 4.70: Auswirkungen der Verbindungsannahme-Alternativen auf CPU-Last der Bastion

eingestellt. In einer Vergleichsmessung, bei der der Proxy-Server die Socket-Größe für jede Verbindung über die "`setsockopt`"-Funktion setzen muß, werden maximal 353 $\pm$ 1 Tr/s (statt 392 Tr/s) erreicht. Mit nur einem Thread beträgt der Transaktionsdurchsatz 320 $\pm$ 1 Tr/s während bei systemweiter Einstellung der Socket-Größe 346 Tr/s erreicht werden.

Fazit: Die parallele Transaktionsmessung zeigt deutlich den Engpaß beim Proxy-Server. Mit 392 Tr/s werden nur etwa 22% des bei direkter Host-zu-Host-Kommunikation möglichen Transaktionsdurchsatzes erreicht. Durch genauere Analysen kann die vom Hauptthread durchgeführte Annahme von Verbindungen und Auswahl des Arbeitsthreads als Engpaß identifiziert werden. Die dezentrale Verbindungsannahme durch die Arbeitsthreads bringt zwar einen etwa höheren Transaktionsdurchsatz, jedoch führt diese Implementationsalternative zu einer sehr hohen CPU-Last, die den Einsatz in realistischen Anwendungsbereichen verhindert.

Die Folgen des Engpasses beim Transaktionsdurchsatz zeigen sich besonders deutlich im nächsten Abschnitt bei den Messungen mit längeren Transaktionen. Zur Erhöhung des Transaktionsdurchsatzes muß versucht werden, durch Parallelisierungskonzepte die Last auf mehrere Bastionen zu verteilen.

4.4.2.7 Transaktionsmessung mit längeren Transaktionen

Typische Transaktionen über einen Proxy-Server sind länger als die 250 Byte Transaktionen, die zur Bestimmung des maximalen Transaktionsdurchsatzes verwendet werden. Um Aussagen über das Verhalten von Proxy-Servern bei typischen Transaktionslängen zu ermöglichen, wurde in Abschnitt 4.2.8 eine Messung mit längeren Transaktionen vorgestellt. Auch bei der Messung über den Proxy-Server sollen sowohl Datendurchsatz als auch Transaktionsdurchsatz in Abhängigkeit von der Anzahl der Threads und der Transaktionslänge ermittelt werden.

Abbildung 4.71 zeigt den Datendurchsatz über den Proxy in Abhängigkeit von der Transaktionslänge für 1, 2 und 10 Threads. Zwischen den Kurven für 1 und 2 Threads fällt der Unterschied noch deutlich aus. Demgegenüber kann mit 10 Threads nur noch ein geringfügig höherer Datendurchsatz erreicht werden. Nicht abgebildet ist die Kurve für 4 Threads, bei der praktisch der gleiche Datendurchsatz wie bei der Messung mit 10 Threads erreicht wird. Eine Ausnahme bildet hierbei der bereits bei der Host-zu-Host-Messung beobachtete Einbruch; dieser Effekt ist bei Messungen mit 4 und 5 Threads noch sehr deutlich erkennbar und fällt erst bei 10 Threads kaum noch auf.

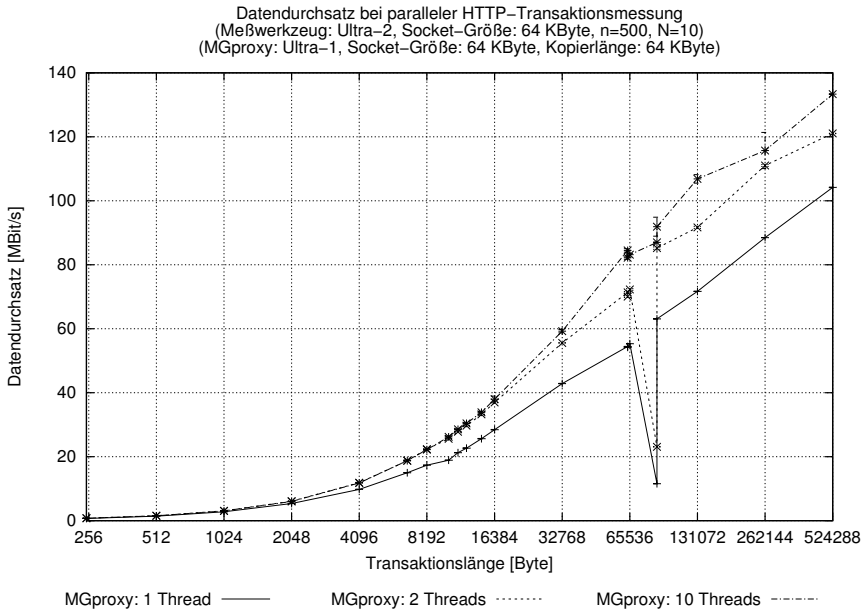


Abbildung 4.71: Gesamtdatendurchsatz über "MGproxy" bei längeren Transaktionen

Im Vergleich der Meßkurven mit einem Thread über den Proxy mit der Kurve bei direkter Host-zu-Host-Kommunikation (s. Abb. 4.72) zeigen sich die Folgen der

höheren Verbindungsaufbauzeiten über den Proxy. Aufgrund dieser längeren Verbindungsaufbauzeiten bleibt der Datendurchsatz unterhalb der Vergleichskurve bei direkter Host-zu-Host-Kommunikation. Bei 32 KByte langen Transaktionen ist der Durchsatz eines Threads über den Proxy mit etwa 40 MBit/s nur halb so hoch wie bei direkter Kommunikation. Selbst mit 512 KByte langen Transaktionen kann über den Proxy immer noch nicht der maximale Netzdurchsatz erreicht werden.

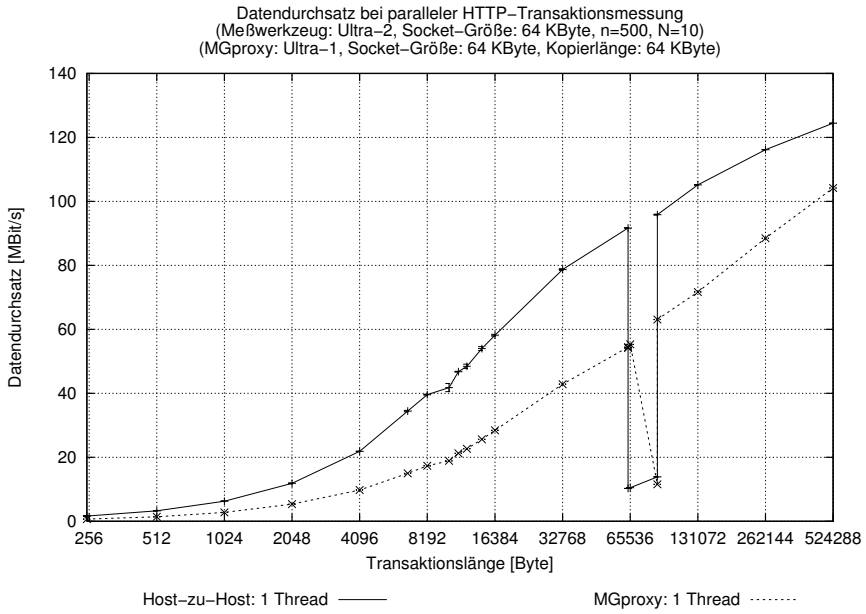


Abbildung 4.72: Vergleich Datendurchsatz über Host-zu-Host und "MGproxy"

Bei Messungen mit mehreren parallelen Threads wird der Unterschied zwischen direkter Kommunikation und Verbindungen über den Proxy noch deutlicher (s. Abb. 4.73). Mit 20 parallelen Threads kann zwar bei 512 KByte langen Transaktionen der maximale Netzdurchsatz auch über den Proxy erreicht werden, aber bei der direkten Host-zu-Host-Messung war dies bereits bei 11 KByte Transaktionen möglich.[146]

Betrachtet man den Datendurchsatz in Abhängigkeit von der Anzahl der parallelen Threads (s. Abb. 4.74), dann zeigt sich, daß der Datendurchsatz bei kleineren (bis 8 KByte) Transaktionen nicht über das Ergebnis einer Messung mit 2 Threads steigen kann. Im Vergleich dazu war auf der direkten Host-zu-Host-Messung bis 8 Threads eine stetige Durchsatzsteigerung möglich (s. Abb. 4.35). Bei längeren Transaktionen steigt der Datendurchsatz über den Proxy bis 4

[146] In der Abbildung 4.73 sind die Kurven für 20 Threads wiedergegeben. Der Einsatz einer größeren Anzahl paralleler Threads ermöglicht keinen höheren Datendurchsatz.

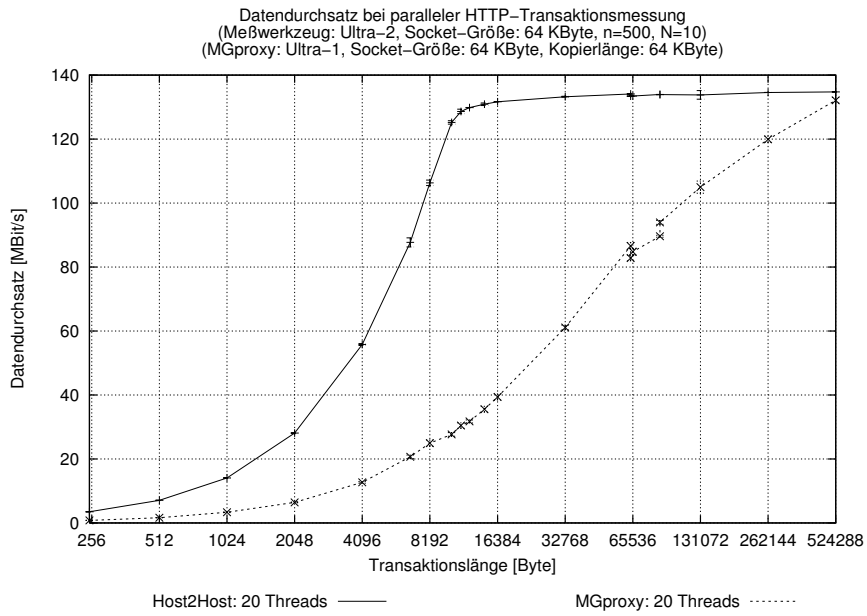


Abbildung 4.73: Vergleich Datendurchsatz über Host-zu-Host und "MGproxy"

Threads (64 KByte Transaktionen), 8 Threads (256 KByte Transaktionen) oder 10 Threads (512 KByte Transaktionen). Diese Steigerungen werden möglich, weil bei längeren Transaktionen der Verwaltungs-"Overhead" beim Verbindungsaufbau im Verhältnis zur Gesamttransaktionsdauer abnimmt. Bei kurzen Transaktionen bildet der Verwaltungs-"Overhead" beim Verbindungsaufbau dagegen einen Engpaß. Dabei sind bei den Kurven für die 256 und 512 KByte Transaktionen für Messungen mit mehr als 15 Threads bereits geringe Durchsatzrückgänge aufgrund des höheren Verwaltungs-"Overheads" der Threads erkennbar. Der Verwaltungs-"Overhead" äußert sich auch in der CPU-Last auf der Bastion, die oberhalb von 15 Threads steil ansteigt (s. Abb. 4.75).

Der Transaktionsdurchsatz in Abhängigkeit von der Transaktionslänge fällt bei Messungen über den Proxy nur sehr flach ab (s. Abb. 4.76). Dabei sind auch bei der Messung über den Proxy bei Transaktionslängen unterhalb von 8 KByte keine Auswirkungen der Transaktionslänge auf den Transaktionsdurchsatz zu erkennen, da die gesamte Nachricht innerhalb einer Dateneinheit (MTU 9.180 Byte) übertragen werden kann. Bei Transaktionslängen oberhalb von 8 KByte fällt der Transaktionsdurchsatz nicht so stark ab, da die erzielten Datendurchsätze im Gegensatz zur direkten Host-zu-Host-Messung jederzeit weit unterhalb des Netzdurchsatzes bleiben.

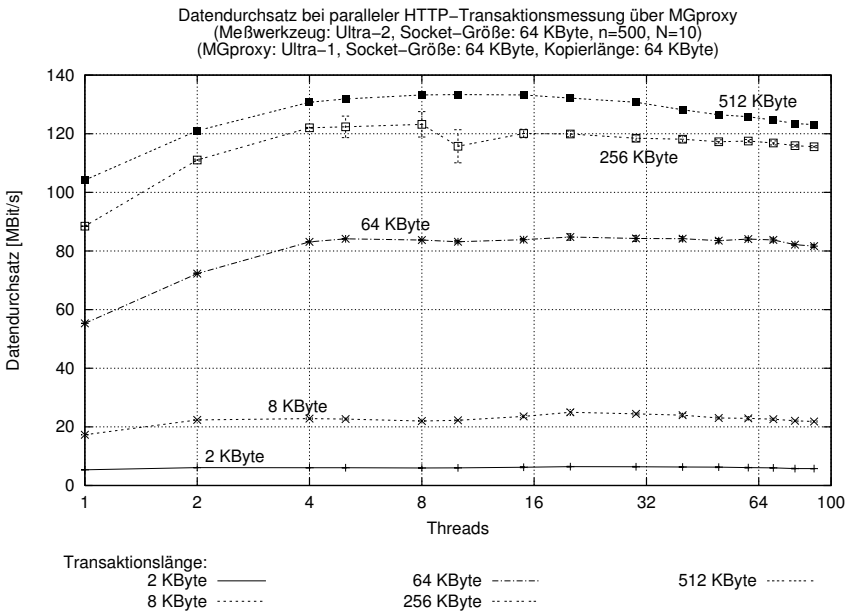


Abbildung 4.74: Gesamtdatendurchsatz über "MGproxy" bei längeren Transaktionen

Fazit: Die hohen Verbindungsaufbauzeiten über den Proxy ermöglichen erst bei sehr langen Transaktionen einen Datendurchsatz, der fast den Durchsatz der Netzwerkschnittstelle erreicht. Auch der Gesamtdatendurchsatz über mehrere parallele Verbindungen kann bei kurzen Transaktionen kaum höhere Werte erzielen, da der geringe Transaktionsdurchsatz des Proxies schnell erreicht ist und somit die theoretisch vorhandene höhere Parallelität nicht zur Steigerung des Gesamtdatendurchsatzes ausgenutzt werden kann. Um den bei der direkten Host-zu-Host-Messung mit 16 KByte langen Transaktionen gemessenen Gesamtdatendurchsatz von 131 MBit/s zu erreichen, ist zum Beispiel ein Transaktionsdurchsatz von etwa 1.000 Tr/s erforderlich. Über den Proxy reicht der Transaktionsdurchsatz von 300 Tr/s nur für etwa 38 MBit/s. Da Verringerungen der Verbindungsaufbauzeiten insbesondere im WAN kaum möglich sind und auch eine Optimierung des Datendurchsatzes kaum Gewinne verspricht, muß eine Steigerung des Transaktionsdurchsatzes des Proxies wichtigstes Optimierungsziel sein.

4.4.2.8 Performanzauswirkungen von Zugriffskontrolle und Verschlüsselung

Die bisher mit "MGproxy" durchgeführten Messungen beziehen sich auf die minimale Proxy-Funktionalität ohne Kontrollfunktionen. In der Praxis werden

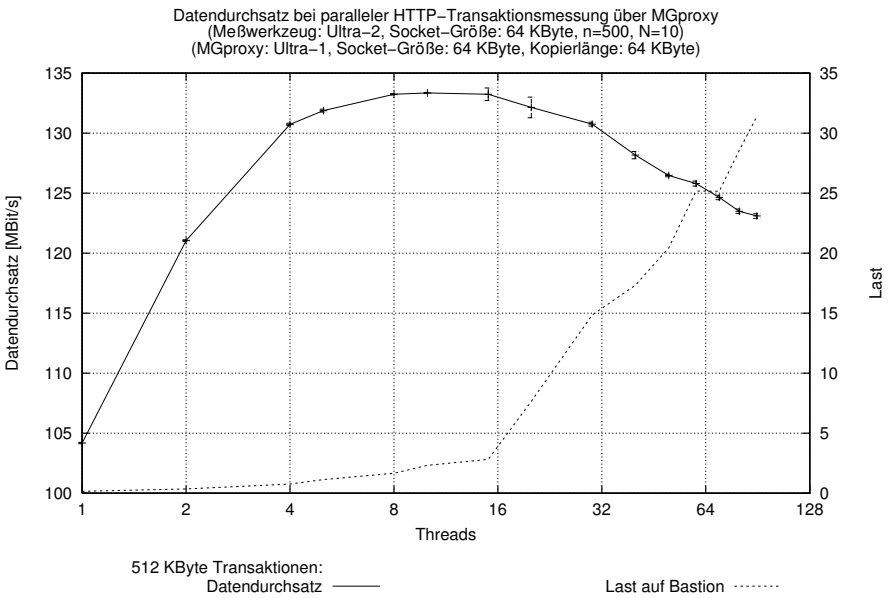


Abbildung 4.75: Gesamtdatendurchsatz und CPU-Last bei längeren Transaktionen

zusätzlich Authentisierungs- und Zugriffskontrollmechanismen (z.B. "Scanner" zur Erkennung ausführbarer Inhalte) benötigt. Außerdem ist damit zu rechnen, daß zukünftig zunehmend Verschlüsselung eingesetzt wird, um die Kommunikation gegen Abhören oder Manipulation abzusichern. Zugriffskontrollmechanismen und Verschlüsselungsverfahren werden auf den gesamten übertragenen Daten angewendet. Authentisierungsmechanismen werden hingegen beim Verbindungsaufbau eingesetzt, um den Kommunikationspartner zu authentisieren. Eine Kombination von Verschlüsselung der übertragenen Daten und sicherer Authentisierung beim Verbindungsaufbau ist zwar sehr empfehlenswert, beide Verfahren können aber auch unabhängig voneinander eingesetzt werden. Während die Belastung des Firewalls bei Authentisierungsverfahren von der Anzahl der in einem Zeitintervall durchgeführten Authentisierungen abhängt, ist bei Zugriffskontrollmechanismen und Verschlüsselungsverfahren die übertragene Datenmenge ausschlaggebend für die Belastung des Firewalls. Daher sind für Untersuchungen der Performanzauswirkungen von Authentisierungsmechanismen andere Messungen erforderlich als für Untersuchungen der Auswirkungen von Zugriffskontrolle und Verschlüsselung. In diesem Abschnitt werden Messungen zu den Performanzauswirkungen von Zugriffskontrolle und Verschlüsselung durchgeführt. Die Auswirkungen von Authentisierungsverfahren auf die Performanz werden im nächsten Abschnitt untersucht.

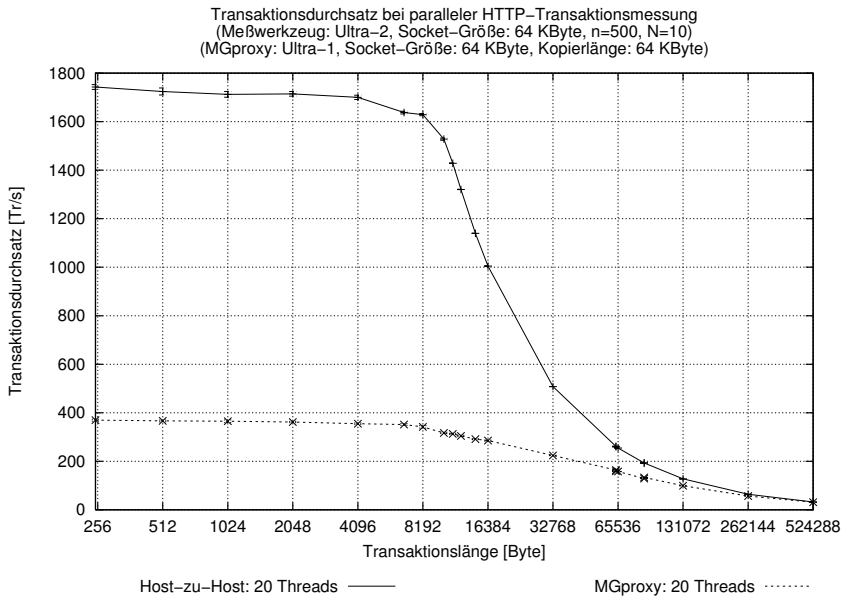


Abbildung 4.76: Transaktionsdurchsatz über "MGproxy" bei längeren Transaktionen

Um die Auswirkungen von Zugriffskontrollmechanismen und Verschlüsselung auf die Performanz von Proxy-Servern zu untersuchen, wird "MGproxy" um drei weitere Varianten ergänzt:

"MGproxy" mit "Executable Content Scanner": Die Sicherheitsrisiken durch die Übertragung von ausführbaren Inhalten, wie Web-Seiten mit "Java", "Java-Script" oder "Active-X" sowie das Versenden von Programmen oder Makros in E-Mails, machen eine Überprüfung im Firewall erforderlich. Durch das "Scannen" von Daten vor der Weiterleitung sind erhebliche Performanzverluste zu erwarten. Um die Größenordnung dieser Verluste quantifizieren zu können, wird "MGproxy" um einen schnellen "Scanner" ergänzt, der in den vom Server an den Client gesendeten Daten nach HTML-Markierungen ("Tags") für ausführbaren Inhalt sucht. Gefundene Markierungen werden vom Proxy entfernt.

Ergänzende Hinweise zur Messung:

- Die vom Meßwerkzeug generierten Daten enthalten keine Verweise auf ausführbare Inhalte.
- Es findet keine Überprüfung der vom Client an den Server gesendeten Daten statt. Eine Überprüfung in dieser Richtung könnte etwa den Zugriff auf bestimmte URLs verweigern. Da der Aufwand für eine

Überprüfung von der übertragenen Datenmenge abhängt und vom Client an den Server nur geringe Datenmengen gesendet werden, sind hier entsprechend geringere Auswirkungen auf die Performanz zu erwarten.

- Die auf hohe Performanz optimierte prototypische Implementation eines Proxies mit "Executable Content Scanner" erreicht noch nicht die notwendige Stabilität, um bei allen Meßpunkten eine fehlerfreie Messung zu garantieren. Nur Meßpunkte bei denen die geforderte Anzahl von Wiederholungen zur Berechnung der Vertrauensintervalle durchgeführt werden konnten, sind bei der Auswertung berücksichtigt worden.

"MGproxy" mit "3DES"-Verschlüsselung: DES ist eine Block-Chiffre ("Block Cipher"), die bei jedem Durchlauf 64 Bit (bzw. 8 Byte) große Blöcke verschlüsselt.[147] Kürzere Nachrichten müssen gegebenenfalls durch Anhängen von Fülldaten ("Padding") vor der Verschlüsselung mit DES (oder 3DES) auf ein Vielfaches von 8 Byte verlängert werden. Dieses "Padding" bei Block-Chiffren führt zu einer Verlängerung der übertragenen Nachrichten. Im ungünstigsten Fall überschreitet die Nachricht nach dem "Padding" gerade die MSS (bzw. MTU), was die Generierung eines zusätzlichen, kurzen TCP-Segments oder allgemein eines zusätzlichen Datagramms erforderlich macht. Die Folge sind ein höherer Verarbeitungsaufwand in den Endsystemen. Außerdem müssen mehr Header-Informationen übertragen werden, wodurch ein geringerer Netzdurchsatz für die Übertragung von Nutzdaten verbleibt.

Nachdem das etablierte DES mit einer Schlüssellänge von nur 56 Bits heute als nicht mehr ausreichend sicher erachtet wird [Blaze et al. 96],[148] wird zukünftig voraussichtlich "Triple-DES" (3DES) als neuer Standard zur Absicherung der Kommunikation eingesetzt werden. Anstelle der einmaligen DES-Verschlüsselung tritt bei 3DES ein Verschlüsseln, Entschlüsseln und erneutes Verschlüsseln mit unterschiedlichen Schlüsseln. Der Arbeitsaufwand ist gegenüber einer normalen DES-Verschlüsselung etwa dreimal so hoch. Da die Verschlüsselung zudem in Software implementiert wird, führt der Einsatz von Verschlüsselung auf einem Firewall zu einer erheblichen, zusätzlichen Belastung.

---

147 Da die einzelnen Blöcke unabhängig voneinander verschlüsselt werden, können Angreifer die Reihenfolge der Datenblöcke ändern. Um diese Manipulationsmöglichkeit auszuschließen, wird über das "Cipher Block Chaining" (CBC) eine Abhängigkeit zwischen den einzelnen Blöcken erzeugt.

148 Siehe auch Abschnitt 2.2.

Für die Messungen wird der "MGproxy" um ein Modul ergänzt, das die vom Server an den Client gesendeten Daten mit 3DES/CBC verschlüsselt.[149] Da der Client die empfangenen Daten nicht auswertet, ist eine Entschlüsselung auf dem Client nicht notwendig. Die vom Client zum Server gesendeten Daten ("GET-Request") werden nicht vom Proxy verschlüsselt, weil diese Daten vom Server ausgewertet werden müssen und aufgrund der geringen Datenmenge auch nur geringe Performanzauswirkungen durch die Verschlüsselung dieser Daten zu erwarten sind.[150] Dieses Vorgehen ermöglicht die Messung, ohne Veränderung an den Meßwerkzeugen erforderlich zu machen. Die Meßergebnisse können daher direkt mit den bereits bei anderen Messungen gewonnenen Daten verglichen werden.

"MGproxy" mit "RC4"-Verschlüsselung: "RC4" ist eine von Ron Rivest entwickelte Strom-Chiffre ("Stream Cipher"). Im Gegensatz zu Block-Chiffren können mit Strom-Chiffren Nachrichten beliebiger Länge verschlüsselt werden, so daß ein "Padding" nicht notwendig ist und die ursprüngliche Nachrichtenlänge erhalten bleibt. Nach [Schneier 96] soll RC4 zudem etwa 10-mal schneller sein als DES. Die Integration von "RC4" in "MGproxy" beschränkt sich wie bei dem für 3DES gewählten Ansatz auf die vom Server zum Client übertragenen Daten.[151]

Da die Performanzauswirkungen dieser Mechanismen von der übertragenen Datenmenge abhängen, sind Messungen mit kurzen Transaktionen wenig sinnvoll, da deren Performanz stark durch die Verbindungsaufbauzeiten dominiert wird. Um Auswirkungen durch Verbindungsaufbauzeiten weitestgehend auszuschließen, soll der Datendurchsatz mit dem bereits zur Messung der maximalen Anzahl paralleler Verbindungen (s. Abschnitt 4.4.2.5) verwendeten Meßskript gemessen werden. Dieses Meßskript führt 30 Sekunden lange Transaktionen durch. Da der Datendurchsatz bei den Messungen zur maximalen Anzahl paralleler Verbindungen bei etwa 40 parallelen Threads einbrach, werden hauptsächlich Messungen mit bis zu 50 Threads durchgeführt. Zwei weitere Meßpunkt liegen bei 60 und 100 Threads.

Die Messungen (s. Abb. 4.77) zeigen starke Auswirkungen dieser drei Verfahren auf den Datendurchsatz. Am stärksten sind die Einbußen, wie erwartet, bei der

---

149 Die 3DES/CBC-Verschlüsselungsroutinen sind "Secure Shell" (SSH) entnommen. Verfügbar über `ftp://ftp.cert.dfn.de/pub/tools/net-crypt/ssh/ssh-1.2.23.tar.gz`

150 Bei den durchgeführten 30-Sekunden Messungen beträgt die vom Server zum Client gesendete Datenmenge ein Vielfaches der vom Client an den Server gesendeten Datenmenge. Selbst bei einem Datendurchsatz von nur 1 MBit/s können in 30 Sekunden fast 4 MB Daten gesendet werden. Dagegen haben die 100 Byte des "GET-Request" vom Client zum Server einen vernachlässigbar geringen Anteil.

151 Die Verschlüsselungsroutine stammt ebenfalls aus SSH. Aus patent- und namensrechtlichen Gründen (siehe auch [Schneier 96, S. 397f]) wird das Verfahren bei SSH als "Arcfour" bezeichnet.

3DES-Verschlüsselung, mit der nur ein Durchsatz von 3,3 - 3,4 MBit/s erreicht werden kann. Aber auch der Datendurchsatz bei RC4-Verschlüsselung erreicht mit maximal 44 MBit/s einen Wert, der weit unter dem bei der reinen Kopierfunktion über "MGproxy" gemessenen Wert von 134 MBit/s liegt (s. Abb. 4.65). Dabei ist zu beachten, daß das Verschlüsselungsverfahren bei den Messungen jeweils nur einmal auf die übertragenen Daten angewendet wird; d.h. der Proxy verschlüsselt die Daten nur auf einer der beiden Verbindungen. Wenn beide Verbindungen (vom Client zum Proxy und vom Proxy zum Server) abgesichert werden sollen, müssen die zu übertragenden Daten zunächst entschlüsselt, gegebenenfalls überprüft und anschließend wieder verschlüsselt werden. Bei diesem "Umschlüsseln"[152] wird die CPU doppelt belastet, so daß mit einer weiteren Halbierung des Durchsatzes gerechnet werden muß.

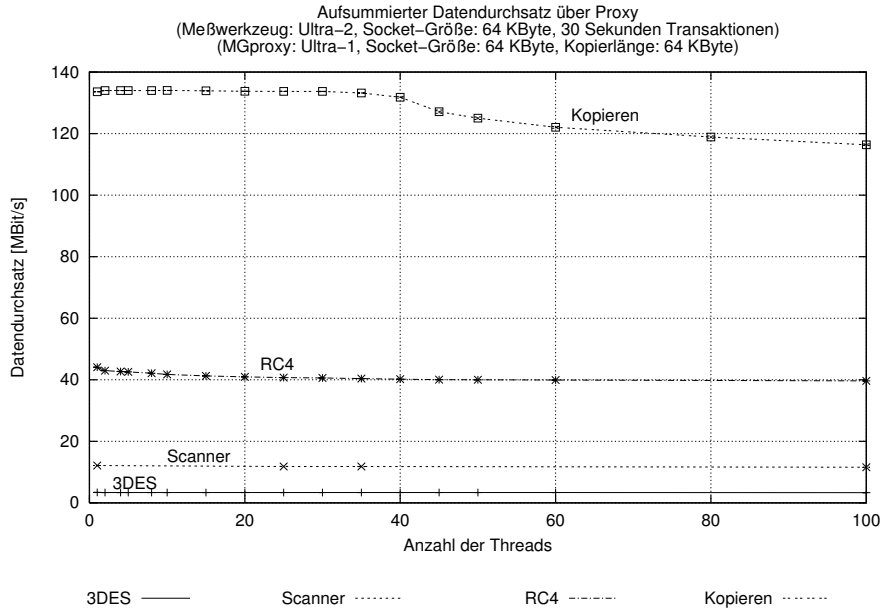


Abbildung 4.77: Datendurchsatz bei Verschlüsselung und Zugriffskontrolle

Überraschenderweise ist der Datendurchsatz bei der "Scanner"-Funktion mit maximal 12 MBit/s geringer als bei der RC4-Verschlüsselung.[153] Über mehrere par-

[152] Das Entschlüsseln und anschließende erneute Verschlüsseln wird hier als "Umschlüsseln" bezeichnet, da es aus Sicherheitsgesichtspunkten nicht empfehlenswert ist, auf beiden Verbindungen dieselben Schlüssel zu verwenden. Für die Performanz des Proxies wäre es hingegen vorteilhaft, wenn auf beiden Verbindungen derselben Schlüssel verwendet werden könnte. Dann wäre es nämlich ausreichend, die Daten zwischenzuspeichern und nur für die Zugriffskontrolle des Proxies zu entschlüsseln. Nach der Überprüfung könnten die zwischengespeicherten verschlüsselten Daten weitergesendet werden.

[153] Mit ähnlicher Funktionalität erreicht der HTTP-Proxy ("http-gw") aus dem TIS-FWTK bei einer Vergleichsmessung sogar nur einen Durchsatz von 0,1 MBit/s.

allele Verbindungen kann kein höherer Gesamtdatendurchsatz erreicht werden als auf einer einzelnen Verbindung. Der Gesamtdatendurchsatz nimmt vielmehr leicht ab, da durch die Bearbeitung mehrerer paralleler Datenströme der "Overhead" auf der Bastion steigt.

Die Belastung der Bastion durch den "Scanner" bzw. die Verschlüsselung zeigen die CPU-Lastkurven in Abbildung 4.78. Im Gegensatz zu der reinen Kopierfunktion, bei der die CPU-Last erst bei mehr als 40 parallelen Verbindungen sprunghaft ansteigt, benötigen die Verschlüsselungs- und "Scanner"-Funktionen bereits bei einer Verbindung die gesamte Prozessorleistung der Bastion. Dadurch steigt die CPU-Last auf der Bastion auf einen Wert, der nur knapp unter der Anzahl der parallelen Verbindungen liegt.[154]

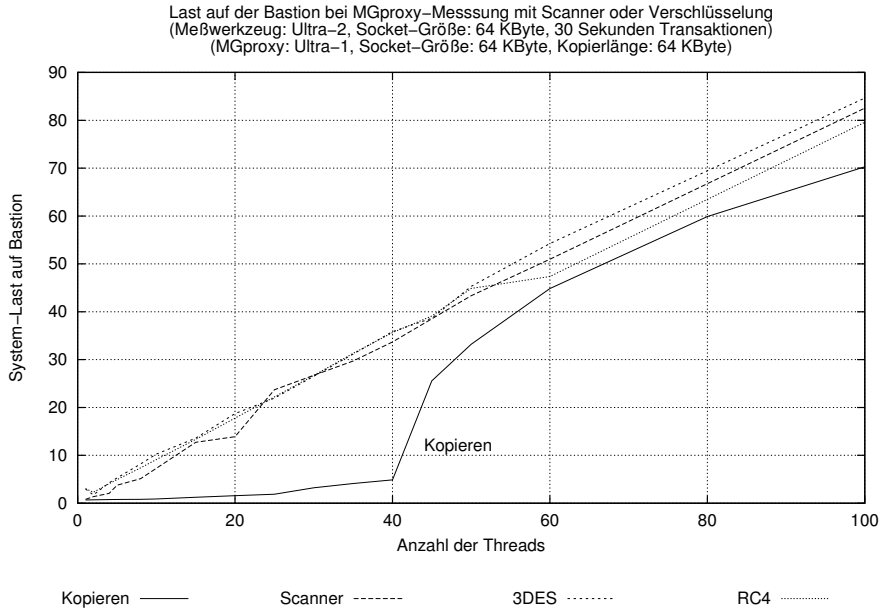


Abbildung 4.78: CPU-Last auf der Bastion durch "Scanner" oder Verschlüsselung

Fazit: Im Gegensatz zu Messungen der Performanz von Proxies mit reiner Kopierfunktion, deren Datendurchsatz in Anbetracht der Verbindungsaufbauzeiten und des begrenzten Transaktionsdurchsatzes nur eine geringe Bedeutung hat, erweist sich der Datendurchsatz bei Operationen wie Verschlüsselung oder Überprüfung der übertragenen Daten auf ausführbare Inhalte als Engpaß. Zwar werden bei den Verbindungsaufbauzeiten und beim Transaktionsdurchsatz keine besseren Werte erreicht, bei längeren Transaktionen treten diese Effekte aber wegen

[154] Während zum Beispiel 100 Verbindungen aktiv sind, macht sich eine CPU-Last von knapp unter 100 dadurch bemerkbar, daß Benutzereingaben von der Bastion nur mit sehr großen Verzögerungen bedient werden.

des geringen Datendurchsatzes in den Hintergrund. Da eine Verschlüsselung auf nur einer Verbindung den Prozessor der Bastion bereits voll auslastet, müssen parallele Verbindungen über geeignete Verfahren auf mehrere Bastionen verteilt werden, um akzeptable Datendurchsätze zu erzielen. Der geringe Datendurchsatz von 3,4 MBit/s bei 3DES würde eine Aufteilung von mindestens 40 Verbindungen auf einzelne Bastionen erforderlich machen, um in der Summe den Datendurchsatz eines 155 MBit/s ATM-Netzes zu erzielen. Aufgrund der damit verbundenen hohen Kosten erscheint der Einsatz von 3DES zur Verschlüsselung von übertragenen Daten für Firewalls kaum durchführbar.[155] Möglicherweise führt hier eine Hardware-Unterstützung für 3DES[156] zusammen mit Parallelisierungskonzepten zu akzeptablen Lösungen.

4.4.2.9 Performanzauswirkungen von kryptographischer Authentisierung

Die Authentisierung der Kommunikationspartner zu Beginn einer Verbindung hat hohe Auswirkungen auf den Transaktionsdurchsatz, wie bereits die Messungen im Abschnitt 4.2.7 (S. 272) gezeigt haben. Insbesondere die RSA-Berechnungen, die für eine kryptographische Authentisierung und für einen Schlüsselaustausch benötigt werden, sind sehr rechenintensiv und belasten daher die Kommunikationspartner sehr. Wie die Messungen in Abschnitt 4.4.2.7 gezeigt haben, ist der Transaktionsdurchsatz von Proxies bereits ohne den Einsatz kryptographischer Authentisierungsmechanismen ein Engpaß. Um zu ermitteln, wie weit sich dieser Engpaß durch den Einsatz kryptographischer Authentisierungsmechanismen verschärft, wird in "MGproxy" die SSL-Bibliothek "SSLeay" integriert. Mit den ebenfalls um SSL-Funktionen erweiterten Meß-Client und -Server, die bereits im Abschnitt 4.2.7 beschrieben wurden, können die Auswirkungen kryptographischer Authentisierungsmechanismen auf Proxies untersucht werden.

Da ein Proxy sowohl Verbindungen von einem Client annehmen als auch zu einem Server eröffnen kann, sind sowohl die Server-seitigen als auch die Client-seitigen SSL-Funktionen beim Proxy erforderlich, damit eine kryptographische Absicherung der Verbindungen mit beiden Kommunikationspartnern über SSL erfolgen kann. Die Verbindungen zum Client und zum Server sind dabei unabhängig voneinander, so daß unterschiedliche kryptographische Verfahren und Schlüssellängen eingesetzt werden können. Der Einsatz unterschiedlicher Schlüssellängen für beide Verbindungen erscheint zunächst unsinnig zu sein, da die Sicherheit im Prinzip durch die geringere der beiden Schlüssellängen begrenzt wird. Firewalls trennen jedoch Netze mit unterschiedlichem Sicherheitsniveau, so daß der Einsatz länge-

[155] Diese Einschätzung gilt neben Firewalls auch für andere Systeme, die gleichzeitig mit vielen Clients kommunizieren müssen (z. B. Web-Server).

[156] Spezielle Hardware-Implementationen von DES können sehr viel effizienter sein als Software-basierte Implementationen. [Eberle 92]

rer Schlüssel für die Kommunikation über das unsicherere Netz durchaus sinnvoll sein kann. Bei den im folgenden diskutierten Messungen werden jedoch vorerst dieselben Schlüssellängen für beide Verbindungen eingesetzt.

Die Auswirkungen der Authentisierung auf den Transaktionsdurchsatz werden mit der bereits vorgestellten Transaktionsmessung mit 250 Byte langen HTML-Seiten gemessen. Wie bei der bereits im Abschnitt 4.2.7 vorgestellten Messung wird die SSL-Version 3 eingesetzt. Der Schlüsselaustausch und die Authentisierung erfolgen über RSA; für die Absicherung der anschließenden Datenübertragung werden 3DES und SHA-1 eingesetzt. Wie bei der direkten Messung zwischen Meß-Client und Meß-Server erfolgt auch bei der Messung über den Proxy die Authentisierung jeweils nur in einer Richtung. Der Proxy authentisiert sich beim Client und der Server authentisiert sich beim Proxy. Für den Einsatz in einem Firewall wäre die Authentisierung in der entgegengesetzten Richtung (Authentisierung des Clients beim Proxy) wichtiger als die hier durchgeführte Authentisierung des Servers beim Client. Da der Proxy aber unabhängig von der Richtung der Authentisierung auf jeweils einer Verbindung selbst Authentisierungsinformationen berechnen und auf der anderen Verbindung lediglich die Authentisierung des Kommunikationspartners überprüfen muß, ist die Belastung für den Proxy bei beiden Authentisierungsrichtungen gleich hoch. Wünschenswert wäre darüber hinaus eine gegenseitige Authentisierung der Kommunikationspartner, die jedoch noch nicht implementiert werden konnte. Zur Messung der Auswirkungen von Authentisierungsmechanismen auf die Performanz von Proxies reicht die vorliegende Implementation mit einer Authentisierung lediglich in einer Richtung aus.

Meß-Client und Meß-Server werden, um die Vergleichbarkeit mit den bereits durchgeführten Messungen zu gewähren, auf "Ultra-2"-Systemen eingesetzt, während der Proxy auf einem "Ultra-1"-System eingesetzt wird. Da wegen des bereits in Abschnitt 4.2.7 erwähnten Fehlers in der SSL-Bibliothek auf dem "Ultra-1"-System keine 2048 Bit langen Schlüssel verwendet werden können, werden nur Messungen mit 512 und 1024 Bit langen Schlüsseln durchgeführt.

Die Ergebnisse der Messungen mit SSL über den Proxy sind in Abbildung 4.79 wiedergegeben. Bei einer Schlüssellänge von 1024 Bit wird der maximale Transaktionsdurchsatz von 6,05 ± 0,00 Tr/s bei 8 Threads erreicht. Der maximale Transaktionsdurchsatz ist beim Einsatz von 512 Bit langen Schlüsseln erwartungsgemäß höher und erreicht bei 16 Threads ein Maximum von 26,10 ± 0,01 Tr/s. Dabei wird aber bereits ab 10 Threads ein Transaktionsdurchsatz von über 26 Tr/s erreicht. Im Vergleich hierzu wurde ohne Einsatz kryptographischer Verfahren über "MGproxy" ein Transaktionsdurchsatz von 392 Tr/s gemessen. Durch den Einsatz von Authentisierungsmechanismen mit 1024 Bit langen Schlüsseln reduziert sich der Transaktionsdurchsatz des Proxies somit um etwa 98%. Selbst mit der allgemein als zu gering eingestuften Schlüssellänge von 512 Bit tritt eine Reduktion um 93% gegenüber dem Transaktionsdurchsatz über den

Proxy ohne Einsatz kryptographischer Authentisierungsmechanismen ein. Die Authentisierungsmechanismen führen also zu einer erheblichen Verschärfung des ohnehin vorhandenen Engpasses von Proxies beim Transaktionsdurchsatz.

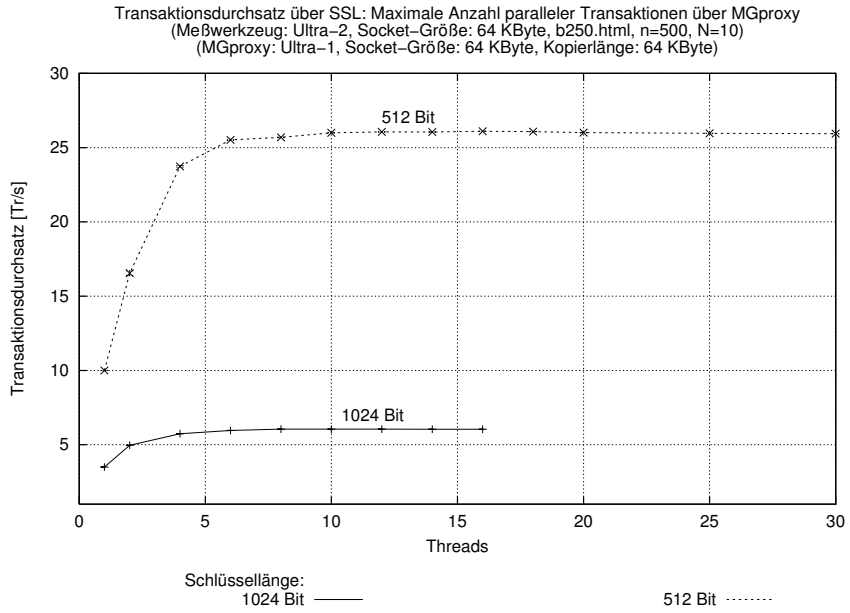


Abbildung 4.79: Transaktionsdurchsatz von "MGproxy" beim Einsatz von SSL

Die CPU-Last der Bastion, auf dem der Proxy eingesetzt wird, ist in Abbildung 4.80 wiedergegeben. Untersuchungen der CPU-Last während der Messungen haben bereits bei den direkten Messungen zwischen Meß-Client und Meß-Server eine ungleichmäßige Verteilung der Last zu Ungunsten des Servers aufgezeigt. Auch bei den Messungen über einen Proxy ist der Meß-Server stärker belastet als der Meß-Client. Der Proxy muß jedoch die RSA-Berechnungen zur Authentisierung sowohl als Server (bei der Verbindung mit dem Meß-Client) als auch als Client (bei der Verbindung mit dem Meß-Server) durchführen, so daß beim Proxy eine mehrfache Belastung entsteht. Diese mehrfache Belastung zeigt sich auch beim Vergleich des Transaktionsdurchsatzes mit einer direkten Messung zwischen Meß-Client und Meß-Server (s. Tab. 4.5, S. 276). Mit 6,74 Tr/s bei 1024 Bit Schlüssellänge und 34,71 Tr/s bei 512 Bit Schlüssellänge werden bei direkten Messungen zwischen ebenfalls auf "Ultra-1"-Systemen eingesetzten Meß-Client und Meß-Server zwischen 11 und 33% höhere Transaktionsdurchsätze erreicht.

Fazit: Der Transaktionsdurchsatz von Proxies, der ohnehin bereits einen Engpaß darstellt, sinkt durch den Einsatz kryptographischer Authentisierungsmechanismen erheblich. Der mit einer akzeptablen Schlüssellänge von 1024 Bit gemessene Transaktionsdurchsatz von etwa 6 Transaktionen pro Sekunde reicht

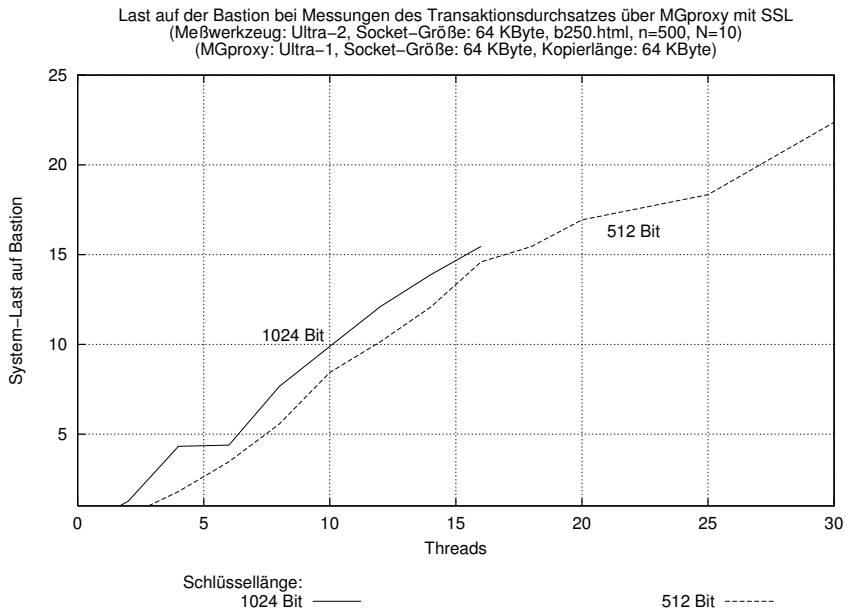


Abbildung 4.80: Last auf der Bastion bei Transaktionsdurchsatzmessung über "MGproxy" mit SSL

insbesondere in Hochgeschwindigkeitsnetzen bei weitem nicht aus. Damit Proxies durch den Einsatz kryptographischer Authentisierungsmechanismen nicht zu einem Engpaß werden, sind zwei Lösungen denkbar. Zum einen kann der Transaktionsdurchsatz durch eine Lastverteilung auf mehrere parallele Proxies erhöht werden. Zusätzlich muß aber auch der wiederholte Verbindungsauf- und -abbau vermieden werden, wie er zum Beispiel bei HTTP üblich ist. Hierfür ist die Entwicklung neuer Protokolle erforderlich.

4.4.3 Abschließende Bewertung der Performanz von Proxy-Servern

Die Messungen in diesem Abschnitt haben den Proxy-Server als Engpaß in Hochgeschwindigkeitsnetzen identifiziert. Im Vergleich zu Packet Screens führt der Einsatz von Proxy-Servern zu viel stärkeren Performanzeinbußen. Bereits bei einfachen Proxy-Servern wie "Socks" bleibt der Datendurchsatz weit unterhalb des zur Verfügung stehenden Netzdurchsatzes. Durch Optimierungen der Kopierfunktion des Proxy-Servers "MGproxy" konnte zumindest für große Schreibzugriffslängen (ab 6 KByte) ein Datendurchsatz von 134 MBit/s erzielt werden. Wichtigster Parameter war dabei die Kopierlänge, mit der der Proxy-Server die Daten von der einen Verbindung liest und sie auf der zweiten Verbindung weiter-

schickt. Muß der Proxy-Server jedoch über die reine Kopierfunktion hinaus Operationen auf den übertragenen Daten ausführen, wird auch der Datendurchsatz des Proxies zum Engpaß. Das Scannen der Daten nach ausführbaren Inhalten ermöglicht nur noch einen Datendurchsatz von 12 MBit/s. Beim Einsatz von Verschlüsselung liegen die maximalen Durchsätze bei 44 MBit/s für RC4 und bei lediglich 3,4 MBit/s für 3DES. Muß der Proxy die Daten umschlüsseln, d. h. zunächst entschlüsseln und anschließend wieder mit einem anderen Schlüssel verschlüsseln, dann ist mit einer weiteren Halbierung der genannten Durchsätze zu rechnen.

Während eine Packet Screen Dateneinheiten unabhängig voneinander verarbeitet und es daher ausreichend ist, den Datendurchsatz auf einer einzelnen Verbindung zu messen, muß ein Proxy für jede Verbindung Ressourcen bereitstellen. Wichtiger als die Datendurchsätze, die auf einer einzelnen Verbindung über einen Proxy erzielt werden können, ist somit der Gesamtdatendurchsatz bei mehreren parallelen Verbindungen, der den zur Verfügung stehenden Netzdurchsatz erreichen sollte. Mit bis zu 30 parallelen Verbindungen kann "MGproxy" den zur Verfügung stehenden Netzdurchsatz ausnutzen. Bei zusätzlichen parallelen Verbindungen führt die Verwaltung der Verbindungen im Proxy zu einer Reduktion des Gesamtdurchsatzes. Der Proxy teilt die zur Verfügung stehenden Betriebsmittel ausreichend gleichmäßig zwischen den parallelen Verbindungen auf.

Der Unterschied zwischen Konfigurationen mit einer Netzwerkschnittstelle als Bastion oder mit zwei Netzwerkschnittstellen als Gateway-Firewall hat nur geringe Auswirkungen auf den erzielten Datendurchsatz über den Proxy. Bei den übrigen Messungen, bei denen der Datendurchsatz eine untergeordnete Rolle spielt, fällt der Unterschied zwischen Bastion- und Gateway-Setup noch geringer aus.

Entscheidender noch als beim Datendurchsatz sind die Auswirkungen auf Nachrichtenlaufzeit, Verbindungsaufbauzeit und Transaktionsdurchsatz. Konzeptionsbedingt müssen hier bei Proxy-Servern hohe Performanzverluste in Kauf genommen werden. Bei dem symmetrischen Aufbau im LAN mit zwei Verbindungen über kurze Distanzen hat die Übertragungszeit einer Nachricht auf dem Netz einen vernachlässigbar kleinen Anteil an der Nachrichtenlaufzeit. Die gemessenen Nachrichtenlaufzeiten werden im LAN allein durch die Verarbeitungszeiten in den Systemen dominiert. Somit ist bei zwei hintereinandergeschalteten Verbindungen selbst im günstigsten Fall zumindest eine Verdoppelung der Nachrichtenlaufzeiten zu erwarten. Bei Verbindungen in WANs ist die für die Übertragung benötigte Zeit nicht mehr vernachlässigbar. Die Nachrichtenlaufzeit muß in WANs aus der Summe der Verarbeitungszeiten in den Systemen und der Nachrichtenübermittlungszeit über das Transportmedium berechnet werden. Da sich die Gesamtlänge des Weges durch die Integration eines Firewalls in der Regel nur geringfügig um wenige Meter verlängert, bleibt die Nachrichtenübermittlungszeit auf dem Medium trotz Einsatz eines Proxy-Servers fast gleich. Die Verdoppelung

der Verarbeitungszeiten in den Systemen führt prozentual jedoch zu einer weniger starken Erhöhung der Nachrichtenlaufzeiten als beim Einsatz von Proxy-Servern in LANs.

Die Verbindungsaufbauzeiten hängen wegen geringer Nachrichtenlaufzeiten im LAN primär von der Implementation des (Proxy-)Servers ab. Zeitintensive Operationen wie das Duplizieren von Server-Prozessen ("`fork`") erhöhen die Verbindungsaufbauzeiten entscheidend. Da in Netzen mit hohem Datendurchsatz die Dauer der Datenübertragung selbst bei Nachrichtenlängen von mehreren KByte gering ist (s. Abschnitt 4.2.4.1), fällt jede Verlängerung der Verbindungsaufbauzeiten um so stärker ins Gewicht. In Abbildung 4.81 wird die Dauer einer Verbindung in Abhängigkeit von der Nachrichtenlänge dargestellt.[157] Die drei Kurven beziehen sich auf die Anzahl von Transaktionen, die bei vorangegangenen Transaktionsmessungen ermittelt wurden. Etwa 35 Transaktionen pro Sekunde sind beim nicht-optimierten "Socks"-Proxy-Server (Standard-Server) möglich. Die Gesamtdauer einer Transaktion beträgt bei dem Standard-Server unabhängig von der Nachrichtenlänge immer mindestens 30 Millisekunden. Etwa 190 Transaktionen können bei der "Multithreaded"-Konfiguration von "MGproxy" erreicht werden. Auf einer direkten Host-zu-Host-Verbindung im LAN sind 520 Transaktionen pro Sekunde möglich. Durch die entsprechend geringen Verbindungsaufbauzeiten kann im LAN innerhalb von 30 Millisekunden bereits fast ein halbes Megabyte Nutzdaten übertragen werden. Deutlich erkennbar ist, wie stark die Verbindungsaufbauzeiten die Gesamttransaktionslänge dominieren. Die drei Kurven werden bei gleichem Verlauf nur um den Betrag der Verbindungsaufbauzeit auf der Y-Achse verschoben. Der prozentuale Anteil des Verbindungsaufbaus an der Gesamttransaktionslänge ist in Abbildung 4.82 wiedergegeben. Bei wiederkehrenden Verbindungen zwischen Client und Proxy besteht daher die wichtigste Optimierungsmöglichkeit durch die Vermeidung des wiederholten Verbindungsauf- und -abbaus. Der Einsatz neuer Protokolle würde es ermöglichen, daß die Verbindung zwischen Client und Proxy länger offengehalten wird und lediglich die Verbindungen zwischen Proxy und unterschiedlichen Servern neu aufgebaut werden müssen. Dieses Vorgehen würde insbesondere auch bei aufwendigen Authentisierungsverfahren Vorteile erbringen, da keine erneute Authentisierung des Clients beim Proxy notwendig ist. Ansätze zur Vermeidung eines wiederholten Verbindungsauf- und -abbaus werden im Abschnitt 5.4 diskutiert.

Besonders deutlich ist in Abbildung 4.82 erkennbar, daß beim Einsatz von Proxy-Servern die Verbindungsaufbauzeiten den größten Engpaß darstellen. Während auf einer direkten Host-zu-Host-Verbindung "nur" bis zu einer Nachrichtenlänge von 32 KByte der Anteil der Verbindungsauf- und -abbauzeiten bei der Gesamttransaktionslänge überwiegt, werden bei Verbindungen über einen Standard-

[157] Für die Übertragungsdauer wird ein Datendurchsatz von 134 MBit/s angenommen.

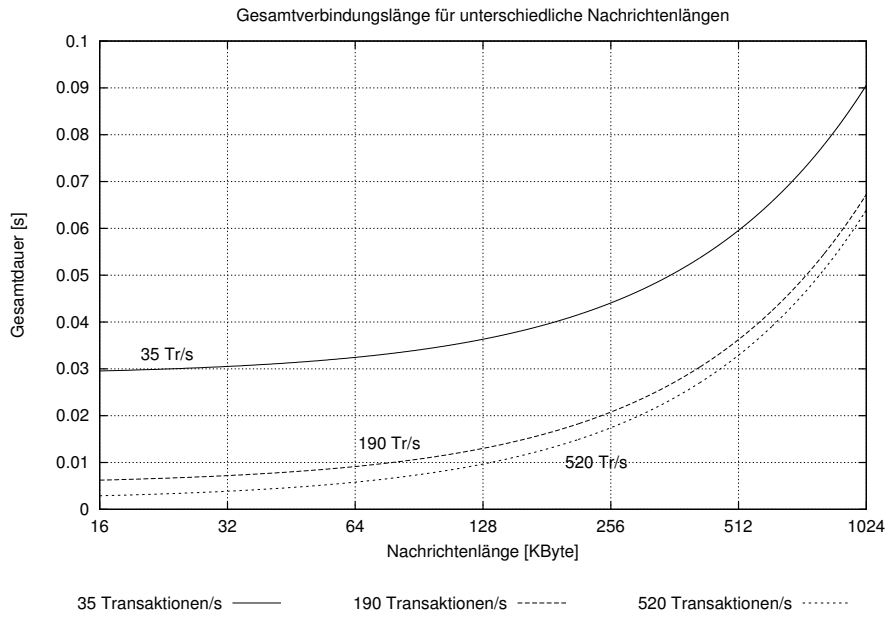


Abbildung 4.81: Anteil des Verbindungsaufbaus an der Übertragung

Proxy-Server Nachrichtenlängen von etwa 500 KByte benötigt, damit die Verbindungsauf- und -abbauzeiten weniger als 50% der Gesamttransaktionslänge ausmachen. Bei fast allen Protokollen sind die durchschnittlichen Nachrichtenlängen (s. Anhang C.1) so gering, daß in einem 155 MBit/s CLIP-Netz praktisch ausschließlich die Verbindungsauf- und -abbauzeiten die Gesamttransaktionslänge bestimmen. HTTP und SMTP liegen mit durchschnittlichen Nachrichtenlängen von 6-7 KByte in einem Bereich, wo auf einer direkten Host-zu-Host-Verbindung 80% und über einen Standard Proxy-Server (35 Tr/s) sogar über 98% der Transaktionslänge vom Verbindungsaufbau abhängt. Wie gering bei dieser Nachrichtenlänge der Einfluß des Datendurchsatzes im Vergleich zu den Verbindungsauf- und -abbauzeiten ist, zeigt Abbildung 4.83. In der Abbildung ist erkennbar, daß es bei 7 KByte langen Nachrichten kaum Abhängigkeiten der Transaktionslänge vom Datendurchsatz gibt, sofern der Datendurchsatz des Proxy-Servers über 10 MBit/s liegt. Noch kürzere Nachrichtenlängen als HTTP weisen die Protokolle DNS und AUTH auf, die mit Nachrichtenlängen von unter 500 Byte sogar außerhalb des in Abbildung 4.82 dargestellten Bereichs liegen. Lediglich bei FTP-Datenverbindungen und bei größeren Datenübertragungen über SSH entfallen mit durchschnittlichen Nachrichtenlängen von 200 bis 300 KByte auch bei Verbindungen über einen Proxy-Server nennenswerte Anteile der Gesamttransaktionslänge auf die Übertragung der Daten. Bei einem Standard-Server mit 35 Transaktionen pro Sekunde werden bei Nachrichtenlängen um 250 KByte aber immer noch 65% der gesamten Transaktionzeit beim Verbindungsaufbau verbraucht. Auf einer di-

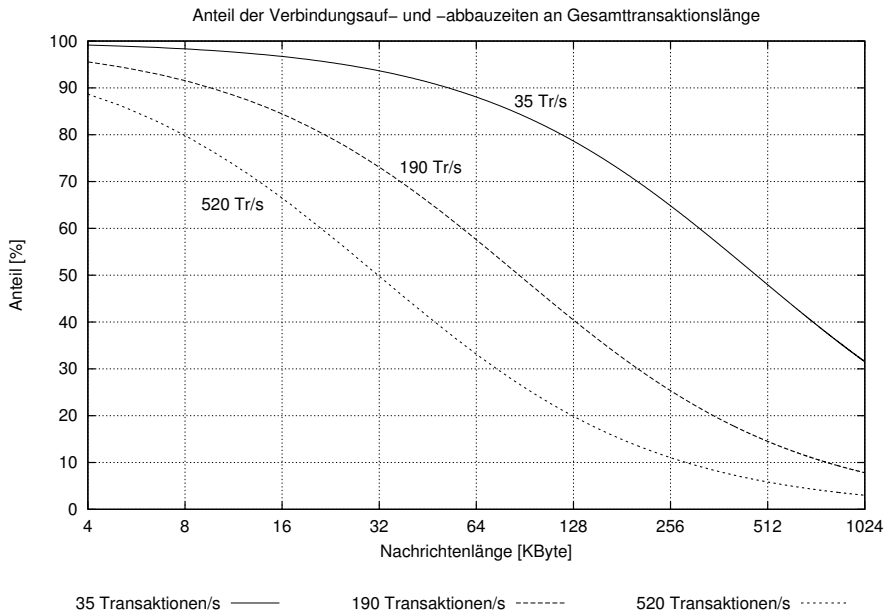


Abbildung 4.82: Anteil der Verbindungsauf- und -abbauzeiten an der Gesamttransaktionszeit

rekten Host-zu-Host-Verbindung entfallen bei dieser Nachrichtenlänge nur etwa 10% auf den Verbindungsauf- und -abbau.

Die Verlängerung der Verbindungsaufbauzeit ist die durch ein Endsystem beobachtete Performanzauswirkung eines Proxy. Während der Wartezeit beim Verbindungsaufbau könnte der Netzdurchsatz durch parallele Verbindungen anderer Clients ausgenutzt werden. Beim aufsummierten Durchsatz über Verbindungen von mehreren Clients wäre trotzdem das Ausnutzen des Netzdurchsatzes möglich, wenn eine ausreichend große Anzahl von Clients gleichzeitig Transaktionen ausführen könnte. Damit eine möglichst große Anzahl von Clients gleichzeitig Transaktionen ausführen kann, ist ein hoher Transaktionsdurchsatz des Proxies erforderlich. Der eingeschränkte Transaktionsdurchsatz erweist sich als Hauptengpaß von Proxy-Servern. “MGproxy” ermöglicht mit einem maximalen Transaktionsdurchsatz von knapp 400 Tr/s erst bei Transaktion von 500 KByte Länge einen aufsummierten Datendurchsatz von über 130 MBit/s (s. Abb 4.73). Um den während der längeren Verbindungsaufbauzeiten über den Proxy ungenutzt bleibenden Netzdurchsatz durch parallele Threads ausnutzen zu können, wäre ein höherer Transaktionsdurchsatz des Proxies notwendig. Der Engpaß wird durch die Verbindungsannahme beim Proxy verursacht. Eine Umgehung des Engpasses ist durch die Vermeidung des wiederholten Verbindungsauf- und -abbaus möglich. Da hierdurch jedoch nur wiederholte Verbindungen durch den-

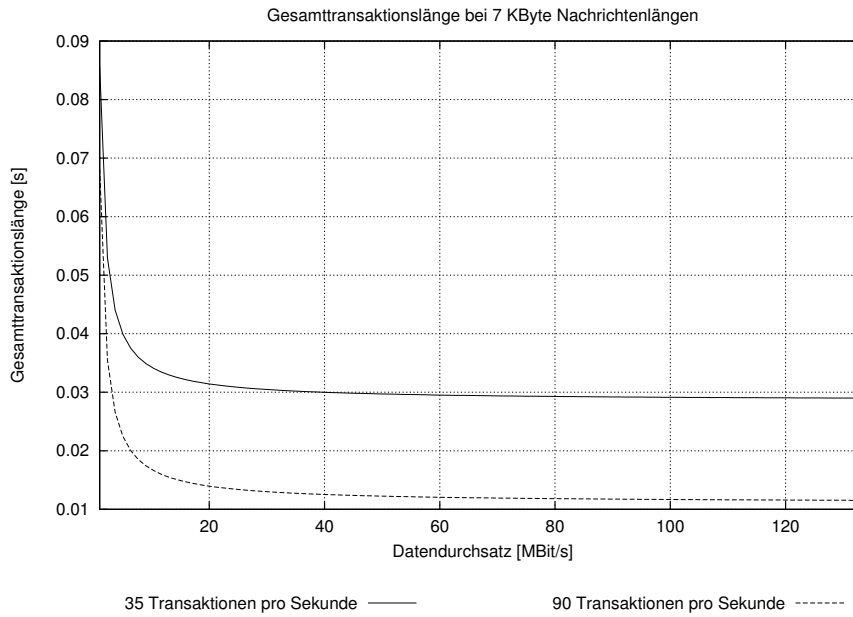


Abbildung 4.83: Transaktionslängen bei unterschiedlichen Datendurchsätzen

selben Client betroffen sind, ist zusätzlich eine Parallelisierung durch Einsatz mehrerer Bastionen erforderlich.

Der Einsatz kryptographischer Authentisierungsverfahren verschärft den Engpaß beim Transaktionsdurchsatz des Proxies erheblich. Bei einer Schlüssellänge von 1024 Bit wird nur noch ein maximaler Transaktionsdurchsatz von etwa 6 Transaktionen pro Sekunde erreicht, so daß selbst bei langen Transaktionen mit mehreren MByte übertragener Daten der zur Verfügung stehende Netzdurchsatz von einem einzelnen Proxy nicht annähernd ausgenutzt werden kann.

Als Fazit der Performanzmessungen kann zusammengefaßt werden, daß die Verlängerung der Verbindungsaufbauzeit und der geringe Transaktionsdurchsatz die wichtigsten Engpässe von Proxy-Servern sind. Zwar schränken Proxies auch den Datendurchsatz ein, jedoch führt erst der Einsatz von aufwendigen "Scannern" oder von Verschlüsselungsverfahren zu deutlichen Auswirkungen auf die Performanz. Die Optimierungen müssen sich daher bei Proxy-Servern in erster Linie auf die Reduzierung der Verbindungsaufbauzeiten und die Steigerung des Transaktionsdurchsatzes konzentrieren. Erst wenn die Überprüfung oder Verschlüsselung der übertragenen Daten erforderlich wird, müssen zusätzlich Lösungen zur Steigerung des Datendurchsatzes gefunden werden. Zur Steigerung der Performanz von Proxies eignen sich zwei Ansätze. Zum einen muß das wiederholte Auf- und Abbauen von Verbindungen vermieden werden. Dies macht neue Protokolle zwischen Client und Proxy erforderlich. Auf der anderen Seite kann durch die

Verteilung der Last auf mehrere parallele Bastionen eine Steigerung der Performanz erzielt werden.

4.5 Performanz von Kombinationen aus Packet Screen und Bastion

In den Abschnitten 4.3 und 4.4 wurde die Performanz von Firewall-*Komponenten* analysiert und bewertet. Beim Aufbau von Firewall-*Architekturen* aus mehreren Firewall-Komponenten ergeben sich kombinierte Performanzauswirkungen, die in diesem Abschnitt diskutiert werden sollen.

Proxy-Server können nur auf Gateway-Firewalls allein eingesetzt werden, da bereits durch den Anschluß des Gateways an beide Netze dafür gesorgt ist, daß jede Kommunikation zwischen den beiden Netzen nur über das Gateway geführt werden kann. Soll ein Proxy-Server demgegenüber auf einer Bastion eingesetzt werden, muß durch zusätzliche Maßnahmen dafür gesorgt werden, daß nur die Bastion mit beiden Netzen kommunizieren darf. Für diese Einschränkung der Kommunikation sorgen in klassischen Firewall-Architekturen eine oder zwei Packet Screens (s. Abschnitt 2.3.2.2). Die Aufgabe der Packet Screen besteht darin, nur Verkehr von und zur Bastion zuzulassen — direkte Kommunikation zwischen internem und externem Netz wird nicht erlaubt. Da die Packet Screen nur Dateneinheiten zu einem Zielsystem (der Bastion) durchlassen muß, kommt dieses Setup mit einer im Vergleich zu reinen Packet Screen Lösungen geringen Anzahl an Filterregeln aus.

Der für die Messungen konfigurierte Aufbau (s. Abb. 4.84) setzt sich zusammen aus einer Packet Screen, realisiert auf einer Workstation mit zwei ATM-Schnittstellen und der Software "IP-Filter", sowie einer mit dem Proxy-Server "MGproxy" konfigurierten Bastion. Die Performanz dieser beiden Firewall-Komponenten wurde bereits in den Abschnitten 4.3 ("IP-Filter") und 4.4.2 ("MGproxy") ausführlich analysiert, so daß ausreichende Vergleichsdaten für die im folgenden diskutierten Messungen zur Verfügung stehen. Bei dem den Messungen zugrundeliegenden Aufbau müssen die zwischen den beiden Netzen übertragenen Daten jeweils einmal von der Packet Screen und einmal vom Proxy-Server verarbeitet werden. Das in Abbildung 4.84 wiedergegebene Setup kann somit als Kombination von Packet Screen und Bastion mit der "Bastion innen" (s. Abb. 2.7) oder "Bastion außen" (s. Abb. 2.9) dienen. Die beiden Varianten unterscheiden sich nur in Sicherheitsgesichtspunkten, die jedoch für die durchgeführten Performanzmessungen keine Relevanz haben. Für Messungen bei "mittiger" oder "beidseitig abgeschirmter Bastion" (s. Abb. 2.8 und 2.10) wird eine andere Testnetz-Konfiguration benötigt. Die Firewall-Architektur mit mittiger Bastion

(s. Abb. 2.8) erfordert eine zweimalige Filterung in derselben Packet Screen. Bei einer "beidseitig abgeschirmten Bastion" (s. Abb. 2.10) werden zwei getrennte Packet Screens für die beiden Filtervorgänge verwendet.

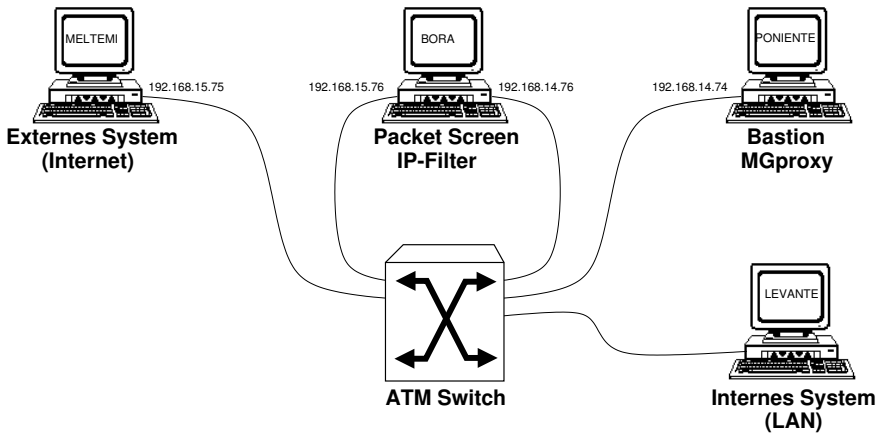


Abbildung 4.84: Testnetz-Konfiguration für Messungen bei Kombinationen von Packet Screen und Bastion

4.5.1 Datendurchsatz über Packet Screen und Bastion

Für die Messung des Datendurchsatzes wird eine TCP-Verbindung vom internen System (`levante`) zum Proxy-Server "MGproxy" auf der Bastion (`poniente`) eröffnet. Der Proxy-Server baut eine zweite Verbindung zum externen System (`meltemi`) auf, deren Datagramme von der Packet Screen (`bora`) überwacht werden. "MGproxy" wird mit einer Kopierlänge von 64 KByte und einer Socket-Größe von 256 KByte konfiguriert. Für die einzelnen Durchsatzmessungen wird auf der Packet Screen die Anzahl der Filterregeln variiert.[158]

Die Messungen zeigen, daß die Datendurchsätze bei Packet Screens mit bis zu maximal 300 Filterregeln durch den Datendurchsatz des Proxy-Servers begrenzt werden. Der Vergleich von Messungen über einen auf einem Gateway-Firewall installierten Proxy-Server (s. Abschnitt. 4.4.2) und den Messungen über die Kombination von Packet Screen und Bastion überdecken sich (s. Abb. 4.85). Lediglich bei 8 KByte langen Schreibzugriffen sind geringfügige Abweichungen erkennbar. Die Abweichungen liegen bei Packet Screens mit bis zu 200 Filterregeln unterhalb von 1 MBit/s. Eine ebenso große Datendurchsatzverringerung wurde bereits

[158] Der Aufbau der konfigurierten Filterregeln wurde bereits in Abschnitt 4.3 ausführlich dokumentiert. Für die Messungen bei Kombinationen von Packet Screen und Bastion wurden dieselben Filterregel-Konfigurationen verwendet.

beim Vergleich von Gateway- und Bastion-Setup (s. Abschnitt 4.4.2.1.2) festgestellt. Die gemessene Datendurchsatzverringerung wird also nicht durch die Packet Screen verursacht, sondern durch die Verwendung einer Bastion mit nur einer Netzwerkschnittstelle. Erst ab 400 Filterregeln wird nicht mehr der volle Datendurchsatz bei 8 KByte langen Schreibzugriffen erreicht. Durch Hinzufügen weiterer Filterregeln sinkt der Durchsatz bei 8 KByte Schreibzugriffen weiter ab, während bei 4 KByte langen Schreibzugriffen Aggregationseffekte erkennbar werden. Zum Beispiel bringt die Erhöhung von 500 auf 600 Filterregeln bei 4 KByte langen Schreibzugriffen so einen um 6 MBit/s höheren Datendurchsatz (118 MBit/s statt 112 MBit/s).

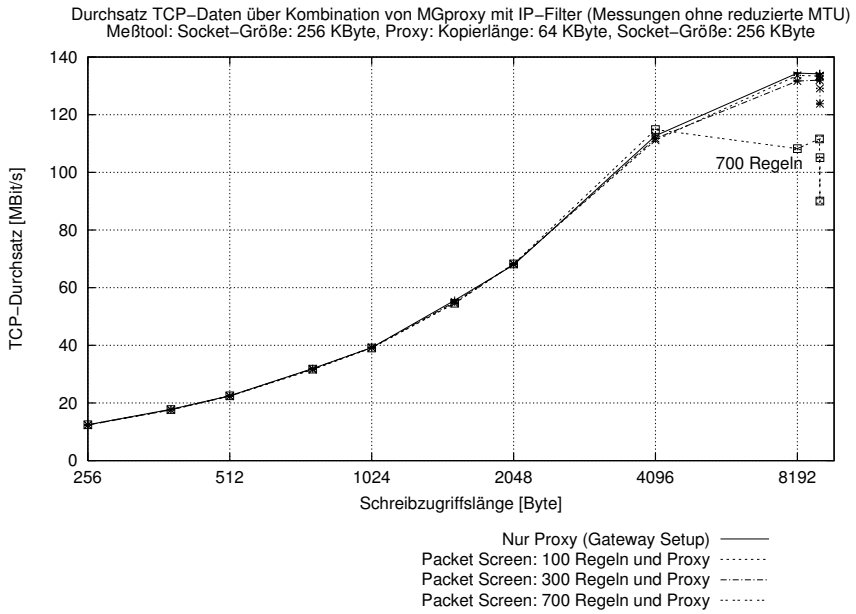


Abbildung 4.85: Durchsatz über Kombination von Packet Screen und Bastion

Im Vergleich zu den Messungen des Datendurchsatzes allein durch eine Packet Screen (ohne zusätzliche Bastion) (s. Abschnitt 4.3) zeigt sich noch einmal sehr deutlich, daß der Datendurchsatz bei Messungen über Kombinationen von Packet Screen und Bastion nur vom Durchsatz des Proxy-Servers bestimmt wird (s. Abb. 4.86). Um eine Vergleichbarkeit der Meßergebnisse zu gewährleisten, müssen alle Messungen mit der voreingestellten MTU-Größe ("Default MTU" von 9.180 Byte) durchgeführt werden. Dies ermöglicht die Aggregation mehrerer kleinerer Schreibzugriffe zu größeren Dateneinheiten und verdeckt somit die Auswirkungen der Packet Screen, die erst bei Messungen mit reduzierter MTU erkennbar sind.[159]

[159] Deutlich zu erkennen sind in der Abbildung 4.86 diese Aggregationseffekte bei der Schreibzugriffslänge von 1.024 Byte, bei der ohne konfigurierte Filterregeln ein schlechterer Durchsatz erzielt wird als mit konfigurierten Filterregeln.

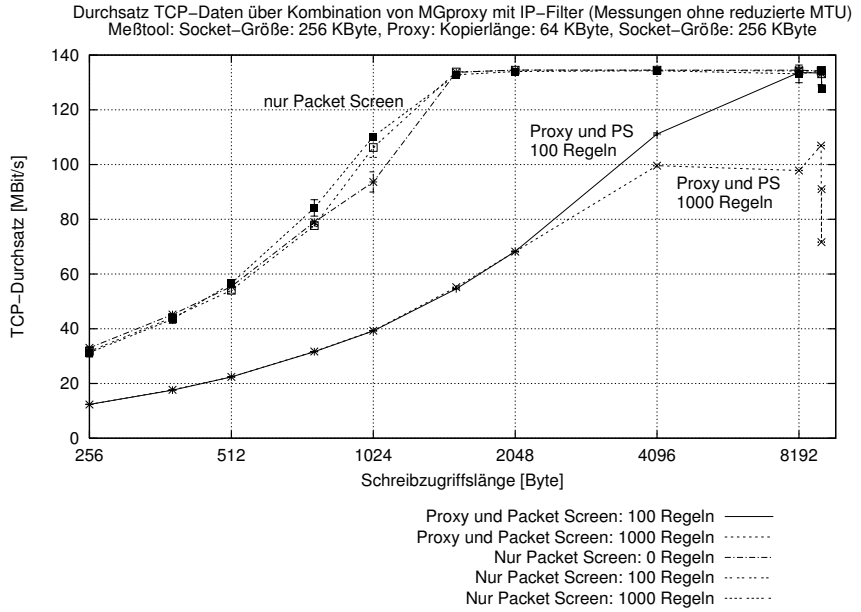


Abbildung 4.86: Durchsatz über Kombination von Packet Screen und Bastion im Vergleich zu Messungen über Packet Screen

4.5.2 Nachrichtenlaufzeit über Packet Screen und Bastion

Für die Messung der Nachrichtenlaufzeiten (s. Abschnitt 4.2.3) wird dasselbe Setup wie bei den im vorigen Abschnitt (4.5.1) beschriebenen Datendurchsatzmessungen verwendet. Auch hier wird wieder von dem internen System über Proxy-Server und Packet Screen zum externen System gemessen. Die gemessene Nachrichtenlaufzeit vom Client zum Server (eine Richtung) ist in Abbildung 4.87 dargestellt. Ebenfalls in der Abbildung enthalten sind die Nachrichtenlaufzeiten über eine Packet Screen (s. Abschnitt 4.3.4) und über den Proxy-Server "MGproxy" (s. Abschnitt 4.4.2.2). Die Nachrichtenlaufzeit über die Kombination von Packet Screen und Bastion ist konstant um etwa 0.3 Millisekunden länger als die Laufzeit über die Packet Screen allein. Die Nachrichtenlaufzeiten der einzelnen Komponenten addieren sich also.[160]

[160] Die gemessenen Nachrichtenlaufzeiten über Packet Screen und Proxy-Server sind um einen konstanten Offset geringer als die Summe der über Packet Screen bzw. Proxy-Server allein gemessenen Nachrichtenlaufzeiten. Dies hängt damit zusammen, daß die bei der Kombination von Packet Screen und Bastion zurückzulegende Entfernung um einen "Hop" kürzer ist als bei den einzelnen Messungen. Bei den Messungen der einzelnen Komponenten müssen jeweils 2 "Hops" (Meßwerkzeug – Proxy – Meßwerkzeug bzw. (Meßwerkzeug – Packet Screen – Meßwerkzeug) zurückgelegt werden. Bei der Kombination von Packet Screen und Bastion wird über drei "Hops" (Meßwerkzeug – Proxy – Packet Screen – Meßwerkzeug) gemessen.

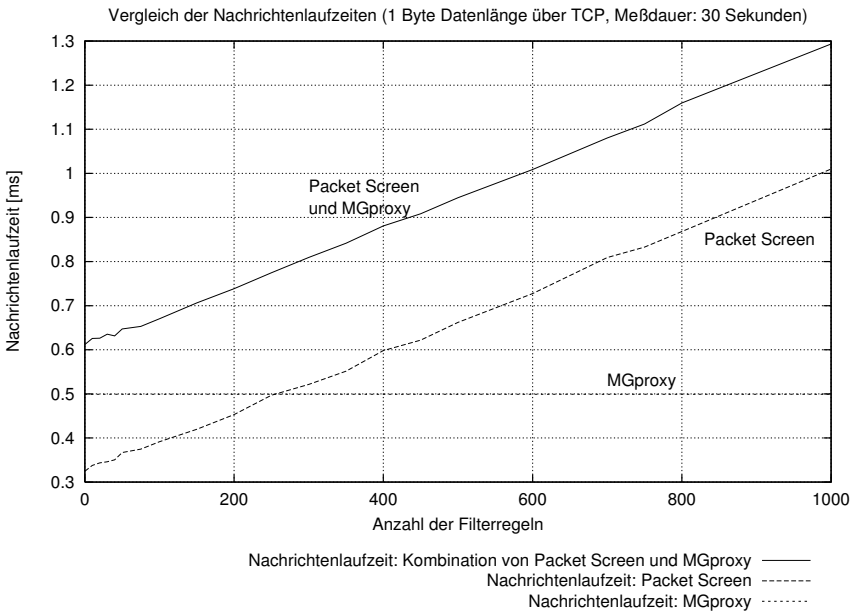


Abbildung 4.87: Nachrichtenlaufzeit über Kombination von Packet Screen und Bastion

4.5.3 Verbindungsaufbauzeit über Packet Screen und Bastion

Da eine Packet Screen nicht aktiv an einem Verbindungsaufbau teilnimmt, sind die Auswirkungen einer Packet Screen auf Verbindungsaufbauzeiten vergleichsweise gering. Abhängig von der Anzahl der Filterregeln sind zwischen 100 und 500 Transaktionen pro Sekunde möglich (s. Abschnitt 4.3.5). Die maximale Anzahl der Transaktionen, die über den eingesetzten Proxy-Server ("MGproxy" als Standard-Server) möglich ist, liegt im Vergleich nur bei geringen 60 bis 70 Transaktionen pro Sekunde (s. Abschnitt 4.4.2.3). Daher ist zu erwarten, daß bei einer Kombination von Packet Screen und Bastion die Ergebnisse durch die Performanz des Proxy-Servers bestimmt werden. Diese Erwartung wird durch die Messungen bestätigt. In Abbildung 4.88 sind die Ergebnisse einer Transaktionsmessung[161] wiedergegeben. Die Anzahl der in der Packet Screen konfigurierten Filterregeln hat nur geringe Auswirkungen auf die Meßergebnisse. Bis zu einer Anzahl von 500 Filterregeln liegen die Meßergebnisse der Transaktionsmessung im selben Bereich von 60 bis 70 Transaktionen pro Sekunde, in dem auch die Meßergebnisse bei Messungen ausschließlich über einen Proxy-Server liegen.

[161] Das Setup entspricht dem in Abschnitt 4.5.1 beschriebenen Aufbau.

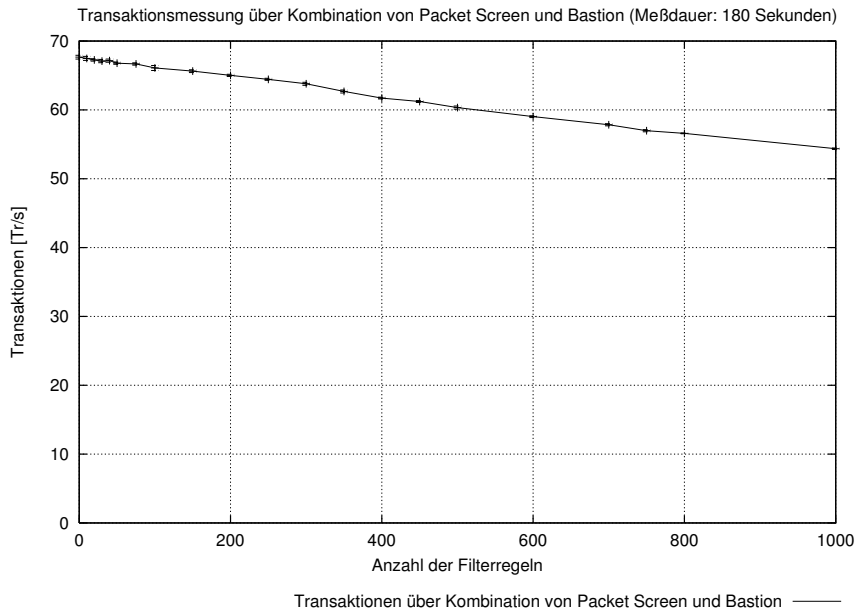


Abbildung 4.88: Transaktionsmessung über Kombination von Packet Screen und Bastion

4.5.4 Abschließende Bewertung der Performanz von Kombinationen aus Packet Screen und Bastion

Die Messungen der Performanz von Kombinationen von Packet Screen und Bastion haben gezeigt, daß im Hinblick sowohl auf Datendurchsatz als auch Verbindungsaufbauzeiten die Messungen durch den auf der Bastion eingesetzten Proxy-Server bestimmt werden. Da die Packet Screen im Gegensatz zum Proxy-Server nicht aktiv in den Verbindungsaufbau eingreift, sind die Auswirkungen auf die Verbindungsaufbauzeiten entsprechend gering. Die sehr viel stärkeren Auswirkungen eines Proxy-Servers im Vergleich zur Packet Screen auf den Datendurchsatz haben die vorherigen Messungen in den Abschnitten 4.3 und 4.4 bereits gezeigt. Der in Abschnitt 4.3 analysierte Engpaß der Packet Screen beim Datagrammdurchsatz kommt bei der Kombination von Packet Screen und Bastion nicht zum Tragen, da zum einen der Durchsatz des Proxy-Servers deutlich unter dem der Packet Screen liegt und auf der anderen Seite die Aggregation mehrerer Schreibzugriffe zu längeren Dateneinheiten durch den Proxy-Server begünstigt werden. Zudem werden bei Kombinationen von Packet Screen und Bastion weniger Filterregeln benötigt als beim alleinigen Einsatz einer Packet Screen als Firewall. Typische Filterkonfigurationen bei Kombinationen aus Packet Screen und Bastion verwenden deutlich weniger als 100 Filterregeln, eine Größenordnung, die kaum Auswirkungen auf die Performanz der Packet Screen hat. Die im

Abschnitt 4.4 diskutierten Messungen zur Ermittlung des Transaktionsdurchsatzes und der maximalen Anzahl paralleler Verbindungen betreffen ebenfalls nur die Performanz des Proxies und werden daher in diesem Abschnitt nicht wiederholt.

Da sich ohne eine Filterung auf Anwendungsebene im Proxy-Server nur selten die benötigte Sicherheit erreichen läßt, ist es erforderlich, Optimierungsbemühungen auf den Proxy-Server zu konzentrieren. Eine Steigerung der Performanz von Packet Screens wird erst dann erforderlich, wenn durch Optimierungen bzw. Parallelisierungen die Performanz von Proxy-Servern deutlich erhöht wurde.

4.6 Abschließende Bewertung der Firewall-Performanz

In diesem Kapitel wurde erstmalig umfassend die Performanz von Firewalls in Hochgeschwindigkeitsnetzen analysiert. Für die in Abschnitt 4.1 identifizierten Performanzkriterien von Firewalls wurden Messungen entwickelt und ausführlich diskutiert (s. Abschnitt 4.2). Dabei konnten nur bei Messungen eines Teils der Performanzkriterien auf bereits vorhandene Meßwerkzeuge zurückgegriffen werden. Für den Transaktionsdurchsatz und die maximale Anzahl paralleler Verbindungen als wesentliche Performanzkriterien von Firewalls waren bisher keine Meßwerkzeuge verfügbar, die sicherstellen konnten, daß während der Messung auch die geforderte Anzahl paralleler Verbindung gleichzeitig aktiv ist. Um diese Anforderung zu erfüllen, setzt das eigens entwickelte Meßwerkzeug (s. Abschnitt 4.2.5) Synchronisationsmechanismen ein. In den Abschnitten 4.3 und 4.4 wurde anschließend die Performanz der Firewall-Komponenten Packet Screen und Proxy gemessen.

Beide Firewall-Komponenten – Packet Screen und Proxy – führen beim Einsatz in Hochgeschwindigkeitsnetzen zu Engpässen. Da die Steigerungen der Leistungsfähigkeit von Netzen zur Zeit weitaus höher sind als die Steigerungen der Performanz bei den für Firewalls eingesetzten Workstations, muß für die Zukunft noch mit einer deutlichen Verschärfung der Problematik gerechnet werden.

Engpässe bei Packet Screens: Der maximale Datagrammdurchsatz eines 155 MBit/s ATM-Netzes liegt bei fast 200.000 Datagrammen pro Sekunde, wenn sehr kurze Dateneinheiten verwendet werden. Bei der im Testnetz eingesetzten Packet Screen ("IP-Filter") lag der Datagrammdurchsatz selbst ohne die Konfiguration von Filterregeln deutlich unterhalb von 10.000 Datagrammen pro Sekunde. Da der Verarbeitungsaufwand von der Anzahl der Filterregeln abhängt, die vor der Übertragung eines Datagramms von der Packet Screen überprüft werden müssen,

sinkt durch die Konfiguration von Filterregeln der Datagrammdurchsatz weiter ab. Der Datagrammdurchsatz reicht dann nur noch bei Datagrammlängen oberhalb von 2 KByte für das Erreichen des Netzdurchsatzes aus. Da auch in ATM-Backbones kaum Datagrammlängen oberhalb von 1500 Byte (MTU von Ethernet) auftreten und die durchschnittliche Datagrammlänge im Internet sogar noch unterhalb von 500 Byte liegt, werden über die Packet Screen aufgrund des eingeschränkten Datagrammdurchsatzes nur etwa 30 bis 40 MBit/s erreicht. In einem 155 MBit/s CLIP-Netz wäre bei einer Datagrammlängen von 500 Byte theoretisch immer noch ein Datendurchsatz von 120 MBit/s erreichbar. Um diesen Datendurchsatz erreichen zu können, müßte der Datagrammdurchsatz der Packet Screen mindestens vervierfacht werden.

Engpässe bei Proxy-Servern: Bei Proxies treten noch sehr viel schwerwiegendere Engpässe auf als bei Packet Screens. Für Benutzer, die über einen Proxy kommunizieren, werden die Performanzauswirkungen des Proxies über die Verlängerung der Verbindungsaufbauzeiten sichtbar. Entscheidend für die Verbindungsaufbauzeiten ist die Methode mit der neue Verbindungen verarbeitet werden. Muß hierfür zunächst ein neuer Prozeß erzeugt werden, ist die Verbindungsaufbauzeit sehr viel höher als bei der Zuweisung einer neuen Verbindung an einen bereits wartenden Thread. Konzeptionsbedingt läßt sich die Erhöhung der Verbindungsaufbauzeit bei Proxies jedoch nicht vollständig vermeiden, da die Verbindungen beim Proxy enden müssen, damit der Proxy Zugriff auf die übertragenen Daten erhält. Ziel muß daher die Vermeidung des wiederholten Verbindungsauf- und -abbaus sein. Dies macht neue Protokolle erforderlich, die die Verbindung zwischen Client und Proxy geöffnet halten, auch wenn die Verbindung zwischen Proxy und Server abgebaut wird.

Trotz der Erhöhung der Verbindungsaufbauzeit braucht ein Proxy nicht zum Engpaß zu werden, wenn die Zugriffe ausreichend vieler Clients gleichzeitig bearbeitet werden können. Das während des Verbindungsaufbaus ungenutzte Netz kann dann zur Datenübertragung bei parallelen, bereits aufgebauten Verbindungen genutzt werden. Leider ist der Transaktionsdurchsatz von Proxies mit etwa 400 Transaktionen pro Sekunde viel zu gering, um zum Beispiel bei typischen HTTP-Transaktionslängen von 4 bis 8 KByte überhaupt eine nennenswerte Parallelität zu erreichen. Dadurch kann bei Transaktionslängen von 8 KByte der Gesamtdatendurchsatz einen Wert von 20 MBit/s nur knapp überschreiten. Verantwortlich für diesen Engpaß ist die Verbindungsannahme durch den Proxy-Server. Auch durch alternative Konzepte wie die dezentrale Verbindungsannahme durch Arbeitsthreads (s. Abschnitt 4.4.2.6) konnten kaum höhere Transaktionsdurchsätze erreicht werden. Kryptographische Authentisierungsmechanismen wie SSL verschärfen diesen Engpaß erheblich, so daß bei einer Authentisierung mit 1024 Bit langen Schlüsseln nur noch ein Transaktionsdurchsatz von etwa 6

Transaktionen pro Sekunde erreicht wird. Durch die Vermeidung von wiederholtem Verbindungsauf- und -abbau kann eine Entlastung des Proxies erreicht werden. Ein höherer Transaktionsdurchsatz kann durch eine Lastverteilung auf parallele Bastionen erreicht werden.

Der Datendurchsatz eines Proxies wird erst durch den Einsatz aufwendiger "Scanner" oder von Verschlüsselung zum Engpaß. Bei der Beschränkung des Proxies auf eine reine Kopierfunktion zwischen beiden Verbindungen können hohe Datendurchsätze erreicht werden. Jedoch muß einschränkend erwähnt werden, daß die Datendurchsätze mit einer sehr stark nach Performanzgesichtspunkten optimierten Eigenentwicklung ("MGproxy") gemessen wurden. Ebenfalls gemessene Standard-Software wie "Socks" kann diese Durchsätze nicht erreichen. Eine Steigerung des Gesamtdurchsatzes kann durch Lastverteilung auf mehrere Bastionen erreicht werden. Bei "Scannern" sind Steigerungen des Datendurchsatzes durch effizientere Verfahren denkbar. Für kryptographische Verfahren ist in Zukunft der Einsatz von spezialisierter Hardware erforderlich, die einen deutlich höheren Datendurchsatz erzielt und gleichzeitig die CPU entlastet.

Trotz der identifizierten Engpässe ist der Einsatz von Proxies notwendig, da die Filterungsmöglichkeiten von Packet Screens allein selten ausreichen, um eine ausreichende Sicherheit zu erzielen. Erst durch den Einsatz von Proxies wird es möglich, Zugriffe von Benutzern zu authentisieren und die übertragenen Daten auf Anwendungsebene zu filtern.

Weiteres Vorgehen: Für die Entwicklung von Firewalls für Hochgeschwindigkeitsnetze ergeben sich folgende Prioritäten: Konzepte für hochperformante Proxy-Server sind wichtiger als die Steigerungen der Performanz von Packet Screens. Bei Proxy-Servern muß in erster Linie ein hoher Transaktionsdurchsatz erreicht werden. Ebenfalls müssen die Verbindungsaufbauzeiten verringert werden. Die Steigerung des Datendurchsatzes bei "Scannern" und der Einsatz kryptographischer Verfahren ist ebenfalls anzustreben.

Im nächsten Kapitel werden Parallelisierungsverfahren zur Erhöhung der Firewall-Performanz diskutiert.

5 Konzepte paralleler Firewalls und weitere Optimierungsansätze

Inhalt

5.1 Parallelisierungskonzepte 373
5.2 Parallele Bastionen 375
5.2.1 Funktionale Trennung 376
5.2.2 Zentrale Verteilung von Verbindungen 377
5.2.2.1 "Round-Robin"-DNS 377
5.2.2.2 Verteilung durch Proxies 381
5.2.2.3 Verteilung auf Netzwerkebene 382
5.2.2.4 Verteilung durch einen "TCP-Router" 385
5.2.2.5 Verteilung über virtuelle Verbindungen 388
5.2.2.6 Verteilung durch NHRP 390
5.2.2.7 Signalisierungskontroller 393
5.2.3 Dezentrale Verteilung von Verbindungen 395
5.2.3.1 Zufällige Auswahl einer Bastion 395
5.2.3.2 Transparentes Umleiten von HTTP-Verbindungen . . 397
5.2.3.3 Transparentes Umleiten von "Socks"-Verbindungen . 398
5.2.3.4 Filtern von Datagrammen 401
5.3 Parallele Packet Screens 404
5.4 Verringerung der Verbindungsaufbauzeiten 411
5.4.1 Offenhalten von Verbindungen und "Pipelining" 412
5.4.2 Leichtgewichtiges TCP 413
5.4.3 "Multiplexing" von Verbindungen 415
5.5 Abschließende Bewertung 417

In diesem Kapitel werden Konzepte vorgestellt, mit denen die im vorigen Kapitel aufgezeigten Engpässe von Firewalls in Hochgeschwindigkeitsnetzen beseitigt werden können. Die Mehrzahl der vorgestellten Konzepte basiert auf der parallelen Bearbeitung auf replizierten Stationen. Unabhängig von Parallelisierungskonzepten kann die Performanz auch durch Einsatz optimierter Protokolle gesteigert werden.

Bei der im vorigen Kapitel untersuchten Performanz von Firewalls zeigte sich, daß bei Packet Screens und Proxies unterschiedliche Engpässe bestehen. Maßnahmen zur Steigerung der Performanz müssen daher bei Packet Screens und Proxies unterschiedliche Ziele verfolgen. Bei Proxies sind die Steigerung des Transaktions- und Datendurchsatzes sowie die Verringerung der Verbindungsaufbauzeit die wichtigsten Optimierungsziele. Bei Packet Screens, die einen geringeren Einfluß auf die Performanz haben als Proxies, ist die Steigerung des Datagrammdurchsatzes am wichtigsten. Daneben werden zukünftig die steigenden Durchsätze in Hochgeschwindigkeitsnetzen auch eine Steigerung des Datendurchsatzes von Packet Screens erforderlich machen.

Eine weitere Möglichkeit zur Steigerung der Performanz ist der Einsatz spezialisierter Hardware. Zwar ist spezialisierte Hardware für Teilaufgaben (Filtern von Paketen und Verschlüsselung) sehr viel besser geeignet als entsprechende Software-Implementationen, die auf Allzweck-Computern (PCs, Workstations) eingesetzt werden. Jedoch sind Lösungen, die auf spezialisierter Hardware basieren, weniger flexibel und können daher nicht alle Aufgaben eines Firewalls übernehmen. Dies gilt insbesondere für die Kontrolle auf Anwendungsebene durch Proxies oder Anwendungs-Gateways, die den wichtigsten Performanzengpaß darstellt. Daher werden Lösungen, die spezialisierte Hardware zur Steigerung der Performanz einsetzen, in diesem Kapitel nicht weitergehend untersucht.

Nach einer kurzen allgemeinen Vorstellung von Parallelisierungskonzepten im Abschnitt 5.1 werden in den darauffolgenden Abschnitten Parallelisierungskonzepte zur Steigerung der Performanz von Proxies (Abschnitt 5.2) und von Packet Screens (Abschnitt 5.3) untersucht. Die Untersuchung von Parallelisierungskonzepten für Firewalls orientiert sich zunächst an Konzepten, die bereits in anderen Einsatzbereichen (z. B. Lastverteilung bei Web-Servern) angewendet werden, und überprüft ihre Eignung für die Parallelisierung von Firewalls. Darüber hinaus werden neue Ansätze zur Parallelisierung von Firewalls vorgestellt und bewertet. Durch Parallelisierungskonzepte kann bei Proxies eine Steigerung des Transaktions- und des Datendurchsatzes erreicht werden. Die Verbindungsaufbauzeiten bei Proxies können durch dynamische Methoden der Parallelisierung jedoch praktisch nicht verringert werden. Abschnitt 5.4 zeigt verschiedene Möglichkeiten zur Verringerung der Verbindungsaufbauzeiten bei Proxies auf. Abschnitt 5.5 faßt schließlich die Ergebnisse dieses Kapitels zusammen.

5.1 Parallelisierungskonzepte

Die Mehrzahl heutiger Firewall-Architekturen basiert auf Workstations, da diese flexibler für die Zugriffskontrolle eingesetzt werden können als spezialisierte Hardware. Trotz der stetigen Leistungssteigerung bei Workstations bilden Firewalls in Hochgeschwindigkeitsnetzen einen Engpaß, da die Netzdurchsätze von Hochgeschwindigkeitsnetzen sehr viel stärker steigen als die Leistungsfähigkeit von Workstations. Die unzureichende Leistungsfähigkeit von Workstations ist dabei kein Firewall-spezifisches Problem, denn auch in anderen Einsatzbereichen verhindert die mangelnde Leistungsfähigkeit von Workstations ein Ausnutzen des in Hochgeschwindigkeitsnetzen zur Verfügung stehenden Netzdurchsatzes [Zitterbart 91]. Um trotzdem auch mit herkömmlichen Workstations die Leistungsfähigkeit von Hochgeschwindigkeitsnetzen ausnutzen zu können, werden Parallelisierungskonzepte als vielversprechende Ansätze diskutiert.

Bei Parallelisierungskonzepten wird grundsätzlich zwischen statischen und dynamischen Methoden unterschieden. Statische Methoden versuchen, die inhärente Parallelität, auch als "strukturelle Parallelität" [Zitterbart 91] bezeichnet, innerhalb eines Protokollstapels auszunutzen. Mit verschiedenen "Pipeline"-Konzepten kann bei statischen Methoden, beispielsweise während der Verarbeitung einer empfangenen Dateneinheit auf Anwendungsebene durch die CPU der Workstation, gleichzeitig ein auf der Netzwerkschnittstelle integrierter Prozessor bereits die Prüfsumme der folgenden Dateneinheit überprüfen. In gleicher Weise kann auch die Ver- und Entschlüsselung von Dateneinheiten mit Hilfe von "Pipeline"-Konzepten parallel zur übrigen Bearbeitung erfolgen. Auch das Senden und Empfangen von Dateneinheiten kann unabhängig voneinander erfolgen. Die erforderliche Parallelisierung der Algorithmen, die zur Protokollverarbeitung eingesetzt werden, ist sehr aufwendig und kann zudem leicht zu Fehlern führen. Um einen hohen Grad an Parallelität zu erreichen, ist bei statischen Methoden ein hoher Implementationsaufwand erforderlich.

Bei dynamische Methoden werden einzelne Datagramme oder einzelne Verbindungen parallel zu anderen Datagrammen oder Verbindungen bearbeitet. Allgemein, unabhängig vom Anwendungsbereich betrachtet, werden bei dynamischen Methoden *Aufträge* (z. B. Datagramme, Verbindungen, HTTP-Anfragen usw.) auf parallele *(Bedien-)Stationen* (z. B. Packet Screens, Proxies, Web-Server usw.) verteilt. Die parallele Verarbeitung von Datagrammen bzw. Paketen wird als Paketparallelität ("Packet-level Parallelism") [Nahum et al. 94] bezeichnet und eignet sich besonders für die Parallelisierung von Packet Screens, da Packet Screens einzelne Datagramme unabhängig voneinander verarbeiten. Firewall-Komponenten auf Anwendungsebene wie Proxies und Anwendungs-Gateways müssen grundsätzlich alle zu einer Verbindung gehörenden Datagramme verarbeiten, so daß die parallele Verarbeitung von Datagrammen für diese Kompo-

nenten kaum geeignet ist. Für die Parallelisierung von Proxies und Anwendungs-Gateways ist die als Verbindungsparallelität ("Connection-level Parallelism") bezeichnete parallele Verarbeitung von Verbindungen besser geeignet. Dynamische Methoden haben gegenüber statischen Methoden den Vorteil, daß sich herkömmliche (sequentielle) Algorithmen sehr viel leichter an die Parallelverarbeitung anpassen lassen. Der erreichbare Grad der Parallelität hängt bei dynamischen Methoden von den verarbeiteten Datagrammen bzw. von den parallel geöffneten Verbindungen ab. Wie gut diese erreichbare Parallelität von einem Verteilungsverfahren ausgenutzt wird, hängt von der verwendeten *Verteilungsfunktion* ab, die eingehende Aufträge parallelen Prozessoren zuteilt.

Dynamische Methoden werden unterschieden nach lastabhängigen und lastunabhängigen Verteilungsverfahren. Lastunabhängige Verfahren verteilen eingehende Verbindungen oder Datagramme nach einem vorher festgelegten Verteilungsschlüssel. Beispielsweise werden beim "Round-Robin"-DNS-Verfahren (s. Abschnitt 5.2.2.1) anfragenden Clients nacheinander an unterschiedliche Server verwiesen, ohne daß die aktuelle Last der einzelnen Server hierbei berücksichtigt wird. Dies kann dazu führen, daß sich die aufwendigeren Aufträge bei einem Server häufen, während sich andere Server im "Leerlauf" befinden. Durch eine lastabhängige Verteilung kann diese ungleichmäßige Verteilung verhindert werden. Lastabhängige Verteilungsverfahren erfordern jedoch eine Erfassung und Übermittlung von Lastdaten an den Verteiler. Dadurch sind lastabhängige Verteilungsverfahren aufwendiger zu implementieren als lastunabhängige Verfahren.

Die dynamischen Methoden können weiter unterschieden werden zwischen einer zentralen oder dezentralen Verteilung von Verbindungen bzw. Datagrammen. Bei der zentralen Verteilung entscheidet eine einzelne Verteilungskomponente darüber, durch welche Station die Verarbeitung erfolgen soll. Ein Beispiel hierfür ist zum Beispiel die Verteilung über "Network Address Translation", wie in Abschnitt 5.2.2.3 beschrieben. Der Vorteil zentraler Verteilungsverfahren ist die einfachere Implementation lastabhängiger Verteilungen. Der Nachteil zentraler Verteilungsverfahren ist, daß die Performanz und Funktionsfähigkeit von dem zentralen Verteiler abhängt. Neben einem potentiellen Engpaß stellt der zentrale Verteiler auch eine zentrale Fehlerquelle dar. Ein wichtiges Merkmal ist bei zentralen Verteilungsverfahren außerdem, ob der Verteiler an der anschließenden Datenübertragung beteiligt ist. Die Verteilung über "Network Address Translation" (s. Abschnitt 5.2.2.3) erfordert beispielsweise das Ersetzen der IP-Adressen bei allen zwischen Client und Server ausgetauschten Datagrammen. Hierdurch kann der Verteiler selbst zum Engpaß werden. Demgegenüber braucht ein Signalisierungskontroller (s. Abschnitt 5.2.2.7) lediglich einmalig zu Beginn, bei der Signalisierung zum Aufbau einer virtuellen Verbindung, einzugreifen. Die nachfolgende Kommunikation über die virtuelle Verbindung erfordert keine weiteren Eingriffe des Signalisierungskontrollers.

Bei dezentralen Verteilungsverfahren entscheidet jede einzelne parallele Station selbst, ob die Verarbeitung durch diese Station erfolgen soll. Ein Beispiel für ein dezentrales Verteilungsverfahren ist die Auswahl von zu übertragenden Datagrammen bei der in Abschnitt 5.3 vorgestellten parallelen Packet Screen. Da bei dezentralen Verteilungsverfahren kein zentraler Engpaß existiert, skalieren diese Verfahren in der Regel besser als zentrale Verteilungsverfahren.

In den folgenden Abschnitten 5.2 und 5.3 wird die Anwendung dynamischer Methoden für die Parallelisierung von Packet Screens und Proxies untersucht. Dabei werden zunächst Parallelisierungsansätze aus anderen Anwendungsbereichen auf Firewalls übertragen. Darüber hinaus werden Parallelisierungsansätze speziell für Firewalls entwickelt. Für jeden Ansatz werden die Vor- und Nachteile untersucht.

5.2 Parallele Bastionen

Firewall-Komponenten auf Anwendungsebene — Proxies und Anwendungs-Gateways — werden auf Bastionen eingesetzt. Für die Parallelisierung dieser Firewall-Komponenten kommen somit Konzepte zum Einsatz, mit denen Verbindungen auf mehrere parallele Bastionen[1] verteilt werden. Ziel der Parallelisierung muß eine Erhöhung des Transaktionsdurchsatzes und des Datendurchsatzes von Proxies und Anwendungs-Gateways sein. Idealerweise sollten die Parallelisierungsansätze eine skalierbare Leistungssteigerung ermöglichen, so daß auch zukünftige Durchsatzsteigerungen bei Hochgeschwindigkeitsnetzen keine neuen Parallelisierungslösungen erforderlich machen. Eine Verringerung der Verbindungsaufbauzeiten, die ebenfalls zur Steigerung der Performanz von Proxies erforderlich ist, kann durch dynamische Methoden zur Parallelisierung nur teilweise erreicht werden, da prinzipbedingt bei einem Proxy zwei Verbindungen (Client→Proxy und Proxy→Server) aufgebaut werden müssen. Zur Verringerung der Verbindungsaufbauzeiten sind Protokolloptimierungen besser geeignet (s. Abschnitt 5.4).

Die meisten der nachfolgenden Konzepte beziehen sich auf IP-Netze und können daher gleichermaßen für ATM-basierte IP-Netze wie für herkömmliche IP-Netze eingesetzt werden. Einzelne Verfahren erfordern hingegen ATM-spezifische Komponenten wie z. B. einen Signalisierungskontroller (s. Abschnitt 5.2.2.7) und sind daher nicht in herkömmlichen Netzen einsetzbar.

[1] In der folgenden Diskussion wird von der Aufteilung auf *Bastionen* gesprochen. Da es sich bei Gateway-Firewalls nach der Definition auch um Bastionen handelt, wurde darauf verzichtet, jeweils explizit die Anwendbarkeit der einzelnen Lösungen auf Gateway-Firewalls zu erwähnen. Wo sich die Lösungen für Gateway-Firewalls und Kombinationen aus Packet Screen und Bastion unterscheiden, werden die Unterschiede diskutiert.

5.2.1 Funktionale Trennung

Die funktionale Trennung ist die einfachste Form, eine Last auf mehrere Systeme zu verteilen. Dabei werden die Proxies für die unterstützten Dienste (HTTP, FTP, ...) so auf die Bastionen verteilt, daß eine möglichst gleichmäßige Verteilung der Last erreicht wird. Denkbar ist zum Beispiel der Einsatz einer dedizierten Bastion für HTTP, während die übrigen Dienste über die zweite Bastion abgewickelt werden. Da der Anteil eines Dienstes an der Gesamtlast vom Kommunikationsverhalten der Benutzer abhängig ist, sind Messungen und Analysen erforderlich, um eine günstige Verteilung zu ermitteln. Diese statische Zuordnung unterschiedlicher Dienste zu unterschiedlichen Bastionen ist der bisher am häufigsten angewendete Ansatz zur Parallelisierung von Firewalls.

Übersteigt die Last eines Dienstes allein bereits die Leistungsfähigkeit einer Bastion, können getrennte Bastionen für hereinkommenden und herausgehenden Verkehr eingesetzt werden. Durch den Einsatz jeweils einer Bastion für unterschiedliche interne Subnetze wird die maximale Lastverteilung erreicht, die mit diesem Konzept erreichbar ist.

Eine funktionale Trennung ist nicht nur für die Lastverteilung, sondern auch vom Sicherheitsstandpunkt her sinnvoll. Ein Angriff auf einen einzelnen Dienst hat keinen direkten Einfluß auf die sichere Abwicklung der anderen Dienste. Blockiert bei Kombinationen aus Packet Screen und Bastion noch eine Packet Screen den direkten Zugriff auf das zu schützende Netz, so ist die Gefährdung des internen Netzes nach einem erfolgreichen Angriff auf eine Bastion geringer, da die Packet Screen dieser Bastion nur Zugriffe auf den Dienst gewähren wird, für den die Bastion eingesetzt wurde. Wenn alle Dienste auf einer Bastion konzentriert sind, kann der Angreifer auch alle Dienste für ein weiteres Eindringen in das zu schützende Netz nutzen.[2]

Der wesentliche Nachteil dieser Lösung ist die statische Zuordnung der Dienste. Sobald das tatsächliche Verkehrsaufkommen von der vorher ermittelten Verteilung abweicht, führt dies zu einer ungünstigen Lastverteilung — im Extremfall muß eine Bastion den gesamten Verkehr übertragen, während die anderen parallelen Bastionen sich im "Leerlauf" befinden. Zu solchen Situationen kann es schnell kommen, wenn die zu übertragenden Datenmengen auf kurzen Zeiträumen konzentriert von verschiedenen Sendern auftreten oder wenn z. B. tagsüber eine andere Verkehrsaufteilung vorherrscht als nachts. Um auch in solchen Situationen eine gleichmäßige Verteilung zu erreichen, ist eine dynamische Aufteilung der Verkehrsflüsse (Verbindungen) während der Laufzeit erforderlich, die dieses Konzept nicht bieten kann.

[2] Dieser Vorteil der funktionalen Trennung der Dienste trifft für Gateway-Firewalls nicht zu.

5.2.2 Zentrale Verteilung von Verbindungen

In diesem Abschnitt werden Konzepte vorgestellt, bei denen ein zentraler Verteiler eingehende Verbindungen an parallele Bastionen zuweist. Diese Zuweisung kann direkt auf eine eingehende Verbindung wirken oder indirekt — z. B. durch Eingriff in DNS-Informationen — einen Client dazu veranlassen, Verbindungen zu einer bestimmten Bastion aufzubauen. Heute werden zentrale Verteilungsverfahren für Zugriffe auf parallele Web-Server eingesetzt. Dieser Abschnitt wird sich zunächst an den für Web-Server eingesetzten Verfahren orientieren und die Anwendbarkeit dieser Verfahren für die Verteilung von Verbindungen auf parallele Bastionen untersuchen. Darüber hinaus werden in diesem Abschnitt neue ATM-spezifische Verteilungsverfahren untersucht.

5.2.2.1 "Round-Robin"-DNS

Die am häufigsten angewendete Methode der Lastverteilung bei Web-Servern ist die "Round-Robin"-Ergänzung im DNS. Bevor ein Client eine Verbindung zu einem Web-Server aufbauen kann, muß er zunächst den Namen des Web-Servers (z. B. `www.fwl.dfn.de`) in dessen IP-Adresse umwandeln. Ist die IP-Adresse nicht bereits innerhalb des Client-Systems gespeichert, muß die Adresse über eine DNS-Anfrage ermittelt werden. Der Client öffnet anschließend eine Verbindung zu der vom DNS-Server angegebenen IP-Adresse. Durch die Auswahl entsprechender IP-Adressen hat der DNS-Server die Möglichkeit, die Verbindungen von verschiedenen Clients an unterschiedliche Web-Server zu dirigieren.

Beim "Round-Robin"-DNS werden die Namen der replizierten Web-Server, auf die die Last verteilt werden soll, als `CNAME` (oder alternativ deren IP-Adressen als "`A`-Record") für die virtuelle Adresse "www" eingetragen (s. Abbildung 5.1). Nach jeder Anfrage für "www" rotiert der DNS-Server die Liste der `CNAME`s, so daß aufeinander folgende Anfragen gleichmäßig auf die zur Verfügung stehenden Web-Server verteilt werden [Brisco 95, Engelschall 98]. Da als `CNAME`s beliebige Adressen verwendet werden können, ist es möglich, hier auch die Adressen räumlich entfernter Systeme zu verwenden. Die replizierten Server müssen sich nicht wie bei vielen anderen Verteilungsverfahren in unmittelbarer räumlicher Nähe befinden.

Diese Methode kann auch für eine Verteilung auf mehrere parallele Bastionen verwendet werden. Eine Reihe von Voraussetzungen schränken jedoch die Anwendbarkeit dieser Lösung ein. "Round-Robin" wird nicht von allen DNS-Server-Implementationen unterstützt. Für den Einsatz dieser Lösung müssen jedoch alle DNS-Server der DNS-Domäne (der "Primary"-DNS-Server und alle "Secondary"-DNS-Server) "Round-Robin" unterstützen.

```
www     CNAME    www1.fwl.dfn.de.
        CNAME    www2.fwl.dfn.de.
        CNAME    www3.fwl.dfn.de.
        CNAME    www4.fwl.dfn.de.
www1    IN   A   192.168.20.1
www2    IN   A   192.168.20.2
www3    IN   A   192.168.20.3
www4    IN   A   192.168.20.4
```

Abbildung 5.1: Konfiguration für "Round-Robin"-DNS

Ein weitere wichtige Einschränkung entsteht durch das Zwischenspeichern ("Caching") der Ergebnisse von DNS-Anfragen beim lokalen DNS-Server sowie im Betriebssystem des Client-Rechners. Bei der ersten Anfrage nach einer IP-Adresse speichert der angefragte, lokale DNS-Server die Antwort des DNS-Servers der entfernten Domäne, weitere DNS-Anfragen werden dann aus dem "Cache" beantwortet. Dies hat zur Folge, daß für alle Clients, die Antworten aus dem "Cache" erhalten, keine weitere Verteilung nach dem "Round-Robin"-Verfahren stattfindet.[3] Zusätzlich speichern auch einige Betriebssysteme ebenfalls die Antworten von DNS-Servern für kurze Zeit (z. B. durch den `nscd` bei "Sun Solaris" Systemen). Schließlich führen auch die Client-Prozesse (z. B. ein "Browser") nicht vor jedem Verbindungsaufbau eine DNS-Anfrage durch. Für das Laden von innerhalb von HTML-Seiten enthaltene Abbildungen oder für nachfolgende HTML-Seiten wird typischerweise keine erneute DNS-Anfrage durchgeführt, wenn diese Abbildungen und weiteren HTML-Seiten ebenfalls auf demselben Web-Server gespeichert sind. Das Zwischenspeichern von DNS-Informationen und die mehrfache Nutzung durch Client-Prozesse führen dazu, daß nur ein kleiner Teil der Verbindungen durch den DNS-Server über "Round-Robin" auf parallele Server verteilt werden kann.

[3] Die bei der Zwischenspeicherung von einem DNS-Server angewendete Strategie ist implementationsabhängig. Eine typische Implementation eines DNS-Servers (Bind v8.1.2) führt beispielsweise jede Stunde eine "Garbage Collection" durch, bei der ungenutzte und alte Einträge entfernt werden. Zwar kann im "Start of Authority" (SOA) die Dauer festgelegt werden, für die DNS-Informationen zwischengespeichert werden dürfen ("Time To Live", TTL), allerdings werden hier üblicherweise Zeiträume von Stunden oder Tagen angegeben. Um die zu häufige Verwendung von zwischengespeicherten Informationen zu verhindern, wäre jedoch ein Zeitraum im Bereich von Minuten oder Sekunden erforderlich. Bei der Reduktion der TTL gibt es zudem zwei wesentliche Einschränkungen. Zum einen steigt bei einer niedrigeren TTL die Anzahl der DNS-Anfragen, so daß der Vorteil der besseren Verteilungsmöglichkeit gegen das höhere Verkehrsaufkommen durch DNS-Anfragen abgewogen werden muß [Colajanni et al. 97]. Außerdem gelten die reduzierten TTL Zeiträume für alle Einträge der DNS-Domäne, so daß die statischen Einträge der übrigen Systeme ebenfalls nur kurzzeitig zwischengespeichert werden dürfen.

Beim Einsatz von "Round-Robin"-DNS zur Lastverteilung zwischen parallelen Bastionen kann sich das Zwischenspeichern von DNS-Informationen hingegen als Vorteil erweisen. Zwar wird auch hier die Verteilung durch die Zwischenspeicherung gestört, aber auf der anderen Seite ist es für die Kontrolle des Firewalls *erforderlich*, daß alle Zugriffe desselben Clients über dieselbe Bastion geführt werden, um Abhängigkeiten zwischen verschiedenen Verbindungen bei der Kontrolle berücksichtigen zu können. Wird hingegen durch "Round-Robin"-DNS für eine nachfolgende Verbindung desselben Clients eine andere Bastion ausgewählt, so kann diese Bastion die Zugriffskontrolle nicht aus dem Kontext der vorigen Verbindungen heraus durchführen.

Auch der bevorstehende Einsatz von "Secure DNS" [Eastlake et al. 97] stellt ein Hindernis für diese Lösung dar, weil "Secure DNS" bisher nicht im "Round-Robin"-Modus betrieben werden kann. Bei "Secure DNS" werden die in einer DNS-Domäne gespeicherten Daten mit Authentisierungsinformationen versehen, so daß die Authentizität der von einem DNS-Server gelieferten Daten von einem Client oder einem anderen DNS-Server überprüft werden kann. Die Authentisierungsinformationen werden im voraus ("offline") erzeugt, damit der DNS-Server während des Betriebs keine zeitaufwendigen Berechnungen der Authentisierungsinformationen durchführen muß und damit durch einen Angriff auf den DNS-Server der Schlüssel zum Erzeugen der Authentisierungsinformationen nicht kompromittiert wird. Für die Erzeugung von Authentisierungsinformationen muß die Reihenfolge der im DNS gespeicherten Einträge festgelegt werden.[4] Da diese Reihenfolge beim "Round-Robin"-Modus ständig verändert wird, kann der "Round-Robin"-Modus nicht zusammen mit "Secure DNS" eingesetzt werden.

Die wichtigste Einschränkung dieses Ansatzes ist aber, daß bei "Round-Robin"-DNS keine Lastverteilung, sondern eine Verteilung der Zugriffe realisiert wird. Durch die Verteilung der Zugriffe wird die Last aber nur dann gleichmäßig verteilt, wenn alle Zugriffe auch eine gleich hohe Last erzeugen. Dies bedeutet zum Beispiel bei WWW, daß alle Dokumente etwa gleich lang sein müssen. Werden hingegen bei FTP auf einzelnen Verbindungen mehrere Megabyte an Daten übertragen, während andere Verbindungen bereits nach wenigen Byte abgebrochen werden, führt "Round-Robin"-DNS zu einer weniger optimalen Verteilung der Last. Das fehlende "Feedback" der replizierten Server an den DNS-Server verhindert nicht nur eine lastabhängige Verteilung, sondern führt auch dazu, daß der

[4] Eine Festlegung der Reihenfolge ist erforderlich, um Aussagen über nicht vorhandene Einträge authentisieren zu können. Anstatt auf die Anfrage nach Adresse B mit der Aussage "Adresse B existiert nicht" zu antworten, wird vom DNS-Server die Antwort "Nach Adresse A folgt Adresse C" geliefert. Da nur "A" und "C" im voraus bekannt sind und die vom Client angefragte Adresse "B" nicht vorhersehbar ist, kann nur die Antwort "Nach Adresse A folgt Adresse C" im voraus mit Authentisierungsinformationen versehen werden.

Ausfall einzelner Server nicht erkannt wird und "Round-Robin"-DNS weiterhin auch die Adressen von ausgefallenen Servern auswählt und an Clients sendet.

"Round-Robin"-DNS wird primär zur Verteilung von Zugriffen auf parallele Web-Server verwendet. Verbindungen, die über einen Firewall geführt werden, haben im Gegensatz zu WWW-Anfragen sehr unterschiedliche Charakteristiken. Während WWW-Anfragen meist nur wenige Sekunden dauern, können Verbindungen über einen Firewall von wenigen Sekunden bis zu mehreren Stunden andauern. Auch die Last, die einzelne Verbindungen auf einem Firewall erzeugen, kann sich erheblich unterscheiden. Dies ist z. B. davon abhängig, ob der Einsatz von "Scannern" oder von Verschlüsselung erforderlich ist (s. Abschnitt 4.4.2.8). Eine lastunabhängige Verteilung von Verbindungen durch "Round-Robin"-DNS ist daher für Firewalls nur in Ausnahmefällen sinnvoll.

"Round-Robin"-DNS-Variante mit lastabhängiger Verteilung: Um die Verteilung von der Last abhängig zu machen, muß in kurzen Abständen die Last abgefragt werden. Anhand diesen Daten muß anschließend der DNS-Server umkonfiguriert werden, so daß ein größerer Anteil der Anfragen an weniger ausgelastete Server weitergeleitet wird. Durch Umkonfigurieren können auch ausgefallene Server zeitweise aus der Verteilung herausgenommen werden, damit keine weiteren Zugriffe mehr zu den ausgefallenen Servern dirigiert werden.

In [Colajanni et al. 97] werden unterschiedliche Strategien untersucht, mit denen eine gleichmäßigere Lastverteilung als mit dem herkömmlichen "Round-Robin"-DNS erreicht wird. Simulationen der unterschiedlichen Strategien zeigen, daß durch ein Einbeziehen detaillierter Lastinformationen der parallelen Server keine gleichmäßigere Lastverteilung erzielt werden kann als durch einfachere Verfahren, bei denen die Server lediglich eine Überlastsituation an den DNS-Server melden. Die Ursache hierfür liegt bei der bereits beschriebenen Zwischenspeicherung von DNS-Informationen. Da wegen der Zwischenspeicherung nur ein sehr kleiner Teil aller Verbindungen über den DNS-Server verteilt werden kann, führen die ermittelten Lastinformationen oft zu einer falsche Einschätzung der zukünftigen Last des jeweiligen Servers.

Ein kontinuierliches Umkonfigurieren von DNS-Servern ist im DNS-Konzept nicht vorgesehen. Als schwierig erweist sich insbesondere, die erfolgten Änderungen auch in die "Secondary"-DNS-Server einzubringen. Selbst wenn dieses Problem gelöst werden kann, wird die Lastverteilung immer noch durch das Zwischenspeichern von DNS-Antworten gestört, so daß eine lastabhängige Verteilung über DNS nur für spezielle, eingeschränkte Probleme einsetzbar ist.

Fazit: Der Einsatz von DNS zur Verteilung stellt eine einfache Möglichkeit für die Verteilung von Zugriffen auf parallele Web-Server dar. Die mit diesem

Konzept verbundenen Einschränkungen können insbesondere beim Einsatz zur Verteilung von Zugriffen auf parallele Bastionen zu einer sehr ungleichmäßigen Lastverteilung führen. Die Berücksichtigung von Lastinformationen bei der Verteilung ist zwar prinzipiell möglich, kann aber wegen der Zwischenspeicherung von DNS-Informationen kaum bessere Ergebnisse liefern als eine lastunabhängige "Round-Robin"-Verteilung.

5.2.2.2 Verteilung durch Proxies

Eine deutlich gleichmäßigere Lastverteilung als bei "Round-Robin"-DNS kann durch Proxies erreicht werden. Alle Zugriffe auf einen Dienst müssen dabei über einen zentralen Proxy erfolgen. Der Proxy führt dann seinerseits Zugriffe auf einen der parallelen Server durch. Der Proxy berücksichtigt bei der Auswahl eines Servers die aktuelle Last der parallelen Server. Die dafür benötigten Lastinformationen kann der Proxy entweder direkt bei den Servern abfragen, oder die Lastinformationen können vom Proxy indirekt zum Beispiel aus der Anzahl der Verbindungen zum jeweiligen Server oder über den Datendurchsatz, der mit dem jeweiligen Server zur Zeit erzielt wird, geschätzt werden.

[Engelschall 98] schlägt einen "Reverse Proxy"[5] als zentralen Ansprechpunkt für Verbindungen zu mehreren parallelen Web-Servern vor. Beim Einsatz zur Parallelisierung von Firewalls würde dieser Proxy als zentraler Verteiler eine lastabhängige Verteilung der Zugriffe auf parallele Bastionen durchführen. Erst auf den Bastionen finden dann rechenintensive (Sicherheits-)Überprüfungen, wie Authentisierung, Entschlüsselung oder Virenkontrolle statt.

Trotz der sehr guten Verteilungsmöglichkeit durch die sehr genaue Ermittlung der aktuellen Last der einzelnen Server, kann der Einsatz eines Proxies zur Lastverteilung nicht überzeugen, da bereits einfachste Proxies in Hochgeschwindigkeitsnetzen zum Engpaß hinsichtlich des Transaktionsdurchsatzes und des Datendurchsatzes werden (s. Abschnitt 4.4). Zudem steigen durch den zusätzlichen Verbindungsaufbau zwischen dem die Verteilung durchführenden Proxy und dem Web-Server die Verbindungsaufbauzeiten für den Client. Somit kann bei diesem Ansatz zwar eine sehr gute Lastverteilung erreicht werden, aber der verteilende Proxy stellt selbst bereits einen Engpaß dar. Der Einsatz dieses Konzeptes ist daher nur in sehr wenigen Einsatzbereichen sinnvoll, bei denen auf den Servern aufwendige Prozesse ausgeführt werden müssen und der Proxy im Verhältnis zu den rechenintensiven Operationen auf den Servern nicht zum Engpaß werden kann.

[5] Die Bezeichnung "Reverse Proxy" wurde von [Engelschall 98] gewählt, weil der Proxy nicht wie beim Haupteinsatzbereich von HTTP-Proxies die Zugriffe mehrerer Clients bündelt, um Anfragen auf dieselbe Web-Seite aus dem "Cache" bedienen zu können. Stattdessen werden beim "Reverse Proxy" Zugriffe auf parallele Web-Server aufgeteilt.

5.2.2.3 Verteilung auf Netzwerkebene

Der im vorigen Abschnitt diskutierte Ansatz, einen Proxy als zentralen Verteiler einzusetzen, ermöglicht durch die damit realisierte zentrale Verteilung zwar eine sehr gleichmäßige Lastverteilung, wird aber aufgrund des hohen Verarbeitungsaufwands im Proxy selbst zum Engpaß. Ein geringer Verarbeitungsaufwand entsteht, wenn die eingehenden Dateneinheiten durch den Verteiler statt auf Anwendungsebene im Proxy bereits auf Netzwerkebene verarbeitet werden. Ein vielversprechender Ansatz ist daher die Lastverteilung auf Netzwerkebene durch den Einsatz von "Network Address Translation" (NAT).

NAT [Egevang et al. 94] wird heute üblicherweise in Verbindung mit einer Packet Screen eingesetzt, um IP-Adressen bei der Verarbeitung durch die Packet Screen auf andere IP-Adressen abzubilden. Transparent für die kommunizierenden Endsysteme ersetzt die NAT-Komponente abhängig von der Übertragungsrichtung die Absender- oder die Empfänger-IP-Adresse übertragener Dateneinheiten. Dadurch ist es für Systeme, deren Adressen nur innerhalb eines bestimmten Subnetzes Gültigkeit haben, möglich, nach außen hin andere Adressen zu verwenden. Dies ist zum Beispiel notwendig, wenn Systeme mit Adressen aus einem privaten Adreßbereich (siehe [Rekhter et al. 96]) mit dem Internet kommunizieren sollen.

Bei der Lastverteilung über NAT [Srisuresh et al. 98] senden Clients IP-Datagramme an eine virtuelle Server-Adresse. Die NAT-Komponente auf dem Pfad zu dieser virtuellen Adresse ersetzt die virtuelle Server-Adresse durch die Empfänger-IP-Adresse des Servers, der das IP-Datagramm bearbeiten soll (s. Abb. 5.2). Die Auswahl des Servers, der das IP-Datagramm bearbeiten soll, kann wie beim Einsatz eines Proxies zur Lastverteilung (s. o.) beispielsweise an den Server mit der derzeit niedrigsten Last erfolgen. Wichtig ist bei der Auswahl des Servers jedoch, daß alle IP-Datagramme, die zu derselben TCP-Verbindung gehören, auch von demselben Server verarbeitet werden müssen. Verschiedene Verbindungen von demselben Client können jedoch an unterschiedliche Server weitergeleitet werden.

Im Gegensatz zur Verteilung über einen Proxy muß die NAT-Komponente lediglich einfache Regeln verwalten, um alle zu derselben TCP-Verbindung gehörenden IP-Datagramme an denselben Server weiterleiten zu können. Da die gesamte Verarbeitung in der NAT-Komponente auf Netzwerkebene erfolgt, ist anders als bei der Verteilung über Proxies keine aufwendige Verarbeitung auf Anwendungsebene erforderlich.

Messungen bei einer Packet Screen zeigen, daß der Mehraufwand durch NAT etwa vergleichbar ist mit der Belastung durch 10 bis 20 herkömmliche Filterregeln in der Packet Screen. Die Performanz beim Einsatz von NAT zur Lastverteilung ist damit erheblich besser als beim Einsatz eines Proxies zur Lastverteilung. Die Messungen mit Packet Screens in einem ATM-Netz (s. Abschnitt

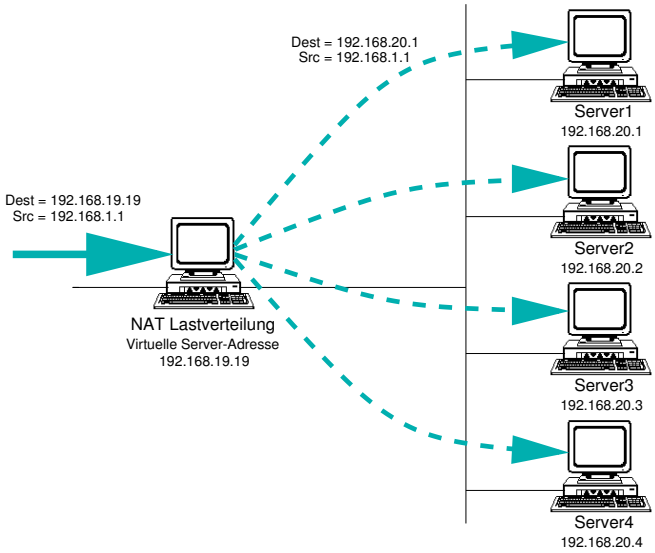


Abbildung 5.2: Lastverteilung über "Network Address Translation" (NAT)

4.3) haben jedoch gezeigt, daß Packet Screens zwar einen hohen Datendurchsatz erreichen, aber aufgrund des beschränkten Datagrammdurchsatzes in Hochgeschwindigkeitsnetzen trotzdem einen Engpaß bilden. Eine Lastverteilung über NAT ist beispielsweise im "Firewall-1" von "Check Point Software Technologies Ltd."[6] und im "LocalDirector" [Cisco 97] von "Cisco" implementiert. "Cisco" gibt den Datendurchsatz des "LocalDirectors" mit 45 MBit/s an. Für Hochgeschwindigkeitsnetze mit Durchsätzen von über 100 MBit/s ist die Performanz des "LocalDirectors" somit nicht ausreichend.

Die Lastverteilung über NAT eignet sich prinzipiell gut für den Einsatz bei Firewall-Architekturen, die ohnehin auf Packet Screens basieren. Abbildung 5.3 zeigt eine Erweiterung der Firewall-Architektur "Beidseitig abgeschirmte Bastion" (s. Abb. 2.10, Seite 42), bei der eine Packet Screen (in Abb. 5.3 links) zur Lastverteilung auf vier parallele Bastionen eingesetzt wird. Die NAT-Komponente führt eine Lastverteilung der Zugriffe durch, die von dem an die Packet Screen angeschlossenen Netz aus eintreffen. Bei den Verbindungen, die dann weiter von den Bastionen aus aufgebaut werden, sind keine Änderungen der IP-Adressen erforderlich, so daß die zweite Packet Screen über keine NAT-Komponente verfügen muß.

Da bei diesem Aufbau nur die Zugriffe von Clients aus einem der Netze auf die parallelen Bastionen verteilt werden, eignet sich dieser Lösungsansatz in erster Linie für Firewalls, bei denen der überwiegende Teil der Verbindungen aus-

[6] Siehe `http://www.checkpoint.com/products/ccontrol/descriptions/ccdata.html`

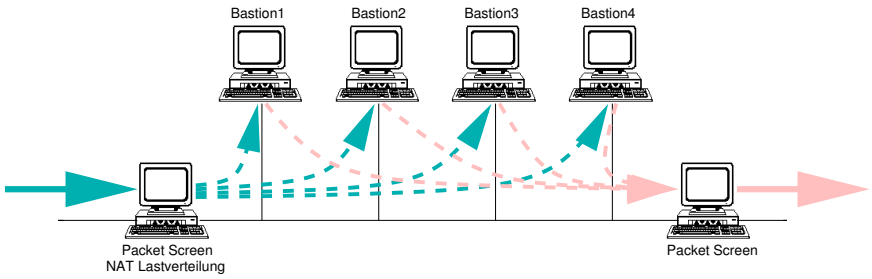


Abbildung 5.3: Lastverteilung über NAT auf parallele Bastionen

schließlich in einer Richtung (z. B. von innen nach außen) aufgebaut wird. In entgegengesetzter Richtung aufgebaute Verbindungen können bei diesem Aufbau nicht verteilt werden. Um auch in entgegengesetzter Richtung eine Lastverteilung durchführen zu können, muß auch in der zweiten Packet Screen (in Abb. 5.3 rechts) eine Lastverteilung durchgeführt werden. Wenn bei der Lastverteilung in beiden Richtungen dieselben Bastionen verwendet werden, ist jedoch zu beachten, daß abhängig vom eingesetzten Verteilungsverfahren beide Verteiler neue Verbindungen an dieselbe vermeintlich gering belastete Bastion zuweisen können und dadurch eine ungleichmäßige Verteilung entstehen kann. Die hierdurch mögliche Überlastung einzelner Bastionen kann verhindert werden, wenn beide Verteiler eingehende Verbindung auf unterschiedliche Bastionen verteilen müssen. Bei der Verteilung in der in Abbildung 5.3 dargestellten Richtung dürfen dann nur noch "Bastion1" und "Bastion2" verwendet werden, während in der Gegenrichtung vom zweiten Verteiler nur "Bastion3" und "Bastion4" ausgewählt werden dürfen. Diese Lösung ist wenig zufriedenstellend, wenn das Verkehrsaufkommen in beiden Richtungen sehr ungleichmäßig ist, so daß dadurch eine ungleichmäßige Verteilung der Last zwischen den entsprechenden Bastionen entsteht. Wenn die Lastverteilung über NAT in beiden Verbindungsaufbaurichtungen für die Verteilung von Verbindungen auf parallele Bastionen eingesetzt werden soll, müssen weitere Untersuchungen zeigen, durch welche Verfahren die Überlastung einzelner Bastionen vermieden werden kann.

Bei der Lastverteilung über NAT muß sichergestellt werden, daß keine logisch zusammenhängenden Datagramme auf unterschiedliche Bastionen verteilt werden. Wie bereits erwähnt, müssen alle Datagramme, die zu derselben TCP-Verbindung gehören, auch von derselben Bastion bearbeitet werden. Diese Eigenschaft wird bereits durch einfache NAT-Komponenten gewährleistet. Zusätzlich gibt es weitere Abhängigkeiten zwischen unterschiedlichen Verbindungen — etwa zwischen FTP-Kontroll- und Datenverbindungen sowie zwischen einer TCP-Verbindung und einer sich auf diese Verbindung beziehenden "Ident"-Anfrage. Weitere Abhängigkeiten bestehen zum Beispiel durch die Verwendung von "Cookies" bei Web-Servern, mit denen versucht wird, einen Zustand zwischen den ein-

zelnen Zugriffen zu erhalten. Um sicherzustellen, daß alle Datagramme eines Clients zum selben Zielrechner auch über dieselbe Bastion abgewickelt werden, darf die Zuordnung von Datagrammen nur anhand der IP-Adressen erfolgen. Für eine weitere Aufteilung, bei der unterschiedliche Verbindungen zwischen denselben Kommunikationspartnern über verschiedene Bastionen laufen, stehen auf Netzwerkebene keine ausreichenden Informationen zur Verfügung, um sicherzustellen, daß logisch zusammenhängende Verbindungen trotzdem über dieselbe Bastion geleitet werden.

Durch IPSEC wird der Einsatz von NAT zukünftig deutlich erschwert. Da bei NAT die Absender- oder Empfänger-Adressen verändert werden, stimmt die im "Authentication Header" (AH) angegebene Prüfsumme nicht mehr überein. Da im Gegensatz zur herkömmlichen "Network Address Translation" bei der Lastverteilung über NAT die von der NAT-Komponente ersetzte Adresse[7] bekannt ist, kann die Bastion vor der Überprüfung des "Authentication Headers" einer empfangenen IP-Dateneinheit die IP-Adresse wieder restaurieren, die ursprünglich in der Empfänger-Adresse gestanden haben muß. Vor dem Senden von Daten muß die Bastion zur Berechnung des AH das IP-Datagramm so zusammenstellen, wie es der Zielrechner (nach dem Ersetzen der Absender-Adresse durch die NAT-Komponente) empfangen würde. Dieses Vorgehen beim Einsatz von IPSEC ist möglich, weil die verschiedenen Komponenten des Firewalls nach außen hin als eine Komponente aufgefaßt werden.

Fazit: Der Einsatz von "Network Address Translation" zur Lastverteilung eignet sich gut für die Aufteilung von Verbindungen auf parallele Bastionen. Für einen Einsatz in Hochgeschwindigkeitsnetzen ist dieser Ansatz jedoch nicht geeignet, da die NAT-Komponente hier einen Engpaß bildet.

5.2.2.4 Verteilung durch einen "TCP-Router"

Eine Variante der Lastverteilung auf Netzwerkebene ist der in [Dias et al. 96] vorgestellte "TCP-Router". Wie bei der Lastverteilung über NAT erfolgen Zugriffe auf einen von den replizierten Servern angebotenen Dienst über eine virtuelle Server-Adresse (`192.168.19.19` in Abbildung 5.4). Dateneinheiten, die an diese Adresse gerichtet werden, erhält der "TCP-Router". Die Lastverteilungsfunktion des "TCP-Routers" wählt für neue TCP-Verbindungen einen Server aus, der die Bearbeitung der Verbindung übernehmen soll. Dateneinheiten für bereits bestehende TCP-Verbindung werden an den zu Beginn der Verbindung ausgewählten Server gesendet. Im Unterschied zur Lastverteilung über NAT verändert der

[7] Wie bereits beschrieben wird eine virtuelle Server-Adresse von der NAT-Komponente durch die IP-Adresse des von der Lastverteilungsfunktion ausgewählten Servers ersetzt.

"TCP-Router" die IP-Adressen der Dateneinheiten vor der Übertragung nicht. Um eine Dateneinheit an den ausgewählten Server weiterzuleiten, wird lediglich die MAC-Adresse (z.B. die Ethernet-Adresse) des entsprechenden Servers verwendet. Wegen der Verwendung der MAC-Adresse ist es erforderlich, daß der "TCP-Router" und alle replizierten Server im selben IP-Subnetz angeschlossen werden. Nur unter dieser Restriktion reicht die Verwendung der MAC-Adresse aus, um den jeweiligen Server zu erreichen. Damit ein Server die an die virtuelle Server-Adresse gerichteten Dateneinheiten annehmen kann, muß die virtuelle Server-Adresse auf jedem Server als zusätzliche Adresse registriert werden.[8] Diese zusätzliche Adresse darf von dem Server aber nicht (z.B. über ARP) bekannt gegeben werden, da alle von Clients eintreffenden IP-Dateneinheiten zu dieser Adresse ausschließlich an den die Verteilung durchführenden "TCP-Router" gesendet werden müssen. Die vom Server als Antwort generierten Dateneinheiten können direkt an den jeweiligen Client zurückgesendet werden und brauchen nicht wie bei der Lastverteilung über NAT vom Verteiler verarbeitet zu werden. Der Server verwendet bei diesen als Antwort generierten Dateneinheiten die virtuelle Server-Adresse als Absender-IP-Adresse. Nachfolgende Dateneinheiten des Clients, die zur selben Verbindung oder zu neuen Verbindungen gehören, werden jedoch in jedem Fall wieder über den "TCP-Router" gesendet, der dann den für die jeweilige Dateneinheit zuständigen Server auswählt.

Prinzipiell hat eine Lastverteilung über einen "TCP-Router" ähnliche Eigenschaften wie die Lastverteilung über NAT. Zwischen der Lastverteilung über einen "TCP-Router" und über NAT gibt es zwei wesentliche Unterschiede: die eingeschränkte Transparenz gegenüber dem Server und die höhere Performanz.

Wie bei der Lastverteilung über NAT ist auch die hier beschriebene Lastverteilung mit einem "TCP-Router" transparent für den anfragenden Client. Für den Server ist dieser Lösungsansatz allerdings nicht transparent, da der Server Dateneinheiten empfangen muß, die an die virtuelle Server-Adresse gerichtet sind. Hierfür ist eine spezielle Konfiguration des Servers erforderlich. Insgesamt sind die erforderlichen Änderungen jedoch nicht umfangreich.

Die Performanz dieser Lösung ist besser als beim Einsatz einer NAT-Komponente zur Lastverteilung, da nur die vom Client an den Server gesendeten Dateneinheiten vom "TCP-Router" verarbeitet werden müssen. In der entgegengesetzten Richtung sendet der Server seine Dateneinheiten hingegen direkt an den Client.

[8] Dieses Annehmen von Dateneinheiten, die an die virtuelle Server-Adresse gerichtet sind, unterscheidet sich wesentlich von der bei einem transparenten NHRP-Proxy (s. Abschnitt 3.3.3.3) erforderlichen Funktionalität, IP-Datagramme annehmen zu müssen, die an *beliebige* Server-Adressen gerichtet sein können. Im Gegensatz zu den hier dargestellten Servern muß ein transparenter NHRP-Proxy nicht nur Dateneinheit mit *beliebigen* Empfänger-Adressen annehmen, sondern muß gleichzeitig auch Dateneinheiten an *dieselbe* Empfänger-Adresse senden können.

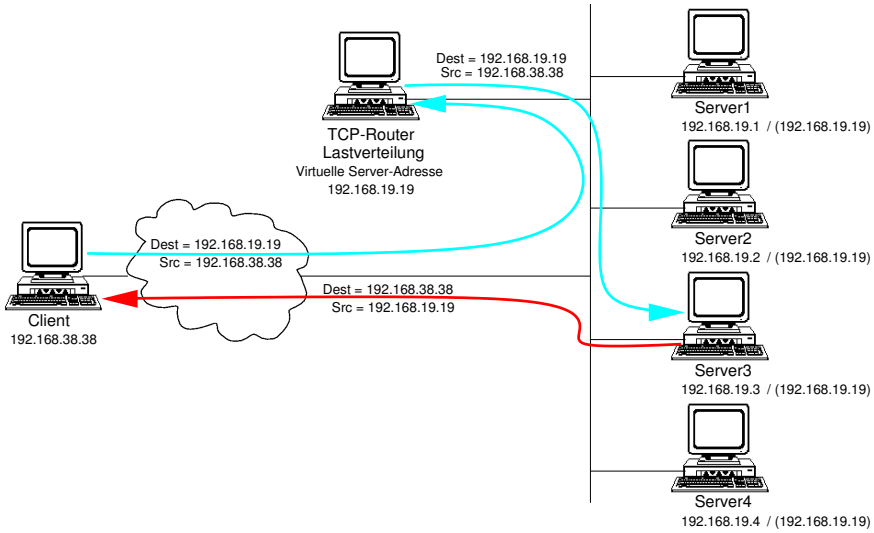


Abbildung 5.4: Lastverteilung über "TCP-Router"

Der "TCP-Router" wird beim gleichen Verkehrsaufkommen also deutlich weniger belastet als eine zur Lastverteilung eingesetzte NAT-Komponente, die den gesamten Verkehr (Hin- und Rückrichtung) bearbeiten muß. In [Dias et al. 96] und in [Gage 98] wird zudem darauf hingewiesen, daß die vom "TCP-Router" zu bearbeitenden Dateneinheiten des Clients in der Regel deutlich kleiner sind als die Dateneinheiten, die vom Server an den Client zurückgesendet werden. In Anbetracht der Ergebnisse der Durchsatzmessungen mit Packet Screens (s. Abschnitt 4.3), die gezeigt haben, daß der Datendurchsatz einer Packet Screen keinen Engpaß bildet, hat dieser vermeintliche Vorteil kaum Relevanz. Beim Datagrammdurchsatz, der bei den Messungen mit Packet Screens als Engpaß identifiziert wurde, muß ein "TCP-Router" nur etwa die Hälfte der Datagramme verarbeiten. Im günstigsten Fall eines unidirektionalen Datentransfers vom Server zum Client muß lediglich für jedes zweite TCP-Segment des Servers eine Bestätigung (ACK) vom Client an den Server gesendet werden.[9] Der "TCP-Router" braucht in diesem Fall somit nur noch 33% der insgesamt übertragenen Datagramme zu bearbeiten.

Der Einsatz eines TCP-Routers zur Lastverteilung auf mehrere parallele Bastionen ist in Abbildung 5.5 wiedergegeben. Im Gegensatz zur Lastverteilung über NAT reicht ein "TCP-Router" zur Verteilung von hereinkommenden und von herausgehenden Verbindungen auf mehrere parallele Bastionen aus. Da es nicht erforderlich ist, zwei getrennte Verteiler für die unterschiedlichen Richtungen einzusetzen, können die im vorigen Abschnitt beschriebenen Überlastsituationen,

[9] Die Bestätigung mindestens jedes zweiten TCP-Segments wird in RFC 1122 [Braden 89, S. 96] gefordert.

bei denen zwei Verteiler eingehende Verbindungen an dieselbe Bastion dirigieren, nicht auftreten.

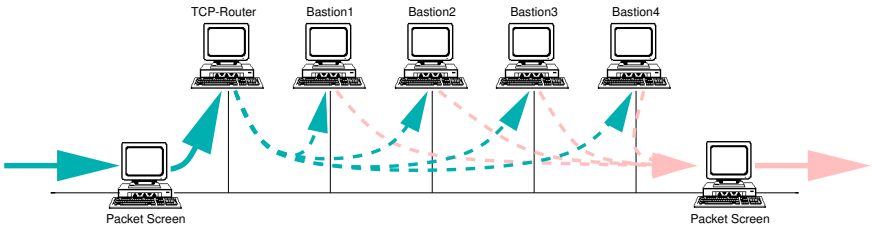


Abbildung 5.5: Lastverteilung über "TCP-Router" auf parallele Bastionen

Fazit: Der "TCP-Router" kann als zentraler Verteiler für eine lastabhängige Verteilung von Verbindungen auf parallele Server eingesetzt werden. Da nur die von den Clients an die Server gesendeten Daten vom "TCP-Router" bearbeitet werden müssen und die für jedes Datagramm erforderlichen Operationen erheblich einfacher sind als bei einer NAT-Komponente, kann vom "TCP-Router" mehr als eine Verdoppelung des Datagrammdurchsatzes im Vergleich zu einer für die Lastverteilung eingesetzten NAT-Komponente erwartet werden. Trotz dieser besseren Performanz bleibt der "TCP-Router" als zentraler Verteiler ein Engpaß, der die Skalierbarkeit dieses Konzeptes einschränkt.

5.2.2.5 Verteilung über virtuelle Verbindungen

Die bisher vorgestellten Parallelisierungskonzepte zur Lastverteilung auf mehrere parallel Bastionen sind für IP-Netze entwickelt worden, können jedoch auch in ATM-basierten IP-Netzen, wie z.B. CLIP-Netzen, eingesetzt werden. Die Ausnutzung ATM-spezifischer Eigenschaften ermöglicht jedoch noch weitere, neue Parallelisierungskonzepte, die in diesem und den folgenden Abschnitten diskutiert werden sollen.

Bei der im Abschnitt 5.2.2.3 diskutierten Lastverteilung mit Hilfe der "Network Address Translation" sind zwei Packet Screens erforderlich (s. Abb. 5.3), um die IP-Adressen der übertragenen Datagramme zu ersetzen und um so Datenströme an ausgewählte Bastionen weiterzuleiten. Durch den Einsatz des in Abschnitt 3.3.3.3 vorgestellten "transparenten Proxies für NHRP-Netze" kann bei der Packet Screen das Ersetzen der IP-Adressen eingespart werden, da ein transparenter Proxy die an ihn gerichteten IP-Dateneinheiten auch dann annimmt, wenn die Empfänger-Adresse nicht mit der eigenen Adresse übereinstimmt. Die Aufgabe der Packet Screen reduziert sich in diesem Fall darauf, ähnlich wie bei dem in Abschnitt 5.2.2.4 beschriebenen "TCP-Router" hereinkommende IP-Datagramme

auf der Basis einer Lastverteilungsfunktion an eine der parallelen Bastionen weiterzuleiten. Dafür muß die Packet Screen bereits vorab jeweils eine virtuelle Verbindung zu jeder Bastion öffnen. Beim Weiterleiten von Dateneinheiten braucht dann lediglich eine der parallelen virtuellen Verbindungen ausgewählt zu werden. Dieses Konzept ist in Abbildung 5.6 skizziert.

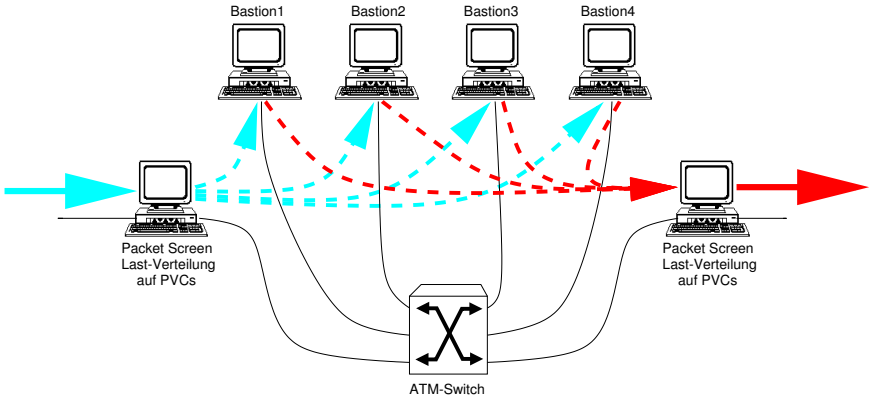


Abbildung 5.6: Lastverteilung über virtuelle Verbindungen auf parallele Bastionen

Das ATM-Netz, an dem die Bastionen angeschlossen sind, ist von Systemen außerhalb dieses Netzes nur über die beiden Packet Screens zu erreichen. Diese Packet Screens leiten ausschließlich IP-Datagramme weiter, so daß die Existenz des ATM-Netzes nach außen hin nicht erkennbar ist. Auch wenn an den Packet Screens wiederum ATM-Netze angeschlossen werden, enden bei diesem Konzept alle virtuellen Verbindungen immer an einer der Packet Screens. Innerhalb des in Abbildung 5.6 dargestellten Netzes reichen PVCs aus, die jede Packet Screen mit jeder Bastion verbinden. Eine Kommunikation der Bastionen untereinander oder der Packet Screens untereinander ist nicht erforderlich, so daß bei dem in Abbildung 5.6 dargestellten Aufbau mit 4 Bastionen insgesamt 8 PVCs ausreichen.

Die Packet Screens nehmen bei diesem Konzept die Aufgabe eines spezialisierten Routers wahr. Dieser spezialisierte Router unterscheidet sich von herkömmlichen Routern hauptsächlich in zwei Punkten. Während ein herkömmlicher Router die Schnittstelle, auf der eine Dateneinheit weitergeleitet werden muß, über die Empfänger-IP-Adresse bestimmt, wird hier lastabhängig der Pfad bestimmt.[10] Der zweite Unterschied betrifft die virtuellen Verbindungen (PVCs), die hier die unterschiedlichen physikalischen Schnittstellen eines herkömmlichen Routers ersetzen.

[10] Zusätzlich zur lastabhängigen Auswahl kann auch eine Sicherheitsrichtlinien-basierte Auswahl des Pfades sinnvoll sein (siehe Abschnitt 3.3.2.4).

Da die Packet Screens von außen hereinkommende Datagramme lediglich an die Bastionen weiterleiten, kann die ursprüngliche Aufgabe der Packet Screens — das Filtern, der gegen die Sicherheitsrichtlinien verstoßenden Datagramme — bei diesem Aufbau entfallen. Allerdings ist das Absichern der Bastionen aufwendiger, wenn nicht nur Dateneinheiten an Ports ankommen können, an denen sichere Proxies installiert sind. Wird auf die Filterung von Datagrammen in den Packet Screens verzichtet, müssen die Bastionen abgesichert werden wie Gateway-Firewalls.

Fazit: Der hier vorstellte Ansatz, bei dem IP-Datagramme von den Packet Screens über virtuelle Verbindungen direkt an die Bastionen weitergeleitet werden, basiert auf der in Abschnitt 5.2.2.3 vorgestellten Lastverteilung über NAT. Im Gegensatz zu üblichen "Routing"-Verfahren, die die Empfänger-Adresse verwenden, bestimmt die Packet Screen hier den Weg des Datagramms über eine Lastverteilungsfunktion. Durch das Entfallen der Änderung der IP-Adressen bei der Übertragung durch die Packet Screens ist diese Lösung effizienter als die Lastverteilung über NAT. Trotzdem stellt der zentrale Verteiler auch bei diesem Konzept immer noch einen Engpaß dar.

5.2.2.6 Verteilung durch NHRP

Ein Verteilungsverfahren, bei dem der Verteiler nicht selbst an der anschließenden Datenübertragung beteiligt ist, wird in ATM-Netzen durch das "Next Hop Resolution Protocol" (NHRP) (s. Abschnitt 2.4.3) ermöglicht. NHRP ist eine Erweiterung des bei CLIP-Netzen eingesetzten ATMARP (s. Abschnitt 2.4.2) — einem Client-Server-basierten Protokoll, über das die zu einer IP-Adresse gehörende ATM-Adresse ermittelt werden kann. Zusätzlich zur Funktionalität von ATMARP können bei NHRP auch die ATM-Adressen von Systemen außerhalb des eigenen Subnetzes ermittelt werden. Dadurch können anschließend zwischen den Endsystemen unterschiedlicher Subnetze direkte virtuelle Verbindungen (SVCs) für die Übertragung von IP-Datagrammen eingesetzt werden. NHRP-Anfragen ("NHRP Resolution Requests") eines "Next Hop Clients" (NHC) werden wie IP-Datagramme entlang des Pfades zum Empfänger von einem Router zum nächsten Router weitergeleitet, bis ein "Next Hop Server" (NHS) die NHRP-Anfrage mit der ATM-Adresse des gewünschten Empfängers oder des letzten direkt über das ATM-Netz erreichbaren Routers beantwortet.

In Abschnitt 3.3.1.3 wurde bereits dargestellt, wie Proxies und andere Sicherheitskomponenten über NHRP eingebunden werden können. Mit derselben Technik kann das "Next Hop Resolution Protocol" auch zur Lastverteilung eingesetzt werden, indem der NHS, der für das Subnetz der parallelen Bastionen zuständig ist, eine Bastion über eine Verteilungsfunktion auswählt und die NHRP-Anfrage eines

NHCs mit der ATM-Adresse der ausgewählten Bastion beantwortet (s. Schritt 1 in Abbildung 5.7). Der NHC wird danach eine virtuelle Verbindung zu der über NHRP ermittelten ATM-Adresse aufbauen (s. Schritt 2) und anschließend IP-Datagramme über diese Verbindung an die Bastion senden. Da diese IP-Datagramme die IP-Adresse des vom NHC gewünschten Servers als Empfänger-Adresse tragen, müssen auch bei dieser Lösung wieder transparente NHRP-Proxies (s. Abschnitt 3.3.3.3) auf den Bastionen eingesetzt werden. Zum Erreichen des vom NHC gewünschten Servers kann der Proxy wiederum nach einer NHRP-Anfrage (s. Schritt 3) eine zweite virtuelle Verbindung zu einem entsprechenden System[11] im jeweils anderen Netz aufbauen (s. Schritt 4). Die beiden NHRP-Anfragen — die Anfrage des NHCs und die Anfrage des Proxies — werden dabei entlang des durch das "IP-Routing" bestimmten Pfades zum Empfänger weitergeleitet und erreichen daher unterschiedliche NHS. Während der NHS "A" die NHRP-Anfrage des NHCs mit der ATM-Adresse einer Bastion beantwortet, wird die Anfrage der Bastion, bei der nach derselben Adresse gefragt wird, vom NHS "B" verarbeitet und kann so die tatsächliche Adresse des Servers ermitteln.[12] Um zu verhindern, daß ein NHC ohne vorherige NHRP-Anfrage eine virtuelle Verbindung direkt zum Server aufbaut, muß ein Signalisierungsfilter im ATM-Switch (s. Abbildung 5.7) den Aufbau direkter virtueller Verbindungen verhindern.

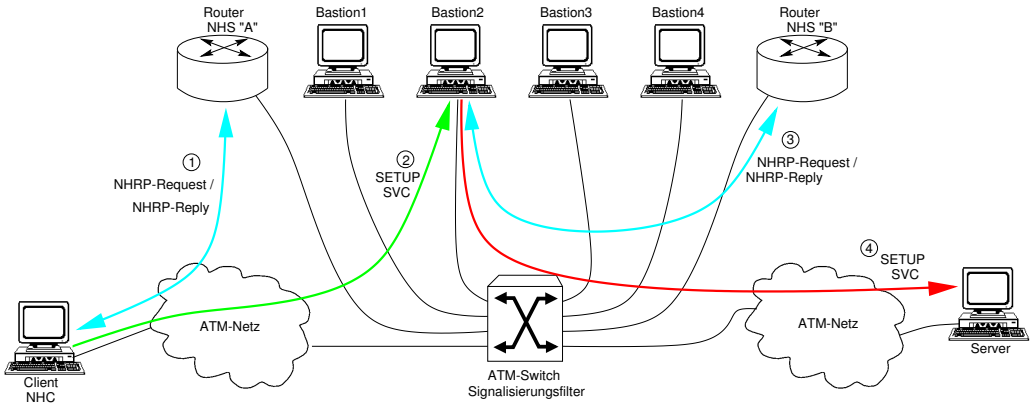


Abbildung 5.7: Lastverteilung mit NHRP

Dieser Ansatz der Lastverteilung über NHRP ähnelt der in Abschnitt 5.2.2.1 vorgestellten Lastverteilung über "Round-Robin"-DNS. Im Gegensatz zur Auflö-

[11] Bei dem System, zu dem die zweite virtuelle Verbindung aufgebaut wird, kann es sich wie in Abbildung 5.7 dargestellt um den Server selbst oder um den letzten direkt an das ATM-Netz angeschlossenen Router auf dem Pfad zum Server handeln.

[12] Für einen NHC, der den Firewall in Rückrichtung überquert, ist der Ablauf entsprechend. Da die NHRP-Anfrage des NHCs hier zuerst den NHS "B" erreicht, muß in diesem Fall der NHS "B" die Lastverteilung zwischen den parallelen Bastionen durchführen. Die Probleme, die sich ergeben, wenn zwei Verteiler eingehende Verbindungen auf dieselbe Reihe von Bastionen verteilen, sind bereits im Abschnitt 5.2.2.3 diskutiert worden.

sung von Hostnamen in IP-Adressen bei DNS werden bei NHRP IP-Adressen in ATM-Adressen aufgelöst. Die ATM-Adresse verwendet ein NHC für den Aufbau einer virtuellen Verbindung (SVC), über die anschließend IP-Dateneinheiten gesendet werden. Dabei verwendet der NHC dieselbe virtuelle Verbindung für alle IP-Dateneinheiten, die an die in der NHRP-Anfrage enthaltene IP-Adresse gerichtet sind. Dadurch ergibt sich eine Bindung zwischen NHC und Bastion, bei der die Bastion auch die TCP-Verbindungen unterschiedlicher Client-Prozesse desselben NHCs verarbeiten muß. Eine weitere Aufteilung von unterschiedlichen TCP-Verbindungen desselben NHCs auf verschiedene Bastionen ist bei der Lastverteilung über NHRP nicht möglich.[13] Dieses Konzept zur Lastverteilung ist somit nur dann anwendbar, wenn kein NHC allein eine Bastion überlasten kann. Das dargestellte Problem verschärft sich dadurch, daß auch Router als NHC eine virtuelle Verbindung aufbauen und diese virtuelle Verbindung für den Datenverkehr mehrerer Systeme verwenden können.

Eine einmal aufgebaute virtuelle Verbindung bleibt üblicherweise solange erhalten, wie zwischen den Kommunikationspartnern Dateneinheiten ausgetauscht werden. Da währenddessen keine neue Lastverteilung zwischen den Bastionen erfolgen kann, wird die Lastverteilung über NHRP häufig zu suboptimale Aufteilungen führen. Dies ist beispielsweise dann der Fall, wenn zum Zeitpunkt der Zuteilung bereits mehrere virtuelle Verbindungen für diese Bastion bestehen, diese aber kaum Verkehr aufweisen und diese wenig belastete Bastion daher von der Verteilungsfunktion ausgewählt wird. Nimmt der Verkehr auf diesen virtuellen Verbindungen anschließend zu, gibt es keine Möglichkeit mehr, die getroffene Entscheidung zu revidieren.[14]

Die Zwischenspeicherung von NHRP-Antworten stellt ähnlich wie die Zwischenspeicherung von DNS-Informationen ein Hindernis für die Lastverteilung dar. Adreßinformationen aus "NHRP-Replies" werden von den NHS auf dem Pfad zum NHC zwischengespeichert. Diese Zwischenspeicherung von Informationen kann bei NHRP über die Angabe der "Holding Time" begrenzt werden. Allerdings hat die Verhinderung der Zwischenspeicherung eine Zunahme des NHRP-Verkehrs zur Folge, da eine NHRP-Anfrage nicht mehr von einem NHS auf dem Pfad zwischen NHC und Bastion beantwortet werden kann. Viel wichtiger als die Zunahme des Verkehrs ist dabei die Verlängerung der Zeit, die zur Beantwortung einer NHRP-Anfrage benötigt wird.

---

[13] Im Unterschied dazu stellt jeder Client-Prozeß eigene DNS-Anfragen, so daß bei einer Verteilung über "Round-Robin"-DNS die TCP-Verbindungen unterschiedlicher Client-Prozesse auf unterschiedliche Bastionen verteilt werden können. Die bereits diskutierte Zwischenspeicherung von DNS-Informationen kann jedoch eine effektive Verteilung behindern.

[14] Dieses Problem kann vermieden werden, wenn bereits bei der NHRP-Anfrage die erforderliche Dienstgüte spezifiziert werden könnte. Damit könnte vorab festgestellt werden, auf welcher Bastion die erforderlichen Betriebsmittel zur Verfügung stehen.

Fazit: Der Vorteil dieser ATM-spezifische Lösung ist, daß der Verteiler nicht an der nachfolgenden Datenübertragung beteiligt ist und daher nicht wie bei den anderen auf einem zentralen Verteiler basierenden Verfahren (z. B. NAT und "TCP-Router") selbst zum Engpaß wird. Nachteil der Lastverteilung über NHRP ist die geringe Parallelität, die mit diesem Verfahren erreicht wird, wenn eine virtuelle Verbindung zu einer Bastion gleichzeitig für mehrere TCP-Verbindungen verwendet wird. Zudem ist diese Lösung von NHRP abhängig, das bisher noch keine nennenswerte Verbreitung gefunden hat. Unabhängig von den zusätzlichen Fragen, die durch die hier diskutierte Lastverteilung entstehen, sind die Vor- und Nachteile sowie insbesondere die Performanzeigenschaften von NHRP heute noch nicht ausreichend untersucht. Auf der Grundlage der heutigen Erkenntnisse mit NHRP kann daher noch keine ausreichende Bewertung eines NHRP-basierten Parallelisierungsverfahrens erfolgen.

5.2.2.7 Signalisierungskontroller

Ein weiteres ATM-spezifisches Konzept ist die Lastverteilung über einen Signalisierungskontroller. Das Einbinden von Sicherheitskomponenten beim Aufbau virtueller Verbindungen mit Hilfe eines Signalisierungskontrollers wurde bereits im Abschnitt 3.3.1.2 ausführlich diskutiert. Bei der Diskussion im Abschnitt 3.3.1.2 wurde auch bereits kurz erwähnt, daß ein Signalisierungskontroller zur Lastverteilung zwischen mehreren parallelen Sicherheitskomponenten auswählen kann. Da die Verteilung durch den Signalisierungskontroller unabhängig von eventuell vorangegangenen NHRP-Anfragen ist und ausschließlich in die Signalisierung eingreift, kann ein Signalisierungskontroller in unterschiedlichen Einsatzbereichen (CLIP, NHRP, LANE, MPOA, "Native"-ATM) für eine Lastverteilung eingesetzt werden. Ein möglicher Aufbau für eine Lastverteilung über einen Signalisierungskontroller ist in Abbildung 5.8 wiedergegeben.

Das Konzept des Signalisierungskontrollers erlaubt einen sehr flexiblen Einsatz von Sicherheitskomponenten, wodurch parallele Bastionen fast beliebig im ATM-Netz plaziert werden können. Der in Abbildung 5.8 dargestellte Aufbau mit direkt an den Signalisierungskontroller angeschlossenen Bastionen stellt lediglich den einfachsten Fall dar.

Die mit einem Signalisierungskontroller erreichbare Parallelität entspricht etwa der Parallelität, die mit einer Verteilung über NHRP erreicht werden kann, denn in beiden Fällen muß jeweils eine virtuelle Verbindung einer Bastion zugeordnet werden. Eine weitere Aufteilung der auf einer virtuellen Verbindung übertragenen Daten ist in beiden Fällen nicht möglich. Im Gegensatz zur Verteilung über NHRP erfordert die Verteilung über einen Signalisierungskontroller keine Unterstützung durch das zugreifende Endsystem. Daher kann es nicht zu Problemen durch die Zwischenspeicherung von Adreßinformationen kommen, wie es zum

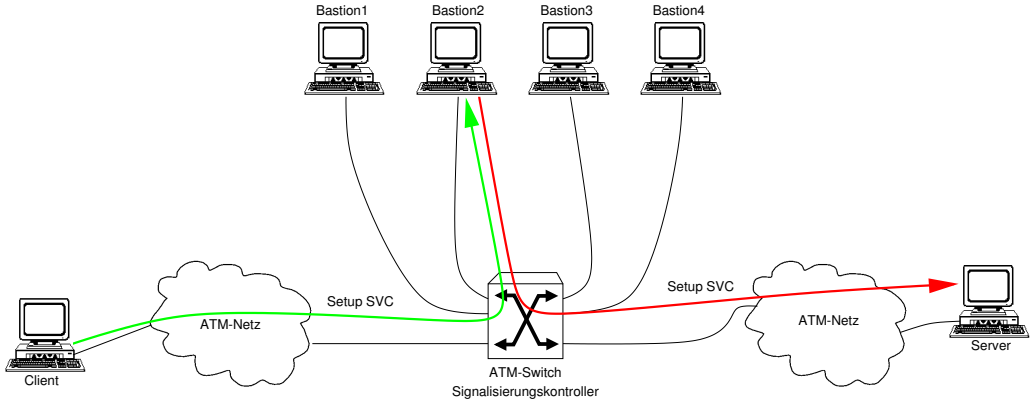


Abbildung 5.8: Lastverteilung über Signalisierungskontroller

Beispiel bei der Verteilung über NHRP der Fall ist (s. Abschnitt 5.2.2.6). Auch für Netze in denen NHRP eingesetzt wird, ist es daher effizienter eine Lastverteilung über einen Signalisierungskontroller zu implementieren als NHRP selbst zur Lastverteilung einzusetzen. Da es bei der Verteilung über einen Signalisierungskontroller nicht erforderlich ist, die Zwischenspeicherung von NHRP-Informationen zu verhindern, können NHRP-Anfragen aus dem Zwischenspeicher eines beliebigen NHS beantwortet werden und müssen nicht, wie bei der Verteilung über NHRP erforderlich, bis zu dem für die Bastionen zuständigen NHS weitergeleitet werden. Dadurch sind die Verzögerungen beim Verbindungsaufbau bei der Verteilung über einen Signalisierungskontroller geringer als bei der Verteilung über NHRP.

Durch die im ATM-Konzept enthaltene Trennung der Signalisierung von der Datenübertragung ist der Eingriff eines Signalisierungskontrollers nur beim Verbindungsaufbau erforderlich. Die anschließende Datenübertragung wird hingegen wie bei jeder virtuellen Verbindung sehr effizient von der Switch-Hardware durchgeführt. Obwohl der Signalisierungskontroller ein zentraler Verteiler ist, verhindert die effiziente Datenübertragung im Switch, daß durch den zentralen Verteiler ein Engpaß entsteht.

Fazit: Die Lastverteilung über einen Signalisierungskontroller ist ein ATM-spezifischer Ansatz, der trotz eines zentralen Verteilers keinen Engpaß bildet. Da die heutigen Konzepte zum Einsatz von IP in ATM-Netzen auch für unterschiedliche TCP-Verbindungen zwischen denselben Kommunikationspartnern dieselbe virtuelle Verbindung verwenden, ist die mit einem Signalisierungskontroller erreichbare Parallelität geringer als bei den speziell auf IP ausgerichteten Parallelisierungskonzepten, bei denen auch verschiedene TCP-Verbindungen zwischen denselben Kommunikationspartnern auf unterschiedliche Bastionen verteilt werden können.

Als einziger hier vorgestellter Ansatz kann der Signalisierungskontroller auch zur Lastverteilung bei "Native"-ATM-Anwendungen eingesetzt werden.

5.2.3 Dezentrale Verteilung von Verbindungen

Alle bisher vorgestellten Verfahren basieren auf einem zentralen Verteiler, der die eingehenden Verbindungen den parallelen Bastionen zuteilt. Die Untersuchungen im Abschnitt 5.2.2 haben gezeigt, daß ein zentraler Verteiler, der an der Datenübertragung beteiligt ist, in der Regel selbst zum Engpaß wird, in jedem Fall aber die Skalierbarkeit des Verteilungsverfahrens einschränkt. Ohne Beteiligung des Verteilers an der Datenübertragung kann hingegen eine zentrale Verteilung nur erreicht werden, wenn Verteilung und Datenübertragung in zwei getrennten Phasen stattfinden. Nachteil dieser Lösungen ist die Verzögerung, die durch die vor der eigentlichen Datenübertragung stattfindende erste Phase bedingt ist.[15] Die Verzögerung durch die in der ersten Phase genutzten Protokolle (DNS und NHRP) ist deutlich höher als bei herkömmlichem Einsatz dieser Protokolle, da beim Einsatz zur Lastverteilung, wie in Abschnitt 5.2.2.1 dargestellt, keine Zwischenspeicherung ("Caching") zur Verringerung der Verzögerungen eingesetzt werden kann. Diese bei zentralen Verteilern entstehenden Performanzprobleme sollen durch dezentrale Verteilungsverfahren vermieden werden.

Bei einem dezentralen Verteilungsverfahren entscheidet jeder der parallelen Prozessoren selbst, ob ein neuer Auftrag bearbeitet werden soll. Bei der hier diskutierten Verteilung von Verbindungen auf parallele Bastionen entscheidet somit jede Bastion selbst, ob sie eine neu aufgebaute Verbindung bearbeiten soll. Damit die Entscheidung, ob eine Verbindung bearbeitet werden soll, dezentral getroffen werden kann, ist es wichtig, daß die zum Treffen der Entscheidung verwendete Verteilungsfunktion eindeutig ist. Durch die Verteilungsfunktion muß also sichergestellt werden, daß genau eine Bastion die hereinkommende Verbindung zur Bearbeitung auswählt.

5.2.3.1 Zufällige Auswahl einer Bastion

Eine einfache Methode zur Verteilung von Verbindungen auf parallele Bastionen ist auf der Grundlage des Zufalls möglich. Dabei werden lediglich mehrere parallele Bastionen installiert und deren Adressen bekanntgegeben. Clients wählen aus den Adressen der Bastionen zufällig eine Adresse aus, die für jeweils einen Zugriff verwendet wird.

[15] Bevor die Datenübertragung beginnen kann, muß bei diesen Verfahren zuerst die DNS-Anfrage (bei "Round-Robin"-DNS, s. Abschnitt 5.2.2.1) oder die NHRP-Anfrage (bei Verteilung über NHRP, s. Abschnitt 5.2.2.6) beantwortet werden.

Da die zufallsbasierte Auswahl einer Bastion nicht lastabhängig erfolgt, kann dieses Verfahren zu einer guten aber auch zu einer schlechten Lastverteilung führen. Insbesondere wenn die zufällige Auswahl einer Adresse dem Benutzer überlassen wird,[16] ist mit ungleichmäßiger Verteilung zu rechnen, da Benutzer dazu tendieren, bestimmte Adressen häufiger auszuwählen als andere Adressen. Wählt der Client transparent für den Benutzer eine Bastion aus, kann eine gleichmäßigere Verteilung der Verbindungen erreicht werden. Allerdings beinhaltet die zufällige Auswahl einer Bastion immer auch die Möglichkeit, zufällig eine ungünstige Wahl zu treffen. Da die von einer Bastion zu verarbeitenden Verbindungen zusammengesetzt sind aus langen und kurzen Verbindungen mit sehr großen Unterschieden bei der zu übertragenden Datenmenge (s. Tab. C.2, S. 471), können ungleichmäßige Verteilungen entstehen, die zur Überlastung einer einzelnen Bastion führen können. Da auch überlasteten Bastionen weiterhin neue Verbindungen zugewiesen werden, kann einen Überlastsituation über längere Zeit bestehen bleiben.

Tendenziell eignet sich die zufällige Verteilung von Verbindungen besser für Anwendungsbereiche, in denen sehr viele kurze Verbindungen verteilt werden müssen, als für Anwendungsbereiche, in denen nur wenige aber lange und lastintensive Verbindungen verteilt werden müssen. Im Extremfall müssen lediglich zwei sehr lange Verbindungen auf zwei Bastionen verteilt werden. Eine zufällige Verteilung ergibt in diesem Fall mit einer Wahrscheinlichkeit von 50% eine Zuordnung beider Verbindungen auf dieselbe Bastion.[17] Die Verteilung von sehr vielen kurzen Verbindungen stellt den anderen Extremfall dar. Mit den in Abschnitt 4.2.7 zur Messung des maximalen Transaktionsdurchsatz verwendeten Verbindungen, bei denen lediglich 250 Byte lange HTML-Dokumente übertragen werden, kann beispielsweise mit den "Ultra-2"-Systemen ein Transaktionsdurchsatz von knapp 1.800 Tr/s erreicht werden. Eine extrem ungleichmäßige Verteilung wie bei dem Beispiel der Verteilung von zwei Verbindungen kann bei dieser hohen Anzahl von Verbindungen kaum noch auftreten.

Fazit: Diese einfache Methode zur dezentralen Verteilung von Verbindungen erfolgt lastunabhängig und kann daher auch zu ungleichmäßiger Lastverteilung führen. Eine einigermaßen gleichmäßige Verteilung der Last ergibt sich durch dieses Verfahren nur dann, wenn die erzeugte Last bei jedem Auftrag etwa gleich

[16] Jeder Benutzer wählt selbständig aus einer ihm vorliegenden Liste der Bastionen eine Adresse aus, die er dann beim Aufruf des Clients angibt.

[17] Da jede der beiden Verbindungen mit einer Wahrscheinlichkeit von 50% einer der beiden Bastionen zugeordnet wird, ergeben sich vier mögliche Verteilungen der beiden Verbindung auf die beiden Bastionen. Jede dieser vier möglichen Verteilungen wird mit einer Wahrscheinlichkeit von 25% erreicht. Da es hierbei zwei ungünstige Verteilungen gibt (beide Verbindungen auf Bastion 1 oder beide Verbindungen auf Bastion 2), müssen mit einer Wahrscheinlichkeit von 50% beide Verbindungen auf der gleichen Bastion bearbeitet werden.

hoch ist und wenn die Leistungsfähigkeit aller Bastionen identisch ist. Dieses Verfahren eignet sich besser zur Verteilung vieler kurzer Verbindungen als zur Verteilung weniger langer Verbindungen.

5.2.3.2 Transparentes Umleiten von HTTP-Verbindungen

Die einfachste Möglichkeit für eine lastabhängige dezentrale Verteilung von Verbindungen ist es, die Verbindung eines Clients zunächst anzunehmen, sie aber gegebenenfalls wieder abzubrechen, falls dies zur Lastverteilung erforderlich ist.

Bei HTTP ist mit dem "Redirect" [Fielding et al. 97, S. 56ff] ein Mechanismus enthalten, der es einem Server ermöglicht, einem Client eine alternative URL [Berners-Lee et al. 94] mitzuteilen. Der "Redirect" ist bei HTTP vorgesehen, um das Verschieben von Dokumenten zwischen unterschiedlichen URLs zu erleichtern. Einem Client, der über die alte Adresse (URL) auf ein Dokument zugreift, kann durch einen "Redirect" die neue Adresse des Dokuments mitgeteilt werden. Dieses Umleiten von Zugriffen erfordert keine Interaktion des Benutzers und ist somit transparent für den Benutzer.

Diese Verweise über "Redirects" können auch zur Lastverteilung zwischen replizierten Web-Servern genutzt werden, indem ein stark belasteter Server Anfragen über einen Verweis an einen weniger belasteten Server umleitet. "Redirects" werden heute in kommerziellen Produkten wie "Check Point Firewall-1"[18] eingesetzt, um eine zentrale Lastverteilung zu implementieren.[19] Web-Server, die "Redirects" für eine dezentrale Lastverteilung einsetzen, sind zur Zeit nicht bekannt.

Das Umleiten von Zugriffen über "Redirects" hat zwei wesentliche Nachteile. Bei einfachen Anwendungen, bei denen wie bei einem Web-Server für jeden Zugriff lediglich eine geringe Datenmenge an den Client gesendet werden muß, hat der Verbindungsaufbau einen wesentlichen Anteil an der Last, die durch einen Zugriff entsteht. Da der Server auch für das Senden eines "Redirects" die Verbindung zuerst annehmen muß, tragen auch abgelehnte Verbindungen in ähnlichem Maße zur Last auf dem Web-Server bei wie bearbeitete Verbindungen. Für einen Client bedeutet ein "Redirect", daß eine weitere Verbindung zum Laden der gewünschten URL benötigt wird. Wie die Messungen im Abschnitt 4.2.4 gezeigt haben, dominieren jedoch die Verbindungsaufbauzeiten bei kurzen Web-Zugriffen die Ge-

[18] Siehe `http://www.checkpoint.com/products/ccontrol/descriptions/ccdata.html`

[19] Bei der Lastverteilung durch den "Check Point Firewall-1" wird die Adresse des Firewalls für Web-Zugriffe verwendet. Der Firewall verfügt jedoch über keinen Web-Server, sondern leitet Zugriffe lediglich über "Redirects" an einen regulären Web-Server weiter. Der Client richtet daraufhin alle folgenden Zugriffe direkt an den Web-Server, so daß über dieses Verfahren pro Client nur eine einmalige Lastverteilung möglich ist.

samttransaktionslänge, so daß sich ein "Redirect" sehr ungünstig für den Client auf die Performanz auswirkt.

Fazit: Das Umleiten von Zugriffen über "Redirects" hat sowohl für den zugreifenden Client als auch für den Server, der mit einem "Redirect" reagiert, negative Performanzauswirkungen, so daß dieses Verfahren nur dann zur Lastverteilung eingesetzt werden kann, wenn das Umleiten von Zugriffen nur selten erforderlich ist.

5.2.3.3 Transparentes Umleiten von "Socks"-Verbindungen

Bei Proxies ist bisher noch keine dezentrale Lastverteilung implementiert worden, obwohl sich die dezentrale Lastverteilung gut für die Verteilung von Verbindungen auf parallele Proxies eignet. "Socks" [Leech et al. 96] ist ein einfaches Protokoll, über das ein Client einen Proxy auf einem Firewall ansprechen kann, um auf Server jenseits des Firewalls zuzugreifen. Anstelle eines direkten TCP-Verbindungsaufbaus zu einem Server öffnet der Client eine TCP-Verbindung zum Proxy und überträgt über das "Socks"-Protokoll die Adresse und Portnummer des Servers, zu dem der Proxy eine Verbindung öffnen soll. Nach Überprüfung der Zugriffsrechte öffnet der Proxy eine Verbindung zum gewünschten Server und teilt dem Client schließlich über das "Socks"-Protokoll mit, ob der Verbindungsaufbau erfolgt ist. Mit dieser abschließenden Statusmeldung vom Proxy an den Client ist die Aufgabe des "Socks"-Protokolls erfüllt. Der hier beschriebene Ablauf eines Verbindungsaufbaus mit "Socks" ist in Abbildung 5.9 dargestellt. Der Server kopiert anschließend empfangene Daten zwischen den beiden offenen Verbindungen, so daß Client und Server die Verbindung über den Proxy wie eine direkte Verbindung nutzen können. Damit Verbindungen mit Hilfe des "Socks"-Protokolls geöffnet werden können, sind nur geringe Anpassungen bei den Clients erforderlich, da bei der Funktion, die das "Socks"-Protokoll zum Öffnen einer Verbindung verwendet, dieselben Parameter benötigt werden wie bei der herkömmlichen "`connect()`"-Funktion.[20]

Über eine kleine Ergänzung des "Socks"-Protokolls kann es einem Proxy ermöglicht werden, einem Client unmittelbar nach dem Aufbau der Verbindung einen anderen, weniger belasteten Proxy zu nennen und die Verbindung abzubrechen (s. Abb. 5.10). Der Client muß dann den genannten Ausweich-Server als Proxy verwenden. Konzeptionell entspricht dieses Vorgehen dem in Abschnitt 5.2.3.2 beschriebenen Einsatz von "Redirects" zur Lastverteilung zwischen parallelen Web-Servern.

[20] Die Anpassung der Anwendungen kann elegant über eine "Shared-Library" erfolgen, so daß keine erneute Übersetzung ("Compilieren") bereits installierter Clients notwendig ist. [Ellermann 93]

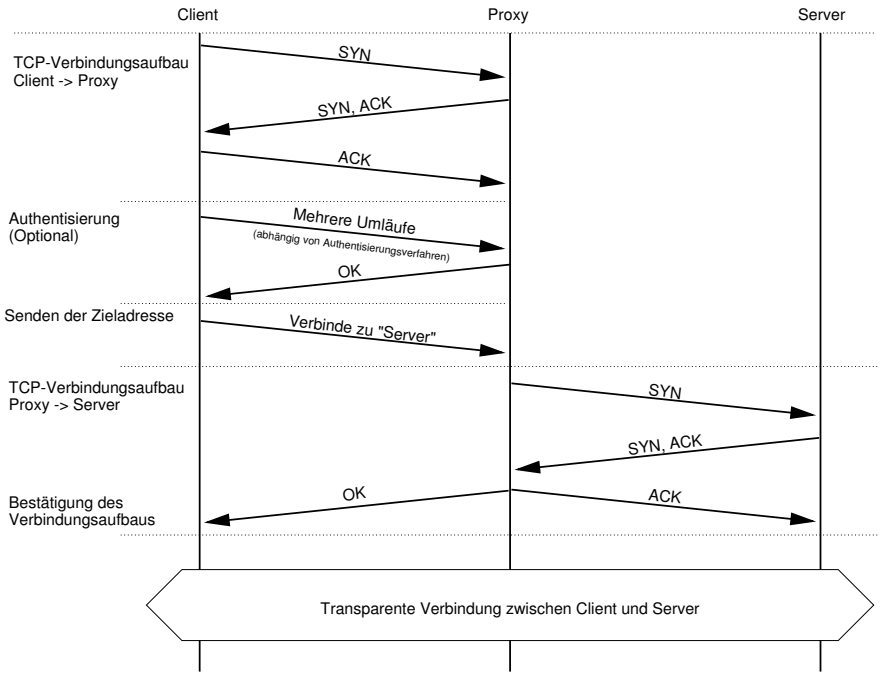


Abbildung 5.9: Verbindungsaufbau über “Socks”

Damit eine dezentrale Lastverteilung erfolgen kann, ist es erforderlich, zwischen den parallelen Bastionen Lastinformationen auszutauschen. Jeder Proxy kann aus diesen Lastinformationen selbst berechnen, ob das Umleiten eines Zugriffs sinnvoll ist und an welche Bastion Zugriffe umgeleitet werden sollen.

Mit diesem Verteilungsverfahren kann jede Verbindung einer eigenen Bastion zugeordnet werden. Lediglich Abhängigkeiten zwischen einzelnen Verbindungen, die vom Proxy bei der Überwachung der Verbindungen berücksichtigt werden müssen, können die erreichbare Parallelität etwas einschränken.

Das Umleiten einer Verbindung auf eine andere Bastion verlängert die Verbindungsaufbauzeiten für den Client und erreicht hinsichtlich des Transaktionsdurchsatzes eines Proxies nur in wenigen Fällen eine Steigerung. Diese Verfahren eignet sich daher besser für die Verteilung weniger langer Verbindungen, bei denen die Verlängerung der Verbindungsaufbauzeit im Verhältnis zur Gesamttransaktionslänge kaum relevant ist und bei denen der Transaktionsdurchsatz des Proxies keinen Engpaß bildet. Für die Verteilung sehr vieler kurzer Transaktionen ist dieses Verfahren hingegen wenig geeignet, da die Verbindungsaufbauzeiten bei kurzen Verbindungen dominieren (s. Abschnitt 4.4.3) und da der Transaktionsdurchsatz durch das Umleiten kurzer Verbindungen nicht gesteigert werden kann. Bei der Messung des maximalen Transaktionsdurchsatzes eines Proxies in Abschnitt 4.4.2.6 zeigte sich, daß bei den zur Messung verwendeten kurzen Trans-

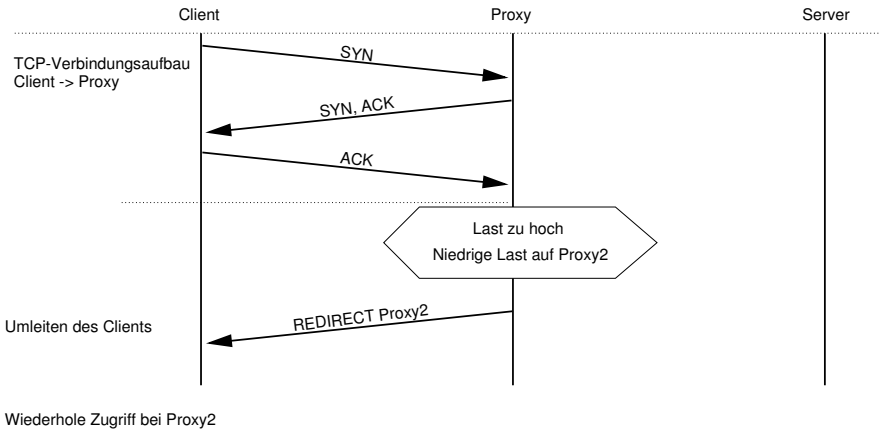


Abbildung 5.10: Umleiten eines Zugriffs zur Lastverteilung

aktionen die Annahme der Verbindungen beim Proxy den Engpaß bildet, der den Transaktionsdurchsatz begrenzt. Da auch zum Umleiten einer Verbindung zuerst die Annahme der Verbindung erforderlich ist, kann durch dieses Verfahren keine Erhöhung des Transaktionsdurchsatzes eines Proxies erreicht werden. Eine Erhöhung des Transaktionsdurchsatzes ist bei kurzen Transaktionen nur dann möglich, wenn nach dem Verbindungsaufbau eine aufwendige kryptographische Authentisierung erforderlich ist, die anstelle der Verbindungsannahme einen Engpaß bildet (s. Abschnitt 4.4.2.9).

Ähnlich wie beim "Redirect" eines Web-Servers (s. Abschnitt 5.2.3.2) ist dieses dezentrale Verteilungsverfahren nur dann empfehlenswert, wenn das Umleiten von Verbindungen nur relativ selten erforderlich ist. Wie häufig das Umleiten von Verbindungen erforderlich ist, hängt auch davon ab, auf welche Bastion von Clients zuerst zugegriffen wird. Beginnen beispielsweise alle Clients mit Zugriffen immer bei derselben Bastion, ergibt sich mit dieser Bastion quasi ein zentraler Verteiler,[21] der selbst zum Engpaß wird. Mit besseren Ergebnissen wird gerechnet, wenn dieses Verteilungsverfahren mit der in Abschnitt 5.2.3.1 beschriebenen zufälligen Auswahl von Bastionen kombiniert wird. Die zufällige Verteilung der Verbindungen auf die parallelen Bastionen kann innerhalb der Funktion zum Verbindungsaufbau über das "Socks"-Protokoll implementiert werden und bleibt so vor dem Benutzer verborgen. Erst in den Fällen, in denen die zufällige Verteilung von Verbindungen zu einer sehr ungleichmäßigen Verteilung führt, braucht eine Umleitung von Verbindungen zu erfolgen. Zusätzlich wäre es denkbar, bereits bevor ein Umleiten von Zugriffen erforderlich wird, innerhalb des "Socks"-Protokolls Informationen zur Priorisierung der Bastionen an Clients zu schicken. Dadurch

[21] Trotz der zentralen Rolle dieser Bastion bei der Verteilung der Zugriffe handelt es sich weiterhin um ein dezentrales Verteilungsverfahren, da auch die anderen Bastionen Zugriffe umleiten können, wenn eine ungleichmäßige Lastverteilung ein Umleiten erfordert.

kann abhängig von der Last der Bastionen eine Gewichtung vorgenommen werden, so daß beim nächsten Zugriff mit einer höheren Wahrscheinlichkeit eine der weniger belasteten Bastionen ausgewählt wird. Welche Strategien sich hierfür am besten eignen, müssen Messungen an Prototyp-Implementationen zeigen.

Fazit: Durch eine Erweiterung des "Socks"-Protokolls kann eine dezentrale, lastabhängige Verteilung von Verbindungen auf parallele Bastionen implementiert werden. Wegen der Verlängerung der Verbindungsaufbauzeit eignet sich dieses Verfahren besser zur Verteilung längerer Verbindungen als zur Verteilung kurzer Verbindungen. Zudem muß dafür gesorgt werden, daß ein Umleiten nur selten erforderlich ist. Die Kombination mit dem in Abschnitt 5.2.3.1 beschriebenen Verfahren zur lastunabhängigen, zufälligen Auswahl einer Bastion ist hierbei besonders vielversprechend.

5.2.3.4 Filtern von Datagrammen

Bei den in den beiden vorigen Abschnitten vorgestellten Verfahren, die durch das Umleiten von Verbindungen eine dezentrale Lastverteilung implementieren, handelt es sich um Verfahren auf Anwendungsebene. Da die Untersuchung zentraler Verteilungsverfahren gezeigt hat, daß Verfahren auf den unteren Protokollschichten eine bessere Performanz erzielen als Verfahren auf Anwendungsebene, soll auch bei dezentralen Verteilungsverfahren die Verteilung auf den unteren Protokollschichten untersucht werden. Das im folgenden vorgestellte Verfahren selektiert Verbindungen durch das Filtern von Datagrammen auf IP-Ebene.

Für die dezentrale Verteilung von Verbindungen auf IP-Ebene wird ähnlich wie bei der zentralen Verteilung über NAT (s. Abschnitt 5.2.2.3) eine gemeinsame virtuelle Adresse für die parallelen Bastionen verwendet. Anders als bei der zentralen Verteilung erhalten jedoch bei der dezentralen Verteilung *alle Bastionen* den gesamten Datenverkehr, der für die gemeinsame virtuelle Adresse eintrifft. Jede Bastion wählt aus den eintreffenden Dateneinheiten anhand einer Verteilungsfunktion die zu bearbeitenden Dateneinheiten aus und ignoriert die übrigen Dateneinheiten.[22] Die Entscheidung wird somit auf allen Bastionen getroffen, jedoch erfolgt die anschließende Bearbeitung der Dateneinheiten nur durch eine der Bastionen. Das Senden der für die virtuelle Adresse eintreffenden Datagramme an alle Bastionen kann aufwandsarm realisiert werden, da in "Broadcast"-Netzen alle Stationen des Netzes ohnehin alle Dateneinheiten erhalten. In verbindungsorientierten ATM-Netzen kann dieselbe Funktionalität ebenfalls mit geringem Aufwand über Punkt-zu-Mehrpunkt-Verbindungen emuliert werden.

[22] Die Idee zu diesem Verfahren basiert auf der in Abschnitt 5.3 vorgestellten dezentralen Verteilung von Datagrammen auf parallele Packet Screens.

Bei der Filterung der Dateneinheiten durch die Verteilungsfunktion ist zu beachten, daß alle Dateneinheiten, die zu derselben TCP-Verbindung gehören, auch von derselben Bastion bearbeitet werden müssen. Die zu einer TCP-Verbindung gehörenden Dateneinheiten können über das 4-Tupel aus Absender-IP-Adresse, Empfänger-IP-Adresse, Absender-Port und Empfänger-Port identifiziert werden. Da die Empfänger-IP-Adresse die virtuelle Adresse der Bastionen enthält und der Empfänger-Port den jeweiligen Proxy identifiziert, sind diese beiden Merkmale konstant und daher nicht zur Auswahl von Verbindungen durch eine dezentrale Verteilungsfunktion geeignet. Für die Realisierung der dezentralen Verteilungsfunktion bestehen mehrere Möglichkeiten. Beispielsweise kann über der Absender-IP-Adresse und dem Absender-Port eines eintreffenden Datagramms eine Prüfsumme berechnet werden. Modulo der Anzahl der parallelen Bastionen ergibt sich aus der Prüfsumme ein Index, der die zuständige Bastion bestimmt. Die durchnumerierten Bastionen können damit unabhängig voneinander feststellen, welche Bastion für die Bearbeitung dieser Dateneinheit zuständig ist. Alternativ kann beim ersten TCP-Segment, das über das SYN-Flag identifiziert werden kann, eine Prüfsumme über die gesamte Dateneinheit oder über einen beliebigen festgelegten Teil der Dateneinheit berechnet werden. Die für diese Dateneinheit zuständige Bastion verarbeitet nicht nur diese erste Dateneinheit, sondern ergänzt im IP-Modul eine Filterregel, die auch alle folgenden Dateneinheiten dieser Verbindung selektiert. Für diese Filterregel wird dabei das 4-Tupel aus Absender-IP-Adresse, Empfänger-IP-Adresse, Absender-Port und Empfänger-Port verwendet, das eine Verbindung eindeutig identifiziert. Nach Beendigung der Verbindung wird die entsprechende Filterregel wieder entfernt.

Neben der Anforderung, daß alle Dateneinheiten, die zu derselben TCP-Verbindung gehören, auch von derselben Bastion bearbeitet werden müssen, kann es noch weitere Abhängigkeiten zwischen einzelnen Verbindungen geben, die auch eine Bearbeitung unterschiedlicher Verbindungen durch dieselbe Bastion erfordern. Beispielsweise müssen die Kontroll- und Datenverbindungen bei FTP [Postel et al. 85] von derselben Bastion kontrolliert werden. Weitere Abhängigkeiten, die eine Bearbeitung durch dieselbe Bastion erforderlich machen, können über "Ident" [St. Johns 93] oder ICMP [Postel 81a] entstehen. Falls in dem Einsatzbereich der Bastion diese zusätzlichen Abhängigkeiten bestehen, kann die Verteilungsfunktion lediglich auf der Basis der Absender-IP-Adresse ein Auswahl von Dateneinheiten vornehmen. Der gesamte Verkehr eines Absenders muß in diesem Fall von derselben Bastion bearbeitet werden.

Neben der Abhängigkeiten zwischen unterschiedlichen Verbindungen, die vom Client zum Proxy aufgebaut werden und eine gemeinsame Bearbeitung auf derselben Bastion erforderlich machen, muß auch sichergestellt werden, daß die Dateneinheiten der vom Proxy zum Server geöffneten Verbindungen nicht von der Verteilungsfunktion ignoriert werden. Da es zu jeder vom Proxy zum Server geöffneten Verbindung immer bereits eine vom Client zum Proxy geöffnete Verbindung

gibt, darf die Verteilungsfunktion auf die vom Proxy zum Server aufgebaute Verbindung nicht angewendet werden. Wenn die Verteilungsfunktion hingegen auch auf die vom Server eintreffenden Dateneinheiten angewendet wird, kann es vorkommen, daß die auf der Absender-IP-Adresse basierende Verteilungsfunktion eine andere Bastion auswählt als für die dazugehörende Verbindung vom Client zur Bastion. Dieses Problem läßt sich einfach vermeiden, indem für die vom Proxy zum Server aufgebaute Verbindung nicht die gemeinsame virtuelle Adresse der Bastionen verwendet wird. Stattdessen wird für diese Verbindung die IP-Adresse der Bastion verwendet, für die anders als bei der virtuellen Adresse keine Lastverteilung durchgeführt wird.

Dieses Verteilungsverfahren erreicht über die Prüfsummen-basierte Auswahl von Datagrammen eine zufällige Verteilung der Verbindungen. Im Gegensatz zu der in Abschnitt 5.2.3.1 diskutierten zufälligen Verteilung von Verbindungen ist bei dem hier diskutierten Verfahren keine direkte Beteiligung des Clients an der Verteilung erforderlich. Bei dem in Abschnitt 5.2.3.1 diskutierten Verfahren wählt der Client zufällig eine Bastion aus und öffnet zu dieser Bastion eine Verbindung. Das hier vorgestellte Verfahren erfordert keine direkte Beteiligung des Clients an dem Verteilungsverfahren, da der Client lediglich eine Verbindung zu der (virtuellen) Adresse der Bastion öffnen muß. Ungeachtet dieses Unterschieds ergeben beide Verfahren eine zufällige Verteilung von Verbindungen auf parallele Bastionen. Wie bereits im Abschnitt 5.2.3.1 dargestellt, kann eine zufällige Verteilung der Verbindungen auch zu einer sehr ungleichmäßigen Verteilung der Last führen. Abhängig von den IP-Adressen der Clients, die über die Bastionen Verbindungen aufbauen, kann die Modulo-Berechnung auf der Prüfsumme, die aus den Adreßinformationen berechnet wird, für alle Verbindungen dieselbe Bastion auswählen.

Eine Anpassung der Verteilungsfunktion, über die die Wahrscheinlichkeit geändert werden kann, mit der eine Bastion weitere Verbindungen erhält, ist zwar prinzipiell möglich. Die Änderung der Verteilungsfunktion darf aber nicht dazu führen, daß Dateneinheiten von bereits bestehenden Verbindungen anderen Bastionen zugeordnet werden, da dann eine Migration der betroffenen Proxy-Prozesse auf die neu zugeordnete Bastion erforderlich wäre. Eine Änderung der Verteilungsfunktion ist kaum möglich, wenn für alle Dateneinheiten eine Prüfsumme gebildet wird und somit für jede Dateneinheit erneut die zuständige Bastion bestimmt wird. Hingegen kann bei der Variante, bei der lediglich beim ersten Segment einer neuen TCP-Verbindung eine Prüfsumme berechnet wird und nachfolgende Dateneinheiten anhand von dynamisch hinzugefügten Filterregeln selektiert werden, die Verteilungsfunktion beliebig verändert werden. Dadurch wird es sogar möglich, einer überlasteten Bastion keine neuen Verbindungen mehr zuzuordnen, bis die Überlastsituation beendet ist. Bei jeder Änderung der Verteilungsfunktion muß darauf geachtet werden, daß die Änderungen synchron auf allen parallelen Bastionen aktiviert werden, da sonst die Gefahr besteht, daß eine

Verbindung zwei unterschiedlichen Bastionen zugeordnet wird oder eine Verbindung von keiner Bastion bearbeitet wird.

Fazit: Das in diesem Abschnitt diskutierte dezentrale Verfahren auf IP-Ebene erreicht eine zufällige Verteilung von Verbindungen auf parallele Bastionen. Bei diesem Verfahren bestimmt jede Bastion selbst über die Berechnung einer Prüfsumme, ob eine empfangene Dateneinheit bearbeitet werden muß oder ob eine andere Bastion für diese Dateneinheit zuständig ist. Da die Verteilung von den Bastionen selbst durchgeführt wird, erfordert dieses Verfahren keine Unterstützung durch den zugreifenden Client, wodurch herkömmliche Clients verwendet werden können. Die Berechnung der Prüfsumme durch eine Bastion ist einfach und kann falls erforderlich sogar effizient in Hardware implementiert werden. Daher verspricht dieses Verfahren eine hohe Performanz. Durch die dezentrale Verteilung der Verbindungen wird zudem eine gute Skalierbarkeit des Verfahrens ermöglicht. Im Gegensatz zu dem im Abschnitt 5.2.3.3 diskutierten Umleiten von Verbindungen führt das hier vorgestellte Verfahren nicht zu einer Verlängerung der Verbindungsaufbauzeit und hat auch keine Auswirkungen auf den Transaktionsdurchsatz einzelner Bastionen.[23] Trotz dieser Vorteile kann mit diesem Verfahren nur eine zufällige Verteilung der Verbindungen erreicht werden. Besonders bei Anwendungsbereichen, in denen nur wenige, sehr lange Verbindungen verteilt werden müssen, kann die ungleichmäßige Verteilung der Verbindungen in einer suboptimalen Auslastung der parallelen Bastionen resultieren.

5.3 Parallele Packet Screens

Eine Zugriffskontrolle durch Packet Screens erfordert deutlich geringeren Aufwand als eine Kontrolle auf Anwendungsebene durch Proxies. Die Messungen der Performanz von Packet Screens in Abschnitt 4.3 haben bestätigt, daß der von Packet Screens erreichte Datendurchsatz für 155 MBit/s ATM-Netze ausreicht. Hinsichtlich des Datagrammdurchsatzes stellen Packet Screens jedoch bei 155 MBit/s ATM-Netzen bereits einen Engpaß dar. Durch die leichte Skalierbarkeit von ATM steigen die Datendurchsätze von ATM-Netzen jedoch schneller als die Rechenleistung von Prozessoren. Für die Zukunft werden daher Konzepte für skalierbare Packet Screens benötigt, mit denen sich der Datagrammdurchsatz und

[23] Vor dem Umleiten einer Verbindung ist bei dem im Abschnitt 5.2.3.3 diskutierten Verfahren zunächst die Annahme der Verbindung erforderlich. Da bei kurzen Transaktionen die Verbindungsannahme einen Engpaß bildet, sinkt beim Umleiten von Verbindungen der Transaktionsdurchsatz der einzelnen Bastion. Da die Verteilung bei dem in diesem Abschnitt vorgestellten Verfahren bereits auf IP-Ebene stattfindet, werden bei allen angenommenen Verbindungen anschließend Transaktionen durchgeführt, und der Transaktionsdurchsatz wird nicht durch das Umleiten von Verbindungen reduziert.

der Datendurchsatz von Packet Screens leicht an die steigende Leistungsfähigkeit der Netze anpassen lassen. Durch eine Verteilung der Filterfunktionen auf mehrere parallele Packet Screens wird es möglich, einen höheren Durchsatz zu erreichen, als es mit einer einzelnen Packet Screen allein möglich wäre. Das Ziel dieser Parallelisierung muß ein leicht skalierbares Konzept sein, bei dem mit geringen Aufwand der Durchsatz durch Hinzufügen zusätzlicher paralleler Packet Screens erhöht werden kann.

Packet Screens sind üblicherweise zustandslos,[24] so daß Filterentscheidungen für jede Dateneinheit unabhängig von allen vorher empfangenen Dateneinheiten getroffen werden können. Daher ist es möglich, nach dem Prinzip der *Paketparallelität* (s. Abschnitt 5.1) eintreffende Dateneinheiten parallel durch unterschiedliche Packet Screens kontrollieren zu lassen. Auch die Verarbeitung von Dateneinheiten, die zu derselben Verbindung gehören, kann dabei durch unterschiedliche Packet Screens erfolgen.

Prinzipiell sind auch bei der Parallelisierung von Packet Screens unterschiedliche Parallelisierungsansätze möglich. Ähnlich wie bei Konzepten zur Verteilung von Verbindungen auf parallele Bastionen können auch bei Packet Screens zentrale oder dezentrale Verfahren eingesetzt werden. Zentrale Verteiler wie zum Beispiel die in Abschnitt 5.2.2.3 und 5.2.2.4 diskutierte Verteilung über NAT oder über einen "TCP-Router" erreichen jedoch von der Größenordnung her nur einen ähnlichen Durchsatz wie eine einzelne Packet Screen und können daher keine nennenswerte Leistungssteigerung von Packet Screens erzielen. Lediglich die ATM-spezifischen Konzepte der Verteilung über NHRP (s. Abschnitt 5.2.2.6) und der Verteilung durch einen Signalisierungskontroller (s. Abschnitt 5.2.2.7) können als zentrale Verteilungsverfahren auch für Packet Screens eingesetzt werden, da hier der Verteiler nicht selbst an der nachfolgenden Datenübertragung beteiligt ist. Der Einsatz dieser beiden Verfahren zur Verteilung von virtuellen Verbindungen auf parallele Packet Screens unterscheidet sich konzeptionell nicht von dem bereits diskutierten Einsatz bei parallelen Bastionen und soll daher an dieser Stelle nicht weiter diskutiert werden.

Zur Verteilung von Datagrammen auf parallele Packet Screens sind dezentrale Verfahren besser geeignet, da hier kein zentraler Engpaß durch einen zentralen Verteiler entsteht. Dabei sind jedoch dezentrale Verteilungsverfahren, die eine Unterstützung durch den zugreifenden Client erfordern, nicht geeignet, da der Einsatz von Packet Screens transparent ist für die kommunizierenden Systeme, deren Datenverkehr kontrolliert wird (s. Abschnitt 2.3.1.1). Daher werden

[24] "Dynamische Packet Screens" (s. Abschnitt 2.3.1.1), die Filterregeln während der Laufzeit hinzufügen und entfernen, werden in diesem Abschnitt nicht untersucht. Da "dynamische Packet Screens" die zu filternden Dateneinheiten verbindungsorientiert bearbeiten, sind die im Abschnitt 5.2 beschriebenen Konzepte für parallele Bastionen auch bei "dynamischen Packet Screens" für eine Parallelisierung anwendbar.

Packet Screens im Gegensatz zu Proxies nicht direkt durch die Kommunikationspartner adressiert. Dezentrale Verfahren wie die zufällige Auswahl einer Bastion durch einen Client (s. Abschnitt 5.2.3.1) oder wie das Umleiten von Verbindungen (s. Abschnitt 5.2.3.3), die eine Unterstützung des Clients erfordern, sind daher für die Verteilung von Dateneinheiten auf parallele Packet Screens ungeeignet.

Für eine Verteilung von Datagrammen auf parallele Packet Screens sind Verfahren gut geeignet, bei denen alle parallelen Packet Screens den gesamten Datenverkehr erhalten und jede Packet Screen unabhängig von den anderen Packet Screens entscheidet, ob die Bearbeitung einer eintreffenden Dateneinheit erforderlich ist. Bereits im Abschnitt 5.2.3.4 wurde ein ähnliches Verfahren zur Verteilung von Verbindungen auf parallele Bastionen vorgestellt, bei dem jede Bastion über eine einfache Modulo-Berechnung auf Basis der in der Dateneinheit enthaltenen (Adreß-)Informationen feststellen kann, ob sie für die Bearbeitung dieser Dateneinheit verantwortlich ist. Dasselbe Konzept eines Vorfilters, der alle Dateneinheiten selektiert, die von der Bastion bearbeitet werden müssen, und die übrigen Dateneinheiten ignoriert, kann auch bei Packet Screens angewendet werden.[25]

Das Hauptproblem bei dem in Abschnitt 5.2.3.4 vorgestellten Verfahren entsteht durch die Anforderung, daß alle Dateneinheiten, die zu derselben Verbindung gehören, auch von derselben Bastion bearbeitet werden müssen. Da Packet Screens, wie bereits diskutiert, Dateneinheiten unabhängig voneinander bearbeiten, bestehen bei einer Verteilung auf parallele Packet Screens geringere Anforderungen an die Verteilungsfunktion als bei einer Verteilung auf Bastionen. Die dezentrale Verteilungsfunktion muß eindeutig genau einer der parallelen Packet Screens die Bearbeitung einer Dateneinheit zuweisen. Der Aufwand für die Berechnung der Verteilungsfunktion muß außerdem klein sein im Verhältnis zur nachfolgenden Verarbeitung der Dateneinheit durch die Packet Screen, andernfalls kann auch bei dezentralen Verteilungsverfahren die Verteilung zum Engpaß werden.

Ohne bei der Verteilungsfunktion eine Zuordnung aller zu einer Verbindung gehörenden Dateneinheiten zu derselben Packet Screen gewährleisten zu müssen, ist die Verteilungsfunktion nicht mehr auf das 4-Tupel aus Absender-IP-Adresse, Empfänger-IP-Adresse, Absender-Port und Empfänger-Port festgelegt. Stattdessen können beliebige Teile der Dateneinheit für eine Auswertung durch die Verteilungsfunktion genutzt werden. Für die Verteilungsfunktion eignen sich besonders die Teile von Dateneinheiten, deren Inhalte bei unterschiedlichen Dateneinheiten möglichst gut streuen, da nur dadurch eine möglichst gleichmäßige Verteilung der eingehenden Dateneinheiten auf die parallelen Packet Screens erreicht werden kann. Informationen wie IP-Adressen oder Portnummern entfallen daher,

[25] Die dezentrale Verteilung von Datagrammen auf parallele Packet Screens durch eine einfache Modulo-Berechnung wurde von Carsten Benecke [Ellermann et al. 98b] vorgeschlagen und in einem Prototyp implementiert.

weil sie zu ungleichmäßig verteilt sind. Wäre die Entscheidung etwa abhängig vom Ziel-Port, dann würde der gesamte Verkehr zu einem bestimmten Dienst über dieselbe Packet Screen geführt. Eine viel gleichmäßigere Verteilung ist auf Basis der "IP-Header"-Prüfsumme möglich. Die "IP-Header"-Prüfsumme wird über eine einfache Einer-Komplement-Arithmetik über alle Felder des "IP-Headers" berechnet. Da jede IP-Dateneinheit vom Sender mit einem "Identification"-Feld versehen wird, das sich für jede Dateneinheit unterscheidet, unterscheidet sich auch die Prüfsumme selbst bei IP-Dateneinheiten, die zwischen denselben Kommunikationspartnern ausgetauscht werden. Eine Prototyp-Implementation mit dem Verteilungsschlüssel "<*'IP-Header'-Prüfsumme*> `Modulo` <*Anzahl der parallelen Packet Screens*>" erreichte bei Messungen eine sehr gleichmäßige Verteilung der Dateneinheiten auf zwei parallele Packet Screens.[26] Obwohl die zur Berechnung der Prüfsumme verwendete Funktion sehr einfach ist und nicht wie kryptographische Prüfsummen-Verfahren selbst bei geringen Unterschieden der Ausgangsdaten eine gleichmäßige Verteilung der berechneten Prüfsummen garantiert, ist die durch diese Funktion erreichte Verteilung der Datagramme auf die Packet Screens sehr gleichmäßig. Da zudem die "IP-Header"-Prüfsumme bereits vom Sender berechnet wird, ist im Gegensatz zu kryptographischen Prüfsummen, die durch die Verteilungsfunktion auf allen parallelen Packet Screens selbst berechnet werden müssen, der durch die Verteilungsfunktion bedingte Verarbeitungsaufwand auf den parallelen Packet Screens bei dem hier eingesetzten Verfahren viel geringer. Ein zusätzlicher Aufwand für theoretisch bessere Verfahren lohnt daher nicht.

Das dargestellte Verfahren eignet sich besonders für "Broadcast"-Netze (Ethernet, Fast-Ethernet, Gigabit-Ethernet, FDDI), bei denen jedes an das Netz angeschlossene System ohnehin alle Dateneinheiten empfängt. Abbildung 5.11 zeigt einen Aufbau für Fast-Ethernet, bei dem vier Packet Screens parallel zwischen zwei "Multiport-Repeatern", die auch kurz als "Hub" bezeichnet werden, installiert sind. Die beiden "Hubs" bilden jeweils eine "Broadcast"-Domäne, so daß jede der abgebildeten Packet Screens alle eingehenden Datagramme erhält.[27] Nach dem oben dargestellten Verteilungsverfahren wählt jede Packet Screen einen Teil

[26] Bei einem Test von etwa 6 Stunden wurden 29.487.830 Datagramme mit unterschiedlichen Datagrammlängen generiert und über die beschriebene Modulo-Funktion auf zwei Packet Screens verteilt. Von der optimalen Verteilung gab es dabei lediglich eine Abweichung um ± 415 Datagramme.

[27] Es gibt zwei unterschiedliche Ansätze, eine Packet Screen zu implementieren. Packet Screens können einerseits als "Bridge" Dateneinheiten auf Schicht 2 übertragen. Alternativ kann eine Packet Screen auch mit einem "Router" kombiniert werden und leitet dann Dateneinheiten auf Schicht 3 des OSI-Modells weiter. Für eine Packet Screen, die mit einer "Bridge" kombiniert ist, reicht der dargestellte Aufbau bereits aus. Bei Packet Screens, die mit einem "Router" kombiniert sind, muß zusätzlich für alle Packet Screens eine gemeinsame Ethernet-(Broadcast-)Adresse konfiguriert werden, über die Systeme in den jeweiligen Netzen Ethernet-Pakete gleichzeitig an alle Packet Screens senden können.

der Datagramme zur weiteren Verarbeitung aus und leitet die Datagramme nach Kontrolle der Filterregeln schließlich in das andere Netz weiter.

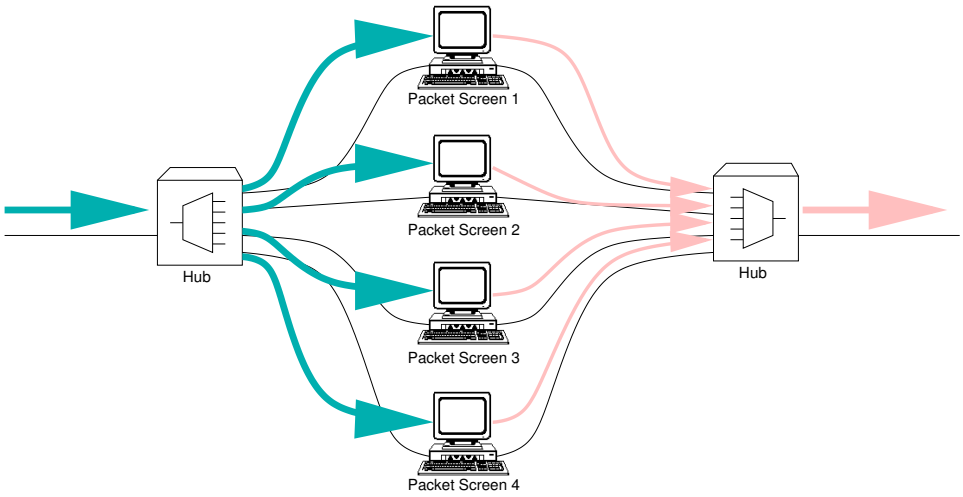


Abbildung 5.11: Parallele Packet Screen für Ethernet

In ATM-Netzen kann die "Broadcast"-Funktionalität durch Punkt-zu-Mehrpunkt-Verbindungen emuliert werden, so daß dieses Verteilungskonzept auch für parallele Packet Screens in ATM-Netzen eingesetzt werden kann. Da die Verteilung der eintreffenden Zellen auf mehrere ausgehende Schnittstellen effizient in der Hardware eines Switches implementiert wird, stellen Punkt-zu-Mehrpunkt-Verbindungen keinen Engpaß dar. Da jede Packet Screen alle Datagramme erhalten muß, ist es jedoch erforderlich alle Schnittstellen des Switches für denselben Durchsatz auszulegen. Diese gilt für Schnittstellen, an die eine Packet Screen angeschlossen ist, sowie für die Schnittstelle, an die das interne bzw. das externe Netz angeschlossen ist. Soll beispielsweise der Verkehr, der über eine 622 MBit/s Schnittstelle hereinkommt, von 8 Packet Screens kontrolliert werden, so braucht jede Packet Screen nur im Durchschnitt einen Durchsatz von maximal 78 MBit/s zu erreichen. Für die übertragenen Dateneinheiten würde also jeweils eine 155 MBit/s Schnittstelle ausreichen. Da aber auch die Dateneinheiten zu allen Packet Screens gesendet werden müssen, die anschließend von der dezentralen Verteilungsfunktion verworfen werden, müssen alle Packet Screens über 622 MBit/s Schnittstellen an den Switch angeschlossen werden.

Da die eintreffenden Dateneinheiten gleichmäßig auf die parallelen Packet Screens verteilt werden, ist die korrekte Funktion *aller* Packet Screens erforderlich, damit eine störungsfreie Kommunikation über die parallelen Packet Screens erfolgen kann. Bereits der Ausfall einer Packet Screen bei der in Abbildung 5.11 dargestellten Konfiguration mit 4 parallelen Packet Screens führt dazu, daß jedes

vierte Datagramm verloren geht. Eine Kommunikation über das Gesamtsystem hinweg ist damit kaum noch möglich. Die Ausfallwahrscheinlichkeit der Gesamtkonfiguration ist somit viermal so hoch wie die Ausfallwahrscheinlichkeit einer einzelnen Packet Screen. Durch den Einsatz von Verfahren zur Erkennung von Ausfällen ist es jedoch möglich, die Verteilungsfunktion nach einem Ausfall an die neue Anzahl der Packet Screens anzupassen. Nach dem Ausfall eines einzelnen Systems übernehmen dann die verbleibenden Systeme den Verkehr des ausgefallenen Systems. Durch diese Möglichkeit zur Umkonfigurierung wird eine hohe Ausfallsicherheit erreicht. Dabei ist die Performanz nach Ausfall eines einzelnen Systems lediglich etwas geringer.

Das dargestellte Problem, daß ohne eine Anpassung der Verteilungsfunktion bereits der Ausfall einer Packet Screen ausreicht, um das Gesamtsystem zu blockieren, kann auch von Angreifern gezielt für "Denial of Service"-Angriffe ausgenutzt werden. Ein Angreifer, der die Verteilungsfunktion kennt, kann gezielt Dateneinheiten konstruieren, die von der Verteilungsfunktion immer derselben Packet Screen zugeordnet werden. Da lediglich eine der parallelen Packet Screens überlastet zu werden braucht, um Paketverluste zu verursachen, reicht für den Angreifer eine viel geringere Datenmenge aus, als es zur gleichzeitigen Überlastung aller Packet Screens erforderlich wäre. Bei der angegriffenen Packet Screen kommt es in Folge der Überlastung zu Paketverlusten, die auch reguläre Verbindungen beeinträchtigen, da Dateneinheiten erneut übertragen werden müssen.[28] Ein einfaches Verfahren, das beim Ausfall einer Packet Screen die Verteilungsfunktion an die geringere Anzahl der Packet Screens anpaßt und so die Last auf die verbleibenden Packet Screens verteilt, bietet keinen Schutz gegen den dargestellten "Denial of Service"-Angriff. Die Umkonfigurierung der Verteilungsfunktion hat lediglich das Ergebnis, daß anschließend eine andere Packet Screen von dem Angriff getroffen wird und somit ein anderes System überlastet wird. Dadurch werden innerhalb kurzer Zeit immer wieder neue Rekonfigurierungen der Verteilungsfunktion notwendig, da immer wieder andere Packet Screens ausfallen.

Abhängig von der Art der vom Angreifer gesendeten Dateneinheiten kann diesem Angriff mit zwei unterschiedlichen Strategien entgegengewirkt werden. Damit Dateneinheiten von der Verteilungsfunktion immer derselben Packet Screen zugeordnet werden, muß der Angreifer entweder in jeder Dateneinheit dieselbe "IP-Header"-Prüfsumme verwenden. Mit etwas höherem Aufwand kann ein Angreifer auch Dateneinheiten mit unterschiedlichen Prüfsummen generieren, deren Prüfsummen von der Verteilungsfunktion trotzdem derselben Packet Screen zu-

[28] Bei dieser erneuten Übertragung wird eine neue IP-Dateneinheit generiert, da die erneute Übertragung durch TCP initiiert wird oder bei UDP durch die Anwendung selbst initiiert werden muß. Die Prüfsumme der neuen Dateneinheit wird sich von der Prüfsumme der verlorengegangenen Dateneinheit unterscheiden, so daß die Dateneinheit durch die Verteilungsfunktion auch einer anderen Packet Screen zugewiesen werden kann.

gewiesen werden. Wenn bei jeder Dateneinheit dieselbe Prüfsumme verwendet wird, kann die Verteilungsfunktion so geändert werden, daß lediglich Dateneinheiten mit genau dieser einen Prüfsumme einer Packet Screen zugeordnet werden. Die Verarbeitung aller anderen Dateneinheiten kann dann über die verbleibenden Packet Screens erfolgen und ist von dem Angriff somit nur geringfügig betroffen. Da die Wahrscheinlichkeit sehr hoch ist, daß reguläre Dateneinheiten, die zufällig die vom Angreifer verwendete Prüfsumme enthalten, auf der überlasteten Packet Screen verloren gehen und daher erneut übertragen werden müssen, ist es sogar noch besser, die Verteilungsfunktion so zu ändern, daß Dateneinheiten mit der beim Angriff verwendeten Prüfsumme von *keiner* Packet Screen verarbeitet werden. Dieses Verfahren führt auch bei regulären Datenübertragungen zu Paketverlusten. Da die Wahrscheinlichkeit, eine Dateneinheit mit genau dieser Prüfsumme zu generieren, jedoch gering ist, stellt der einmalige Paketverlust mit dem erforderlichen erneuten Senden der Daten kein Problem dar.

Alternativ ist es für einen Angreifer aber auch möglich, bei jeder Dateneinheit eine andere Prüfsumme zu erzeugen, die trotzdem jedesmal zu der Verarbeitung der jeweiligen Dateneinheit auf derselben Packet Screen führt. In diesem Fall empfiehlt sich eine andere Strategie, diesem Angriff entgegenzuwirken. Durch eine Rekonfigurierung muß dann versucht werden, die vom Angreifer gesendeten Dateneinheiten möglichst gleichmäßig auf die zur Verfügung stehenden Packet Screens zu verteilen. Da von außerhalb nicht erkennbar ist, von welcher Packet Screen eine Dateneinheit verarbeitet wird, verliert der Angreifer hierdurch seinen Vorteil, daß er nur eine der parallelen Packet Screens überlasten muß. Um nach der Rekonfigurierung noch eine Überlastung zu erzeugen, ist der Angreifer gezwungen, gleichzeitig alle Packet Screens zu überlasten. Dafür ist die Generierung einer sehr viel höheren Menge von Dateneinheiten erforderlich.

Fazit: Dieser Abschnitt hat gezeigt, daß die Paketparallelität gut für die Parallelisierung von Packet Screens geeignet ist. Mit den hier vorgestellten dezentralen Verteilungsverfahren kann jede der parallelen Packet Screens durch eine einfache Modulo-Berechnung über der "IP-Header"-Prüfsumme feststellen, welche Packet Screen für die Bearbeitung der Dateneinheit verantwortlich ist. Da der für die Verteilung erforderliche Aufwand sehr gering ist, kann dieses Verfahren sehr effizient eingesetzt werden.

Da Dateneinheiten nach dem Konzept der Paketparallelität unabhängig voneinander auf parallele Packet Screens verteilt werden, eignet sich die hier beschriebene Lösung nicht für Packet Screens, die Entscheidungen anhand eines Kontextes mehrerer Dateneinheiten treffen. Für die hiervon betroffenen dynamischen Packet Screens eignen sich besser Konzepte zur Verteilung von Verbindungen, wie die

in Abschnitt 5.2.3.4 vorgestellte dezentrale Auswahl von Verbindungen durch das Filtern von Datagrammen.

5.4 Verringerung der Verbindungsaufbauzeiten

Zur Beseitigung der bei Proxies bestehenden Engpässe hinsichtlich des Daten- und Transaktionsdurchsatzes wurden im Abschnitt 5.2 Parallelisierungskonzepte für den Einsatz von Proxies auf parallelen Bastionen diskutiert. Neben den Engpässen beim Daten- und Transaktionsdurchsatz zeigen die Messungen im Abschnitt 4.4 zusätzlich starke Auswirkungen von Proxies auf die Verbindungsaufbauzeit. Für die längeren Verbindungsaufbauzeiten gibt es mehrere Ursachen. Da bei der Kommunikation über einen Proxy immer zwei Verbindungen (Client→Proxy und Proxy→Server) aufgebaut werden müssen, entsteht prinzipbedingt durch den Einsatz eines Proxies eine längere Verbindungsaufbauzeit. Darüber hinaus haben die Annahme einer Verbindungen beim Proxy (insbesondere beim `fork()` von Prozessen), der Aufwand durch eine Authentisierung (insbesondere bei kryptographischen Authentisierungsverfahren) sowie das Audit deutliche Auswirkungen auf die Verbindungsaufbauzeiten. Zusätzlich werden zur Steuerung des Verbindungsaufbaus vom Proxy zum Server gegebenenfalls weitere Protokolle (z. B. "Socks" [Leech et al. 96]) eingesetzt, die durch mehrere Umläufe ebenfalls Verzögerungen erzeugen können. Da die Verbindungsaufbauzeiten in Abschnitt 4.4 bei exklusiver Nutzung des Proxies durch einen einzelnen Client gemessen wurden, können durch Konzepte, die auf der Verbindungsparallelität basieren keine geringeren Verbindungsaufbauzeiten erzielt werden. Durch Parallelisierungsverfahren kann lediglich dafür gesorgt werden, daß durch parallel von anderen Clients durchgeführte Zugriffe keine *zusätzlichen* Verzögerungen verursacht werden. Da in Hochgeschwindigkeitsnetzen die Bedeutung der Verbindungsaufbauzeit gegenüber der daran anschließenden Zeit für die eigentliche Datenübertragung zunimmt, sind Maßnahmen zur Verringerung der Verbindungsaufbauzeit bei der Kommunikation über Proxies dringend erforderlich.

Ähnlich wie bei den im Abschnitt 5.2 diskutierten Parallelisierungskonzepten hat der Erfolg von HTTP und die damit verbundenen Performanzprobleme auch den wesentlichen Anstoß für die Entwicklung von Verfahren zur Verringerung der Verbindungsaufbauzeiten gegeben. In wie weit die zur Verringerung der Verbindungsaufbauzeiten bei HTTP entwickelten Ansätze auch bei Proxies eingesetzt werden können, soll dieser Abschnitt zeigen.

5.4.1 Offenhalten von Verbindungen und "Pipelining"

Bei HTTP [Berners-Lee et al. 96] wird für jeden Zugriff eine eigene TCP-Verbindung aufgebaut und nach dem Zugriff wieder abgebaut. Da ein Client üblicherweise nicht nur einen einzelnen Zugriff auf einen Web-Server durchführt, müssen für die auf der Web-Seite enthaltenen "Icons" und Bilder sowie für den Zugriff auf folgende Web-Seiten jedesmal erneut Verbindungen auf- und abgebaut werden. Bei jedem Zugriff treten dabei Verbindungsaufbauzeiten auf. Um den wiederholten TCP-Verbindungsaufbau zu vermeiden, wird bei "Persistent-Connection"-HTTP (P-HTTP) [Mogul 95] und bei HTTP/1.1 [Fielding et al. 97] dieselbe TCP-Verbindung für mehrere aufeinanderfolgende Zugriffe verwendet. Durch die zusätzlich angewendete "Pipelining"-Technik kann der Client bereits weitere Zugriffe an den Server senden, ohne auf die Beendigung bisher gesendeter Zugriffe zu warten. Der Server bearbeitet die Zugriffe sequentiell. Damit der Client die auf derselben Verbindung gesendeten Antworten des Servers voneinander unterscheiden kann, markiert der Server das Ende jeder Antwort.

Der durch P-HTTP oder HTTP/1.1 erreichte Performanzgewinn geht zu Lasten einer höheren Komplexität bei Client und Server. Eine besondere Beschränkung dieser Lösungen ist, daß die Antworten nacheinander vom Server gesendet werden. Einzelne sehr lange Antworten können dadurch die nachfolgenden Antworten sehr stark verzögern. Zudem gibt es keine Möglichkeit für den Client einmal gesendete Zugriffe abzubrechen, ohne dafür die Verbindung zu beenden. Trotz dieser Beschränkungen lassen Simulationen des Performanzverhaltens von P-HTTP [Mogul 95] und Messungen mit HTTP/1.1 [Nielsen et al. 97] Performanzsteigerungen durch Einsparungen des erneuten Verbindungsaufbaus und eine deutliche Entlastung der HTTP-Server erwarten.

Beim Einsatz von Firewalls machen sich die Einsparungen durch den entfallenden zusätzlichen Verbindungsaufbau für nachfolgende Zugriffe besonders bemerkbar, da die Verbindungsaufbauzeiten durch die zusätzliche Bearbeitung im Proxy und durch den doppelten Verbindungsaufbau (Client→Proxy und Proxy→Server) besonders hoch sind. Darüber hinaus wird durch diese neuen Protokolle eine Entlastung der Proxies erreicht.

Fazit: Da es sich bei P-HTTP und HTTP/1.1 um Anwendungsprotokolle für einen speziellen Einsatzbereich handelt, sind die beiden Lösungen nicht allgemein zur Verringerung der Verbindungsaufbauzeiten bei Proxies einsetzbar. Auch wenn HTTP einen großen Anteil am Verkehrsaufkommen eines Firewalls hat, müssen eine Reihe weiterer Protokolle unterstützt werden, für die ebenfalls eine Verringerung der Verbindungsaufbauzeiten wünschenswert wäre.

5.4.2 Leichtgewichtiges TCP

Zum Aufbau einer TCP-Verbindung muß zuerst ein "3-Way-Handshake" (s. Abb. 2.2, Seite 14) durchgeführt werden. Erst nach einem erfolgreichen "Handshake" können die Kommunikationspartner Daten austauschen. TCP stellt Mechanismen zum erneuten Übertragen verlorengegangener Dateneinheiten, zur Wiederherstellung der Reihenfolge vertauschter Dateneinheiten beim Empfänger und zur Flußkontrolle zur Verfügung. Alternativ zum verbindungsorientierten TCP können Anwendungen auch Daten über das verbindungslose UDP austauschen. Da UDP keinen Verbindungsaufbau erfordert, können Daten bereits beim ersten UDP-Datagramm gesendet werden. Dem verbindungslosen UDP fehlen jedoch wichtige Eigenschaften, die bei TCP vorhanden sind, so daß Anwendungen, die UDP einsetzen, z. B. selbst über geeignete Mechanismen zur erneuten Übertragung, zur Wiederherstellung der Reihenfolge und zur Flußkontrolle verfügen müssen.

"Transaction TCP" (T/TCP) [Braden 92, Braden 94] ist eine Erweiterung von TCP, die auch über das Ende einer Verbindung hinaus Zustandsinformationen zwischenspeichert, um bei folgenden Verbindungen zu demselben System bereits mit dem ersten Segment des TCP-Verbindungsaufbaus Daten übertragen zu können. Mit Hilfe eines sogenannten Verbindungszählers ("Connection Count", CC) werden alle Verbindungen zwischen zwei Kommunikationspartnern durchgezählt, so daß für nachfolgende Verbindungen monoton steigende CC Werte verwendet werden. Über die Monotonie-Eigenschaft des im TCP-Segment mitgeschickten CCs kann der Empfänger bereits beim ersten TCP-Segment (SYN) feststellen, daß es sich nicht um ein altes, außerhalb der Reihenfolge empfangenes Segment handelt. Im ersten Segment gesendete Daten können daher bereits an die zuständige Anwendung (z. B. einen Web-Server) weitergeleitet werden. Im Idealfall kann dadurch bereits beim zweiten TCP-Segment (SYN-ACK) des "Handshakes" die Antwort der Anwendung an den Client zurückgesendet werden. Wenn von der Anwendung nur geringe Datenmengen übertragen werden und daher sowohl für die Anfrage als auch die Antwort jeweils ein TCP-Segment ausreicht, kann bereits im dritten Segment der Verbindungsabbau bestätigt werden (FIN-ACK). Bei T/TCP können somit drei Segmente ausreichen, um beispielsweise eine Web-Seite oder ein "Icon" zu übertragen. Wenn größere Datenmengen zu übertragen sind, kann die mit T/TCP aufgebaute Verbindung aber auch wie eine herkömmliche TCP-Verbindung verwendet werden. Da T/TCP bereits während des "3-Way-Handshakes" Nutzdaten überträgt und dadurch weniger Segmente zur Übertragung derselben Datenmenge benötigt als TCP, wird T/TCP auch als "leichtgewichtiges" Protokoll bezeichnet.

Vorteil dieser Lösung ist, daß keine Änderungen des Anwendungsprotokolls erforderlich sind, so daß Anwendungen nicht wie bei P-HTTP erforderlich selbst das "Multiplexing" übernehmen müssen. Stattdessen können Anwendungen

T/TCP-Verbindungen ebenso unabhängig voneinander verwenden wie herkömmliche TCP-Verbindungen. Von der Funktion her bildet T/TCP einen Mittelweg zwischen TCP und UDP. Da bei T/TCP kein "3-Way-Handshake" erforderlich ist, bevor Daten an die Anwendung weitergeleitet werden, ist T/TCP sehr viel anfälliger gegenüber "IP-Spoofing"-Angriffen (s. Abschnitt 2.1.1.2) als herkömmliches TCP. [FreeBSD-SA-98:03]

Im Anwendungsbereich von Firewalls kann T/TCP ebenfalls zur Verringerung der Verbindungsaufbauzeiten eingesetzt werden. Da T/TCP eine Erweiterung von TCP ist, kann T/TCP zur Beschleunigung des Verbindungsaufbaus bei diversen TCP-basierten Anwendungen eingesetzt werden, ohne Änderungen der Anwendungsprotokolle zu erfordern. Auch ohne Veränderungen beim Anwendungsprotokoll werden für den Einsatz von T/TCP Anpassungen der Applikationen erforderlich, da die herkömmlichen Funktionsaufrufe für den TCP-Verbindungsaufbau (`connect()` und `accept()`) keine gleichzeitige Datenübertragung ermöglichen und daher den wesentlichen Vorteil von T/TCP — das Übertragen von Daten beim "Handshake" — nicht ausnutzen können.

Da T/TCP-Verbindungen voneinander unabhängig sind, können sie auch von Proxies unabhängig voneinander bearbeitet werden. Die vorhandenen Konzepte für Proxies können somit bei T/TCP unverändert eingesetzt werden. Die leichtere Angreifbarkeit von T/TCP-Verbindungen durch "Spoofing"-Angriffe stellt nur teilweise ein Problem für Firewalls dar, weil auch herkömmliche TCP-Verbindungen durch "Spoofing"-Angriffe gefälscht oder durch "TCP-Hijacking"-Angriffe übernommen werden können. Daher sind die schwachen Mechanismen von TCP zur Sicherung der Authentizität und Integrität ohnehin für Firewalls nicht ausreichend, so daß zusätzliche kryptographische Verfahren (z. B. SSL) eingesetzt werden müssen. Da jede neue Verbindung, ob mit T/TCP oder herkömmlichen TCP geöffnet, eine erneute kryptographische Authentisierung[29] erfordert, ist der Performanzgewinn durch die niedrigere Verbindungsaufbauzeit bei T/TCP nur gering im Vergleich zu der aufwendigen kryptographischen Authentisierung, die sowohl bei T/TCP als auch bei TCP erforderlich ist. Aus diesem Grund ist T/TCP nur dann sinnvoll im Zusammenhang mit Firewalls einsetzbar, wenn nicht für jede Verbindung eine erneute kryptographische Authentisierung erforderlich ist. Auf eine kryptographische Authentisierung bei jeder Verbindung kann verzichtet werden, wenn zum Beispiel alle internen Clients autorisiert sind, herausgehende Verbindungen zu öffnen,[30] oder wenn die Authentisierung über andere Verfahren (z. B. IPSEC oder PGP) sichergestellt werden kann.

[29] Abhängig vom Einsatzbereich kann eine gegenseitige Authentisierung zwischen Client und Proxy erforderlich sein, oder es muß sich lediglich der Client beim Proxy authentisieren.

[30] Wenn alle Clients berechtigt sind, herausgehende Verbindungen zu öffnen, wird auf eine Authentisierung der Clients oft verzichtet.

Fazit: T/TCP kann auch zur Verringerung der Verbindungsaufbauzeiten bei Firewalls eingesetzt werden. Durch die leichte Angreifbarkeit von T/TCP sind die Anwendungsbereiche jedoch auf spezielle Umgebungen beschränkt.

5.4.3 "Multiplexing" von Verbindungen

Durch das "Multiplexing" mehrerer Verbindungen über dieselbe TCP-Verbindung werden wesentliche Eigenschaften der in den vorigen beiden Abschnitten diskutierten Ansätze integriert. Das WebMUX-Protokoll [Gettys et al. 98] ist ein einfaches Protokoll, mit dem mehrere Verbindungen, die im folgenden als WebMUX-Verbindungen bezeichnet werden, gleichzeitig über dieselbe TCP-Verbindung übertragen werden können.

Bei WebMUX wird zunächst eine herkömmliche TCP-Verbindung aufgebaut, über die anschließend WebMUX-Dateneinheiten übertragen werden. Jede WebMUX-Dateneinheit verfügt über einen 8 Byte langen "WebMUX-Header", über den unterschiedliche WebMUX-Verbindungen identifiziert werden können und in dem "Flags" für die Steuerung des Auf- und Abbaus von WebMUX-Verbindungen (SYN, FIN, RST) enthalten sind. Applikationen greifen auf WebMUX-Verbindungen über eine "Socket"-Schnittstelle oder eine vergleichbare Schnittstelle zu. Dabei können beide Kommunikationspartner WebMUX-Verbindungen über dieselbe TCP-Verbindung initiieren. Ähnlich wie bei T/TCP werden Daten bereits zusammen mit SYN-Nachrichten übertragen. Während für die erste WebMUX-Verbindung zunächst ein herkömmlicher TCP-Verbindungsaufbau erforderlich ist, fallen ab der zweiten WebMUX-Verbindung nur noch die Verarbeitungszeiten in den Endsystemen an. Die Verbindungsaufbauzeiten sind somit ab der zweiten WebMUX-Verbindung deutlich geringer.

Dieses Verfahren ermöglicht bei der Anwendung für HTTP die Übertragung mehrerer Zugriffe über dieselbe TCP-Verbindung. Im Gegensatz zu P-HTTP können die Zugriffe jedoch auch parallel erfolgen, so daß einzelne sehr lange Zugriffe keine nachfolgenden Zugriffe blockieren können. Da die einzelnen WebMUX-Verbindungen zudem unabhängig voneinander sind und Anwendungen eine ähnliche Schnittstelle wie bei TCP-Verbindungen zur Verfügung steht, macht WebMUX keine Änderungen des Anwendungsprotokolls erforderlich. Da jede WebMUX-Verbindung wie eine TCP-Verbindung genutzt werden kann, ist auch der Abbruch einzelner Zugriffe möglich, ohne daß wie bei P-HTTP die TCP-Verbindung abgebrochen werden muß. Um dies zu ermöglichen, wird jedoch der "WebMUX-Header" von 8 Byte Länge benötigt, dessen Übertragung einen Teil des zur Verfügung stehenden Datendurchsatzes beansprucht.

Das "Multiplexing" von Verbindungen über eine gemeinsame TCP-Verbindung eignet sich gut für die Verringerung der Verbindungsaufbauzeiten beim Zugriff

eines Clients auf einen Proxy. Selbst für Zugriffe auf unterschiedliche Systeme jenseits des Firewalls braucht ein Client somit keine erneuten TCP-Verbindungen zum Proxy aufzubauen (s. Abb. 5.12). Beim Einsatz von WebMUX reicht es für den Client dann aus, sich einmalig beim Proxy zum Beispiel über SSL zu authentisieren. Anschließend kann die über SSL abgesicherte TCP-Verbindung für beliebige WebMUX-Verbindungen genutzt werden, ohne daß für diese Verbindungen ein erneuter Schlüsselaustausch und eine erneute Authentisierung beim Proxy erforderlich wird. Durch den Einsatz von "Socks" [Leech et al. 96] kann über jede WebMUX-Verbindung ein anderes System jenseits des Firewalls angesprochen werden. Für die Verbindungen vom Proxy zu einem Server jenseits des Firewalls können alternativ herkömmliche TCP-Verbindungen verwendet werden oder es wird auch hier WebMUX eingesetzt, um mehrere Verbindungen über dieselbe TCP-Verbindung übertragen zu können.

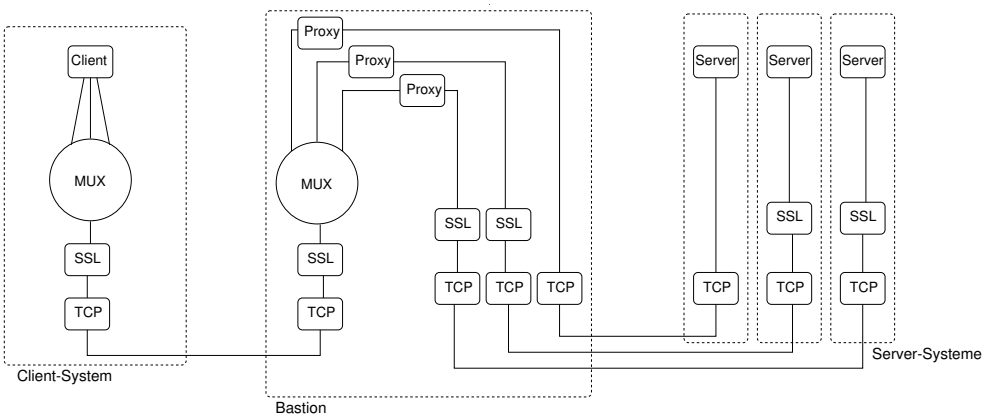


Abbildung 5.12: Gleichzeitiger Zugriff eines Clients auf verschiedene Server über dieselbe TCP-Verbindung zum Proxy

Ein Punkt, der sich insbesondere im Zusammenhang mit der Anwendung des WebMUX-Protokolls bei Firewalls als problematisch erweisen könnte, ist die Zugriffskontrolle in den Endsystemen. Da die nach dem TCP-Verbindungsaufbau durchgeführte SSL-Authentisierung implizit auch für anschließend aufgebauten WebMUX-Verbindungen gilt, können die Rechte des autorisierten Benutzers durch einen Angreifer ausgebeutet werden, wenn es dem Angreifer innerhalb des Endsystems gelingt, ebenfalls eine WebMUX-Verbindung über die bereits authentisierte TCP-Verbindung zu öffnen. Die erreichte Sicherheit hängt daher teilweise von der Zugriffskontrolle in den Endsystemen ab. Dieses Problem ist grundsätzlich nicht neu, da auch herkömmliche TCP-Verbindungen bisher schon von anderen Benutzern desselben Systems mißbraucht werden können und auch dabei die Sicherheit von der Zugriffskontrolle in den Endsystemen abhängt. Ein zusätzliches Risiko besteht bei WebMUX, da nicht nur bestehende Verbindungen übernommen, sondern auch zusätzliche WebMUX-Verbindungen geöffnet werden

können. Das Öffnen einer neuen, zusätzlichen Verbindung fällt dabei deutlich weniger auf als das Übernehmen einer bestehenden Verbindung.

Beim Einsatz von WebMUX im Zusammenhang mit Firewalls ist es prinzipiell auch möglich, einen Proxy zum transparenten Weiterleiten einer TCP-Verbindung zu verwenden und WebMUX zwischen Client und Server einzusetzen. Ein solches Vorgehen hat eine Reihe von Sicherheitsimplikationen. Da insbesondere WebMUX-Verbindungen in beiden Richtungen über die bestehende TCP-Verbindung geöffnet werden können, wird sich der Einsatz von WebMUX zwischen Client und Server in der Regel nicht mit den Sicherheitsrichtlinien, die vom Firewall durchgesetzt werden sollen, vereinbaren lassen.

Fazit: Der Einsatz eines Protokolls zum "Multiplexing" mehrerer Verbindungen über dieselbe TCP-Verbindung eignet sich gut für den Zugriff auf Proxies. Insbesondere bei gleichzeitigem Einsatz von verbindungsorientierten Authentisierungsverfahren wie SSL kann diese Lösung hohe Performanzvorteile erzielen, da eine wiederholte Authentisierung vermieden wird.

5.5 Abschließende Bewertung

In diesem Kapitel wurden Parallelisierungsverfahren für Bastionen (s. Abschnitt 5.2) und Packet Screens (s. Abschnitt 5.3) diskutiert. Außerdem wurden verschiedene Verfahren zur Verringerung der Verbindungsaufbauzeiten bei Proxies (s. Abschnitt 5.4) vorgestellt, da die Verbindungsaufbauzeiten ein weiteres wichtiges Performanzproblem von Proxies sind und hierfür dynamische Methoden der Parallelisierung nicht einsetzbar sind.

Der Vergleich zentraler und dezentraler Verfahren zur Verteilung von Verbindungen auf parallele Bastionen zeigte, daß ein zentraler Verteiler (s. Abschnitt 5.2.2) eine hohe Parallelität und eine gleichmäßige Verteilung des Verkehrs ermöglicht. Allerdings besteht bei zentralen Verfahren grundsätzlich immer die Gefahr, daß der Verteiler selbst zum Engpaß wird. Es ist daher wichtig, eine Verteilung mit möglichst geringem Aufwand zu erreichen oder besser dafür zu sorgen, daß der Verteiler selbst an der eigentlichen Datenübertragung nicht mehr beteiligt ist. Verteilungsverfahren, bei denen der Verteiler an der eigentlichen Datenübertragung nicht mehr beteiligt ist, haben üblicherweise das Problem, daß sich durch die Verteilung Verzögerungen ergeben, da sich eine Zwischenspeicherung von DNS- oder NHRP-Informationen und eine Nutzung des jeweiligen Protokolls zur Verteilung von Verbindungen gegenseitig ausschließen.

Dezentrale Verfahren zur Verteilung von Verbindungen auf parallele Bastionen (s. Abschnitt 5.2.3) ermöglichen gegenüber den zentralen Verteilern eine besse-

re Skalierbarkeit, da auch die Verteilungsfunktion selbst bei diesem Verfahren parallelisiert wird. Die einfachste Möglichkeit für eine dezentrale Verteilung ist die zufällige Auswahl einer Bastion durch den Client (s. Abschnitt 5.2.3.1). Die zufällige Verteilung von Verbindungen eignet sich gut für Anwendungsbereiche, bei denen sehr viele kurze Verbindungen auf parallele Bastionen verteilt werden müssen. Weniger geeignet ist diese Lösung, wenn nur eine geringe Anzahl von langen Verbindungen auf parallele Bastionen verteilt werden muß.

Demgegenüber ist die dezentrale Verteilung durch das Umleiten von Verbindungen (s. Abschnitt 5.2.3.3) zwar gut für die Verteilung weniger langer Verbindungen geeignet, kann aber bei der Verteilung vieler kurzer Verbindungen keine zufriedenstellenden Ergebnisse liefern. Bei jedem Umleiten einer Verbindung verlängern sich die Verbindungsaufbauzeiten. Zudem erfordert auch die Umleitung zuerst die Annahme der Verbindung. Da aber die Verbindungsannahme bei kurzen Verbindungen einen Engpaß bildet, kann durch das Umleiten von Verbindungen der Transaktionsdurchsatz kaum gesteigert werden.

Da Proxies sowohl für kurze als auch für sehr lange Verbindungen verwendet werden, ist für die Verteilung von Verbindungen auf parallele Bastionen am besten ein kombiniertes Verfahren mit zufälliger Auswahl einer Bastion und dem Umleiten von Verbindungen geeignet. Die zufällige Auswahl einer Bastion ermöglicht dabei eine ausreichende Verteilung bei kurzen Verbindungen und wird auch lange Verbindungen in vielen Fällen ausreichend gleichmäßig verteilen. Das Umleiten von Verbindungen ist daher erst in den wenigen Fällen erforderlich, wenn es zu einer sehr ungleichmäßigen Verteilung kommt.

Dieses kombinierte Verteilungsverfahren kann als Erweiterung des heute eingesetzten "Socks"-Protokolls entwickelt werden. Von der hier skizzierten Lösung ist über die Spezifikation des Protokolls, die Entwicklung von Prototypen bis zur breiten Anwendung dieser Lösung noch ein erheblicher Entwicklungsaufwand erforderlich. Die Verfügbarkeit entsprechender Produkte, die den Einsatz dieser Lösung in Unternehmensnetzen erst ermöglichen, läßt sich heute noch nicht abschätzen. Bis zur Einsatzfähigkeit dieser Lösung können Verbindungen noch über den "TCP-Router" (s. Abschnitt 5.2.2.4) auf parallele Bastionen verteilt werden. Die Leistungsfähigkeit des "TCP-Routers" reicht für die Verteilung von Verbindungen in heute gebräuchlichen Hochgeschwindigkeitsnetzen beispielsweise in Fast-Ethernet noch aus. Die unzureichende Skalierbarkeit einer "TCP-Router"-basierten Lösung wird jedoch in Zukunft deutlich skalierbarere Lösungen wie die hier vorgestellte dezentrale Verteilung von Verbindungen erfordern.

Für parallele Packet Screens (s. Abschnitt 5.3) eignet sich das Konzept der Paketparallelität, bei dem Dateneinheiten unabhängig voneinander auf parallele Packet Screens verteilt werden. Da Packet Screens Dateneinheiten auf Netzwerkebene verarbeiten, stellen sie einen viel geringeren Engpaß als Proxies dar. Für Hochgeschwindigkeitsnetze ist die Leistungsfähigkeit von Packet Screens jedoch

nicht ausreichend, so daß auch hier Parallelisierungskonzepte untersucht werden müssen. Auch bei Packet Screens darf die Verteilung selbst nicht zum Engpaß werden. Zentrale Verteiler auf Netzwerkebene, die für die Verteilung von Verbindungen auf parallele Proxies noch ausreichend waren, bilden im Vergleich zu einer Packet Screen einen Engpaß und können daher nicht zur Lastverteilung auf parallele Packet Screens eingesetzt werden. Das vorgestellte Verfahren basiert daher auf einem dezentralen Verteilungsverfahren, bei dem jede Packet Screen selbst die Dateneinheiten zur Verarbeitung auswählt.

Die Verbindungsaufbauzeiten über einen Proxy können im Gegensatz zum Daten- und Transaktionsdurchsatz nicht durch dynamische Methoden zur Parallelisierung reduziert werden. Im Abschnitt 5.4 wurden daher verschiedene Verfahren zur Reduktion der Verbindungsaufbauzeiten über Proxies diskutiert. Da der Anteil der Verbindungsaufbauzeiten an der Gesamttransaktionslänge bei den steigenden Datendurchsätzen in Hochgeschwindigkeitsnetzen an Bedeutung gewinnt, sind Lösungen zur Verringerung der Verbindungsaufbauzeiten eine wichtige Voraussetzung für den Einsatz von Firewalls in Hochgeschwindigkeitsnetzen.

Weitgehend unabhängig von dem jeweils eingesetzten Verteilungsverfahren bestehen mehrere grundsätzliche Probleme bei der Parallelisierung von Firewall-Konzepten. Eine wichtiges Problem ist die variierende Länge der Zugriffe. Da die Migration von Prozessen zwischen den parallelen Firewall-Komponenten (Packet Screens oder Bastionen) ausgeschlossen ist, kann jeder Zugriff durch die Verteilungsfunktion nur einmal einer Firewall-Komponente zugeordnet werden. Bei sehr kurzen Zugriffen, wie zum Beispiel der Verteilung von Datagrammen auf parallele Packet Screens, ist die Last, die durch jeden Zugriff erzeugt wird, etwa gleich, so daß selbst einfache Verteilungsverfahren eine gleichmäßige Verteilung der Last ermöglichen. Bei parallelen Proxies muß jede Verbindung von dem durch die Verteilungsfunktion ausgewählten Proxy verarbeitet werden. Verbindungen können unterschiedlich lange andauern und im Laufe einer Verbindung eine unterschiedlich starke Last erzeugen.[31] Daher können beim Eintreffen einer neuen Verbindung noch keine genauen Aussagen zur Last gemacht werden, die die eintreffende Verbindung zukünftig erzeugen wird. Darüber hinaus können auch für die bereits zugeteilten Verbindungen keine genauen Aussagen über die Last gemacht werden, die zukünftig durch diese Verbindungen auf einer Bastion entstehen wird. Dieses Problem der ungenauen Einschätzung der zu erwartenden Last verschärft sich noch bei Verteilungsverfahren wie z. B. "Round-Robin"-DNS und die Verteilung über NHRP, bei denen nicht jede neue TCP-Verbindung verteilt werden kann. Aufgrund der ungenauen Einschätzung der zukünftigen Last kann die Verteilung immer wieder zu Engpässen auf einer der parallelen Bastionen führen. Ohne die Korrektur einer Überlastsituation durch aufwendige Migration von Prozessen zwischen parallelen Bastionen kann dieses Problem nur verhindert

[31] Der Durchsatz kann auf einer Verbindung kurzzeitig ansteigen.

werden, wenn bereits beim Verbindungsaufbau die für die jeweilige Verbindung erforderlichen Betriebsmittel angegeben werden müssen. Bei den Anwendungen, die heute im Internet durch Firewalls überwacht werden, kommen bisher noch keine Verfahren zur Reservierung von Betriebsmitteln zum Einsatz, so daß ein Einsatz für Verbindungen, die über Firewalls geführt werden, bisher noch an der mangelnden Unterstützung von Reservierungsverfahren im Internet scheitert.

Eine weiteres grundsätzliches Problem sind die Abhängigkeiten, die zwischen verschiedenen Dateneinheiten untereinander bestehen. Neben der Anforderung, daß alle Dateneinheiten, die zu derselben TCP-Verbindung gehören auch von derselben Bastion verarbeitet werden müssen, bestehen auch noch Abhängigkeiten zwischen verschiedenen Verbindungen (FTP-Kontroll- und FTP-Datenverbindungen, "Ident"-Anfragen usw.), die ebenfalls eine Bearbeitung auf derselben Verbindung erfordern. Darüber hinaus treten ICMP-Nachrichten auf, die sich ebenfalls auf bestehende TCP-Verbindungen oder UDP-Dateneinheiten beziehen können. Diese Abhängigkeiten müssen von der Verteilungsfunktion berücksichtigt werden. In vielen Fällen kommt erschwerend hinzu, daß die bestehenden Abhängigkeiten oft erst während der Verarbeitung der Dateneinheit und somit nach erfolgter Verteilung erkennbar werden. Der einfachste Weg um bei der Verteilung sicherzugehen, daß alle Dateneinheiten, zwischen denen Abhängigkeiten bestehen, von derselben Bastion bearbeitet werden, ist die Zuteilung aller Dateneinheiten eines Endsystems zu derselben Bastion. Da verschiedene Verbindungen desselben Endsystems nicht mehr durch unterschiedliche Bastionen kontrolliert werden können, wird jedoch die erreichbare Parallelität eingeschränkt.

Dieses Kapitel gab einen Überblick über parallele Firewall-Konzepte, die wie herkömmliche Firewall-Konzepte weiterhin auf gebräuchlichen Workstations basieren. Eine Steigerung der Performanz von Firewalls ist auch in einigen Fällen (z. B. bei Packet Screens oder Verschlüsselung) durch spezialisierte Hardware möglich. In einigen Teilbereichen — insbesondere bei kryptographischen Verfahren — wird der Einsatz spezialisierter Hardware zukünftig einen wichtigen Beitrag zur Performanzsteigerung leisten. Für eine Zugriffskontrolle auf Anwendungsebene durch Proxies oder Anwendungs-Gateways sind Lösungen, die auf spezialisierter Hardware basieren, jedoch nicht flexibel genug. Parallelisierungsverfahren, die durch den Einsatz herkömmlicher Workstations eine höhere Flexibilität ermöglichen, sind besser zur Performanzsteigerung von Firewalls geeignet als Lösungen, die auf spezialisierter Hardware basieren.

Ein wichtiger Bereich, der in diesem Kapitel nicht diskutiert wurde, ist die Verwaltung von parallelen Firewalls. Die Aufteilung der Aufgaben auf mehrere parallele Firewall-Komponenten erfordert eine konsistente Konfigurierung aller Komponenten, die nur über eine rechnergestützte, zentralisierte Verwaltung der Konfigurationen erreicht werden kann. Da die hier vorgestellten Konzepte für parallele Firewalls jedoch von einer Verteilung auf identische, parallele Firewall-

Komponenten ausgehen, muß lediglich die identische Konfiguration aller Komponenten sichergestellt werden. Nur wenn die einzelnen parallelen Komponenten unterschiedliche Aufgaben haben, sind auch unterschiedliche Konfigurationen erforderlich, deren konsistente Konfiguration erheblich aufwendiger ist.

6 Zusammenfassung und Ausblick

Inhalt

6.1 Zusammenfassung der Ergebnisse 423

6.1.1 Firewalls in ATM-Netzen 423

6.1.2 Performanz von Firewalls 428

6.2 Ausblick 431

6.1 Zusammenfassung der Ergebnisse

Diese Dissertation untersucht den Einsatz von Firewalls in Hochgeschwindigkeitsnetzen. Hochgeschwindigkeitsnetze stellen für Firewalls in zweifacher Hinsicht ein Problem dar. Einerseits sind Firewalls bewußt als Engpaß konzipiert, um an diesem Engpaß die Kommunikation zwischen zwei Netzen überwachen zu können. Andererseits wirken Firewalls in Hochgeschwindigkeitsnetzen aber auch als Performanzengpaß, da die Leistungsfähigkeit bestehender Firewall-Konzepte für Hochgeschwindigkeitsnetze nicht ausreicht. Zusätzlich zu diesen Performanzproblemen entstehen im Falle von ATM durch Hochgeschwindigkeitsnetze auch technische Herausforderungen für Firewalls. Die Technik von ATM-Netzen unterscheidet sich erheblich von der Technik herkömmlicher Netze, so daß vorhandene Firewall-Konzepte nicht direkt für ATM-Netze übernommen werden können. In dieser Arbeit wurden sowohl die technischen Aspekte als auch die Performanzaspekte des Einsatzes von Firewalls in Hochgeschwindigkeitsnetzen untersucht. Kapitel 3 untersuchte die technischen Probleme und die neuen Möglichkeiten, die sich beim Einsatz von Firewalls in ATM-Netzen ergeben. In Kapitel 4 wurde die Performanz von Firewalls in Hochgeschwindigkeitsnetzen analysiert.

6.1.1 Firewalls in ATM-Netzen

Die Integration eines Firewalls in ein Netz erfordert zunächst eine Analyse der Sicherheitsrisiken, gegen die der Firewall einen Schutz bieten soll bzw. die für den

Firewall selbst eine Bedrohung darstellen. Die Untersuchungen im Abschnitt 3.2 haben gezeigt, daß ATM-Netze nicht sicherer sind als herkömmliche Netze. Angriffe durch das Abhören übertragener Daten, "Spoofing"-Angriffe und Angriffe auf die Verfügbarkeit sind auch in ATM-Netzen möglich. Zwar ist es bei dem verbindungsorientierten Konzept von ATM nicht mehr möglich einzelne Dateneinheiten (Zellen) zu fälschen, da deren Adreßinformationen nur eine lokale Bedeutung im Kontext einer virtuellen Verbindung haben. Da aber beim Aufbau einer virtuellen Verbindung keine Authentisierung der Kommunikationspartner erfolgt, kann ein "Spoofing"-Angriff bereits hier ansetzen. Darüber hinaus enthalten CLIP, NHRP, LANE und MPOA, die zur Übertragung von Dateneinheiten herkömmlicher Protokolle in ATM-Netzen eingesetzt werden, keine wirksamen Sicherheitsmechanismen, so daß vielfältige Angriffsmöglichkeiten bestehen. Da ATM-Netze vorerst in erster Linie zur Übertragung von IP-Dateneinheiten verwendet werden, bestehen natürlich weiterhin alle IP-basierten Angriffsmöglichkeiten, die bereits heute in herkömmlichen Netzen Sicherheitsmechanismen wie Firewalls erforderlich machen.

Durch die **Kontrolle der Signalisierung** beim Aufbau virtueller Verbindungen ergeben sich in ATM-Netzen neuartige Möglichkeiten für Firewall-Komponenten, die in Abschnitt 3.3.1 untersucht wurden. Einfache **Signalisierungsfilter**, die auf der Basis der Adressen der Kommunikationspartner ("Calling Party Number" und "Called Party Number") einen Verbindungsaufbau zulassen oder unterbinden, sind bereits heute in ATM-Switches verfügbar. Bei der Analyse der Signalisierungsnachrichten wurden weitere zusätzliche Möglichkeiten für eine Zugriffskontrolle identifiziert, die zukünftig für eine wesentlich detailliertere Zugriffskontrolle genutzt werden können.

Über die Zugriffskontrolle von Signalisierungsfiltern hinausgehend erlaubt das neue Konzept der **Sicherheitsdienstgüte-basierten Wegwahl** das Einbinden weiterer Sicherheitskomponenten in eine virtuelle Verbindung. Der **Signalisierungskontroller** bestimmt bei der Signalisierung anhand von Sicherheitsrichtlinien die erforderliche Sicherheitsdienstgüte für die aufzubauende virtuelle Verbindung. Durch die Veränderung der Signalisierungsnachricht kann der Signalisierungskontroller für die Kommunikationspartner transparent zusätzliche Sicherheitskomponenten in die virtuelle Verbindung einbinden, die anschließend als Proxy oder Packet Screen für eine Überwachung der übertragenen Daten sorgen. Es wurden verschiedene Ansätze untersucht, die das Einbinden von Sicherheitskomponenten innerhalb der bestehenden PNNI-Spezifikation implementieren. Darüber hinaus wurde das Einbinden von Sicherheitskomponenten durch eine NHRP-basierte Lösung untersucht, die sich jedoch nur für die Kontrolle von IP-Kommunikation eignet und nicht wie ein Signalisierungskontroller auch für "Native"-ATM-Anwendungen genutzt werden kann.

Für die in herkömmlichen Netzen häufig genutzten Packet Screens stellt die Technik von ATM-Netzen ein wesentliches Problem dar, weil ein Datagramm in einem ATM-Netz auf mehrere Zellen aufgeteilt wird und nicht wie in herkömmlichen Netzen in einem einzigen Rahmen übertragen wird. Durch das Zusammenfügen ("Reassembly") der Zellen zu einem Datagramm können Packet Screens zwar auch in ATM-Netzen eingesetzt werden, eine höherer Effizienz wird jedoch erst mit "**Cell Screens**" erreicht, die Datagramme auch ohne vorherige Reassemblierung kontrollieren können. Die Möglichkeiten und Grenzen von "Cell Screens" wurden im Abschnitt 3.3.2 untersucht.

Auch in ATM-Netzen ermöglichen Firewall-Komponenten auf Anwendungsebene die detaillierteste Kontrolle der übertragenen Daten. Für die bei herkömmlichen Firewalls eingesetzten IP-basierten Proxies und Anwendungs-Gateways stellt der Einsatz von ATM kein unmittelbares Problem dar, weil die Firewall-Komponenten als Prozesse auf Anwendungsebene keinen direkten Kontakt zum ATM-Netz haben und die Dienste zur Übertragung von IP-Dateneinheiten (CLIP, LANE, NHRP und MPOA) eine transparente Nutzung des ATM-Netzes ermöglichen.

Aufgrund der geringen Verbreitung von "Native"-ATM-Anwendungen sind bisher noch keine **"Native"-ATM-Proxies** verfügbar. Der Einsatz von Proxies für "Native"-ATM-Anwendungen wurde im Abschnitt 3.3.3.2 untersucht. Ein wesentlicher Unterschied zu herkömmlichen Proxies entsteht durch die Möglichkeit von "Native"-ATM-Anwendungen, eine bestimmte Dienstgüte für eine virtuelle Verbindung anzufordern. Mechanismen, durch die eine angeforderte Dienstgüte garantiert werden kann, stehen bei heutigen Proxies nicht zur Verfügung. Untersuchungen der "Native"-ATM-Anwendungen Telefonie und "Video on Demand" haben jedoch gezeigt, daß es heute bei diesen "Native"-ATM-Anwendungen noch keinen Bedarf für die Kontrolle von Video- und Audio-Verbindungen gibt, für die besondere Dienstgüteanforderungen bestehen. Für die Kontrollverbindungen, bei denen eine Überwachung durch Proxies erforderlich ist, bestehen bei diesen Anwendungen jedoch keine besonderen Dienstgüteanforderungen, so daß vorerst herkömmliche Proxy-Konzepte auch für die Kontrolle von "Native"-ATM-Anwendungen eingesetzt werden können.

Damit Proxies von einem Signalisierungskontroller ohne Benachrichtigung der Kommunikationspartner zur Überwachung einer virtuellen Verbindung eingesetzt werden können, müssen sich die Proxies gegenüber dem Client transparent verhalten. Bei "Native"-ATM-Proxies ist die Transparenz vergleichsweise einfach zu erreichen, da ausschließlich bei der Signalisierung zum Aufbau einer virtuellen Verbindung Adreßinformationen verwendet werden. Bei herkömmlichen IP-basierten Proxies war das transparente Einbinden bisher nicht möglich, da jede über den Proxy übertragene IP-Dateneinheit eigene Adreßinformationen enthält. Durch die Kombination eines Proxies mit einer NAT-Komponente ermöglicht das

neue in Abschnitt 3.3.3.3 vorgestellte Konzept des **transparenten Proxies für NHRP-Netze** auch bei IP-basiertem Verkehr eine transparente Kontrolle durch den Proxy. Mit einem Prototyp wurde die Realisierbarkeit dieses Konzeptes demonstriert.

Für die Kontrolle von IP-Verkehr in einem ATM-Netz können die bekannten für IP-Netze entwickelten Firewall-Architekturen — Packet Screens, Gateway-Firewalls sowie Kombinationen von Packet Screen und Bastion — auch in CLIP- oder LANE-Netzen nachgebildet werden. Bei dieser direkten Migration der herkömmlichen Firewall-Architekturen sind allenfalls geringe Änderungen an den eingesetzten Firewall-Produkten erforderlich. Die aus diesem Ansatz resultierenden Firewalls sind jedoch nur unzureichend an ATM-Netze angepaßt und nutzen daher wesentliche Eigenschaften von ATM-Netzen nicht aus.

Durch den Einsatz von Signalisierungsfiltern können ATM-Switches zum **Aufbau virtueller Netze** verwendet werden. Der Aufbau herkömmlicher Firewall-Architekturen vereinfacht sich dadurch erheblich. Vom Firewall getrennte Netze auf unterschiedlichem Sicherheitsniveau können durch den Einsatz eines Signalisierungsfilters auf demselben Switch als virtuelle Netze konfiguriert werden. Die Konfiguration des Signalisierungsfilters bleibt hierbei relativ einfach, da mit den Systemen im Firewall-Umfeld nur wenige Systeme über die Konfiguration von Filterregeln einem virtuellen Netz zugeordnet werden müssen.

Beim Einsatz von Firewalls in herkömmlichen Netzen hat sich bisher gezeigt, daß eine Zugriffskontrolle auf IP-Ebene allein meist nicht ausreichend ist und praktisch immer Proxies als Zugriffskontrollmechanismen auf Anwendungsebene benötigt werden. Packet Screens werden in herkömmlichen Netzen fast ausschließlich in Kombination mit Bastionen verwendet. Die Aufgabe der Packet Screen ist es dabei sicherzustellen, daß ausschließlich die Bastion mit beiden Netzen — dem internen und dem externen Netz — kommunizieren kann. In ATM-basierten Netzen kann die Aufgabe der Packet Screen durch einen Signalisierungsfilter übernommen werden. Da für jede Kommunikation in ATM-Netzen zuerst der Aufbau einer virtuellen Verbindung erforderlich ist, kann ein Signalisierungsfilter durch eine Zugriffskontrolle beim Aufbau der virtuellen Verbindung sicherstellen, daß keine direkten virtuellen Verbindungen zwischen internem und externem Netz aufgebaut werden und daß ausschließlich die Bastion mit Systemen beider Netze kommunizieren kann. Diese **Kombination von Signalisierungsfilter und Bastion** ist eine neue ATM-spezifische Firewall-Architektur, die sowohl zur Kontrolle von IP- als auch von "Native"-ATM-Diensten angewendet werden kann. Durch den Einsatz eines Signalisierungskontrollers statt eines Signalisierungsfilters besteht zusätzlich die Möglichkeit, die Bastion transparent in die virtuelle Verbindung zu integrieren. Der Signalisierungskontroller kann zudem verschiedene virtuelle Verbindungen unterschiedlichen Bastionen zuordnen und dadurch für eine Lastverteilung zwischen parallelen Bastionen sorgen.

Die Ausdehnung des **Konzeptes der virtuellen Netze** auf das gesamte interne Netz ermöglicht es, das interne Netz nach Sicherheitsgesichtspunkten zu strukturieren. Dieses neue Konzept erlaubt es, auch räumlich benachbarte Systeme unterschiedlichen virtuellen Netzen zuzuordnen. Hierfür ist es erforderlich, eine Zugriffskontrolle auf Signalisierungsnachrichten bei allen Switches anzuwenden. Da auch bei herkömmlichen Netzen zunehmend Switches eingesetzt werden, wären virtuelle Netze prinzipiell auch bei herkömmlichen Netzen denkbar. Da die Switches in herkömmlichen Netzen allerdings alle Dateneinheiten filtern müssen und nicht wie bei ATM-Netzen ausschließlich beim Verbindungsaufbau einzugreifen brauchen, ist der Einsatz virtueller Netze bei herkömmlichen Netzen kaum praktikabel.

In Abschnitt 3.4.5 wurden drei unterschiedliche Lösungen für das Verbinden virtueller Netze durch Firewalls vorgestellt. Dezentrale Firewalls werden unabhängig voneinander an jedem Übergang zwischen zwei Netzen eingesetzt. Abhängig von der Topologie kann es daher erforderlich sein, mehrere Firewalls auf dem Weg zum gewünschten Ziel zu überqueren. Bei einem zentralen Firewall gibt es hingegen nur einen einzigen Übergang zwischen allen virtuellen Netzen. Um mit einem System in einem anderen virtuellen Netz zu kommunizieren, reicht es hierbei aus, lediglich den zentralen Firewall zu überqueren. Ein zentraler Firewall stellt jedoch auch eine zentrale Fehlerquelle dar und kann erhebliche Umwege zur Folge haben, wenn räumlich benachbarte Systeme unterschiedlicher virtueller Netze miteinander kommunizieren. Die flexibelste Lösung ist der "**virtuell zentrale Firewall**", bei dem dezentral angeordnete Firewall-Komponenten anhand einer einheitlichen, zentral vorgegebenen Sicherheitsrichtlinie die Kommunikation zwischen unterschiedlichen virtuellen Netzen überwachen. Die für die Zugriffskontrolle erforderlichen dezentralen Firewall-Komponenten werden bei diesem Konzept von einem Signalisierungskontroller beim Verbindungsaufbau in eine virtuelle Verbindung eingebunden. Die Auswahl der Firewall-Komponente trifft der Signalisierungskontroller anhand der Sicherheitsrichtlinien und der derzeitigen Last der Firewall-Komponenten. Das Firewall-Konzept des "virtuell zentralen Firewalls" ermöglicht nicht nur wie herkömmliche Firewall-Konzepte eine Zugriffskontrolle bei Zugriffen von außen, sondern auch bei Zugriffen zwischen internen Systemen unterschiedlicher virtueller Netze. Durch die Auswahl dezentraler Firewall-Komponenten wird eine hohe Ausfallsicherheit und eine gute Skalierbarkeit des "virtuell zentralen Firewalls" erreicht.

Die sich von herkömmlichen Netzen stark unterscheidende ATM-Technologie hat vielfältige Auswirkungen auf Firewalls. Die bei herkömmlichen Firewalls häufig eingesetzten Packet Screens eignen sich nur unzureichend für einen Einsatz in ATM-Netzen. Durch die neuen Konzepte des Signalisierungskontrollers, der transparenten Proxies, der virtuellen Netze und des "virtuell zentralen Firewalls" entstehen jedoch beim Einsatz von ATM-Netzen vielfältige leistungsfähige Mög-

lichkeiten für eine Zugriffskontrolle, so daß die neuen Möglichkeiten gegenüber den technischen Problemen überwiegen.

6.1.2 Performanz von Firewalls

Herkömmliche Firewalls werden in Hochgeschwindigkeitsnetzen zum Performanzengpaß. Über die Performanzauswirkungen von Firewalls gab es bisher keine detaillierten Untersuchungen. Durch Messungen an Firewalls in einem ATM-Netz wurden im Kapitel 4 die prinzipbedingten Engpässe von Firewalls in Hochgeschwindigkeitsnetzen analysiert. Die Ergebnisse der Messungen an herkömmlichen IP-basierten Firewall-Komponenten können auch auf andere Hochgeschwindigkeitsnetze übertragen werden, da bei dem Einsatz der Firewall-Komponenten in einem CLIP-Netz keine ATM-spezifischen Eigenschaften ausgenutzt werden.

Unter dem Oberbegriff "Performanz" verbergen sich eine Reihe von Kriterien: Datendurchsatz, Datagrammdurchsatz, Nachrichtenlaufzeit, Verbindungsaufbauzeit, maximale Anzahl paralleler Verbindungen und Transaktionsdurchsatz. Für einige Performanzkriterien standen bereits Meßwerkzeuge zur Verfügung, die für Performanzmessungen an Firewalls lediglich angepaßt werden mußten. Für die bei Proxies besonders wichtigen Messungen mit parallelen Verbindungen mußte jedoch zunächst ein geeignetes Meßwerkzeug entwickelt werden. Das Problem der Messungen mit parallelen Verbindungen ist es sicherzustellen, daß während der gesamten Messung auch die geforderte Anzahl paralleler Verbindungen aktiv ist. Das entwickelte **Meßwerkzeug für Messungen mit parallelen HTTP-Verbindungen** verfügt zu diesem Zweck über Synchronisationsmechanismen, die mit Hilfe von Anlauf- und Auslaufphasen sicherstellen, daß die geforderte Anzahl paralleler Verbindungen während der gesamten Meßdauer aktiv ist.

Mit einem **Datendurchsatz** um 134 MBit/s können sowohl Packet Screens als auch Proxies die in einem 155 MBit/s ATM-Netz über CLIP maximal erreichbaren Datendurchsätze erzielen. Dieser Datendurchsatz konnte zwar bei einem Proxy erst mit einer Performanz-optimierten Implementation erreicht werden, aber dieses Ergebnis zeigt, daß selbst die Kopierleistung auf Anwendungsebene prinzipiell für die in einem 155 MBit/s ATM-Netz möglichen Datendurchsätze ausreicht. Im Vergleich mit der Kurve des theoretisch erreichbaren Datendurchsatzes zeigen Proxy und Packet Screen bei kürzeren Nachrichtenlängen starke Einbrüche. Die Ursache für diese Einbrüche liegt im beschränkten Datagrammdurchsatz beider Firewall-Komponenten.

Der **Datagrammdurchsatz** stellt den wesentlichen Engpaß von Packet Screens dar, weil für jedes übertragene Datagramm Teile der Filterregeln oder im ungünstigsten Fall sogar alle konfigurierten Filterregeln ausgewertet werden müssen. Dieser Aufwand fällt unabhängig von der Länge des Datagramms an. Bei großen

Dateneinheiten reicht der Datagrammdurchsatz der Packet Screen aus, um den zur Verfügung stehenden Datendurchsatz auszunutzen. Bei sehr kurzen Dateneinheiten kann die untersuchte Packet Screen jedoch mit einem maximalen Datagrammdurchsatz im Bereich von 8.000 Datagrammen/s bei weitem nicht den im Netz zur Verfügung stehenden Durchsatz erzielen. Da die durchschnittliche Datagrammlänge im Internet mit weniger als 500 Byte bereits in einem Bereich liegt, in dem der Datagrammdurchsatz den Engpaß von Packet Screens bildet, wirkt die bei den Messungen verwendete Workstation-basierte Packet Screen bereits in einem 155 MBit/s ATM-Netz als Engpaß.

Durch die Kontrolle der Firewall-Komponenten steigt die **Nachrichtenlaufzeit**. In LANs, in denen die Nachrichtenlaufzeit ohnehin primär von der Verarbeitung in den Systemen bestimmt wird, führt die zusätzliche Kontrolle des Firewalls zu einer erheblichen Steigerung der Nachrichtenlaufzeit.

Auf die **Verbindungsaufbauzeit** haben Packet Screens als Firewall-Komponenten auf IP-Ebene keinen unmittelbaren Einfluß, da sie nicht direkt am Verbindungsaufbau beteiligt sind. Bei Verbindungen über einen Proxy ist die Verlängerung der Verbindungsaufbauzeit aus der Sicht des Benutzers hingegen die wichtigste Performanzauswirkung des Proxies. Die Verlängerung der Verbindungsaufbauzeit entsteht, weil die Kommunikation mit dem gewünschten Empfänger erst nach dem Aufbau von zwei Verbindungen erfolgen kann. Nachdem der Client eine Verbindung zum Proxy aufgebaut hat, ist zunächst eine Authentisierung beim Proxy erforderlich. Erst anschließend erfolgt der Aufbau der zweiten Verbindung vom Proxy zum Server. In einem LAN, bei dem aufgrund der geringen Entfernungen die Nachrichtenlaufzeiten im Netz eine untergeordnete Rolle spielen, entsteht im Vergleich zu einer direkten Host-zu-Host-Verbindung mindestens eine Verdoppelung der Verbindungsaufbauzeiten. In Hochgeschwindigkeitsnetzen macht sich die Erhöhung der Verbindungsaufbauzeit besonders stark bemerkbar, da mit steigenden Datendurchsätzen die für die reine Übertragung einer Nachricht benötigte Zeit abnimmt.

Während aus der Sicht eines Benutzers die Steigerung der Verbindungsaufbauzeit die wichtigste Performanzauswirkung eines Proxies ist, muß ein Proxy dadurch nicht zwangsläufig zum Engpaß werden, wenn die Wartezeit von anderen parallel arbeitenden Benutzern ausgenutzt werden kann. Messungen mit MGproxy zeigen, daß maximal 250 parallele Verbindungen vom Proxy bedient werden können. Die **maximale Anzahl paralleler Verbindungen**, bei der in der Summe der Einzeldurchsätze aller Verbindungen noch der zur Verfügung stehende Datendurchsatz ausgenutzt wird, liegt jedoch sehr viel niedriger bei 30 parallelen Verbindungen.

Bei kürzeren Transaktionen, wie z. B. HTTP-Zugriffen, erweist sich der **Transaktionsdurchsatz** von Proxies als Engpaß. Dieser Engpaß wird durch die Ver-

bindungsannahme beim Proxy verursacht und führt dazu, daß erst bei Transaktionslängen oberhalb von 500 KByte der Netzdurchsatz ausgenutzt werden kann.

Beim **Einsatz von Zugriffskontrolle oder von Verschlüsselung** wird die CPU der Bastion bereits von einer einzelnen Verbindung voll ausgelastet. Die gemessenen Datendurchsätze liegen weit unterhalb des in Hochgeschwindigkeitsnetzen zur Verfügung stehenden Datendurchsatzes.

Durch **kryptographische Authentisierungsmechanismen und Schlüsselaustauschverfahren** sinkt der Transaktionsdurchsatz von Proxies erheblich während gleichzeitig die Verbindungsaufbauzeiten steigen. Die Reduktion des Transaktionsdurchsatzes ist besonders schwerwiegend, da der Transaktionsdurchsatz von Proxies bereits ohne die kryptographische Authentisierung einen Engpaß darstellt.

Die in Kapitel 5 untersuchten Konzepte zur Entschärfung der identifizierten Engpässe konzentrierten sich auf Proxies, da die Engpässe bei Proxies durch die Verarbeitung auf Anwendungsebene wesentlich einschneidender sind als die Performanzprobleme bei Packet Screens. Zur Steigerung des Daten- und des Transaktionsdurchsatzes von Proxies wurden dynamische Methoden zur Verteilung von Verbindungen auf **parallele Bastionen** untersucht.

Der **Vergleich zentraler und dezentraler Verfahren** zur Verteilung von Verbindungen auf parallele Bastionen zeigte, daß zentrale Verteiler eine hohe Parallelität und eine gleichmäßige Verteilung ermöglichen, aber auch leicht selbst zum Engpaß werden können. Da bereits für verwandte Anwendungsbereiche Produkte existieren (z. B. für die Verteilung von Zugriffen auf parallele Web-Server), können Verfahren, die auf zentralen Verteilern basieren, relativ kurzfristig zur Verteilung von Verbindungen auf parallele Proxies eingesetzt werden.

Dezentrale Verteilungsverfahren ermöglichen eine bessere Skalierbarkeit als zentrale Verteilungsverfahren. Abhängig von der Art des zu verteilenden Verkehrs sind unterschiedliche Verfahren besser geeignet. Die zufällige Verteilung von Verbindungen eignet sich besonders für kurze Transaktionen. Bei langen Transaktionen kann eine zufällige Verteilung jedoch zu einer sehr ungleichmäßigen Verteilung führen. Für lange Transaktionen ist stattdessen das Umleiten von Verbindungen besser geeignet. Da Proxies sowohl bei langen als auch bei kurzen Transaktionen eingesetzt werden, ist ein kombiniertes Verteilungsverfahren mit zufälliger Verteilung und Umleiten von Verbindungen erfolgversprechend. Eine derartige Lösung ist als Weiterentwicklung auf der Basis des "Socks"-Protokolls denkbar.

Eine **Reduktion der Verbindungsaufbauzeiten** über einen Proxy kann durch Verfahren zur Verteilung von Verbindungen auf parallele Bastionen nicht erreicht werden. Die Auswirkungen dieses Engpasses von Proxies können durch zwei

unterschiedliche Strategien reduziert werden. Mit Hilfe eines **vereinfachten "Handshakes"** kann der Aufbau einer TCP-Verbindung beschleunigt werden. Der beschleunigte Aufbau der TCP-Verbindung reicht jedoch nicht aus, wenn eine aufwendige kryptographische Authentisierung für die langen Verbindungsaufbauzeiten verantwortlich ist. Wichtig sind daher ebenfalls **Verfahren zur Vermeidung von wiederholtem Verbindungsauf- und -abbau**.

6.2 Ausblick

Die Untersuchungen in dieser Arbeit haben gezeigt, daß der Einsatz von Firewalls und ATM sich nicht gegenseitig widersprechen. Die ATM-Technologie bietet stattdessen eine Reihe von Eigenschaften, die neue leistungsfähige Firewall-Konzepte ermöglichen. Bis zur Produktreife der in dieser Arbeit vorgestellten neuen Firewall-Komponenten — dem Signalisierungskontroller, den "Native"-ATM-Proxies und den transparenten Proxies für NHRP-Netze — ist jedoch noch erheblicher Entwicklungsaufwand erforderlich.

Auch in ATM-Netzen wird IP weiterhin einen wesentlichen Anteil am Verkehrsaufkommen haben. "Native"-ATM-Anwendungen werden voraussichtlich nur in speziellen Anwendungsbereichen wie zum Beispiel bei Video-Konferenzsystemen und bei der Telefonie eingesetzt werden. Firewalls werden daher auch zukünftig in erster Linie für die Kontrolle von IP-Verkehr benötigt.

Relativ schnell wird sich auch in der Praxis zeigen, daß die Performanz von herkömmlichen Firewalls für Hochgeschwindigkeitsnetze nicht ausreicht. Die identifizierten Performanzengpässe bei Proxies werden sich durch den immer wichtiger werdenden Einsatz von "Scannern" zur Überwachung von ausführbaren Inhalten noch verschärfen.

Zur Lösung dieser Performanzprobleme bei Proxies sind zentrale Verteilungsverfahren bereits kurzfristig einsetzbar. Jedoch ermöglicht die mangelnde Skalierbarkeit dieser Verfahren lediglich eine kurzzeitige Lösung des Problems, so daß zentrale Verteiler bereits in absehbarer Zeit ebenfalls zu einem Engpaß werden.

Skalierbare Lösungen, die auch bei der steigenden Leistungsfähigkeit von Hochgeschwindigkeitsnetzen nicht zum Engpaß werden, müssen auf Basis von dezentralen Verteilungsverfahren entwickelt werden. Bei den in dieser Arbeit skizzierten dezentralen Verteilungsverfahren sind noch weitere Untersuchungen erforderlich, bevor deren Einsatz bei parallelen Packet Screens und parallelen Proxies auch eine Lösung für den Einsatz von Firewalls in zukünftigen noch leistungsfähigeren Hochgeschwindigkeitsnetzen bringt.

Anhang

A. Literaturverzeichnis

[Alles 95] Anthony Alles. "ATM Internetworking". Cisco Systems, May 1995. (`http://www.cisco.com/warp/public/614/12.html`)

[Andrikopoulos et al. 96] Ilias Andrikopoulos, T. Ors, M. Matijasevic, H. Leitold, S. P. W. Jones, R. Posch. "TCP/IP Throughput Performance Evaluation for ATM Local Area Networks". *Proceedings of 4th IFIP Workshop on Performance Modelling and Evaluation of ATM Networks, Ilkley, UK, July 1996*, 1996. (`http://www.ee.surrey.ac.uk/Personal/I.Andrikopoulos/Papers/ifip96.ps.gz`)

[Andrikopoulos et al. 97] Ilias Andrikopoulos, T. Ors, M. Matijasevic, H. Leitold, R. Posch. "A Study of TCP/IP Throughput Performance in Heterogeneous ATM LANs". *In: "ATM Networks: Performance Modelling and Analysis", Volume III, First Edition, Ed. D.D Kouvatsos, Published by Chapman & Hall*, 1997. (`http://www.ee.surrey.ac.uk/Personal/I.Andrikopoulos/Papers/IFIPatm.ps.gz`)

[Armitage 96] Grenville Armitage. "Support for Multicast over UNI 3.0/3.1 based ATM Networks". *RFC 2022*, November 1996. (`ftp://ftp.isi.edu/in-notes/rfc2022.txt`)

[Armitage 97] Grenville Armitage. "Security issues for ION protocols". *Internet Draft*, 28. Juli 1997.

[Armitage et al. 97] Grenville Armitage, Cliff X. Wang. "Security issues for the ATMARP protocol". *Internet Draft*, 11. Oktober 1997.

[Armitage et al. 98] Grenville Armitage, Peter Schulter, Markus Jork. "IPv6 over ATM Networks". *Internet Draft*, 17. Oktober 1998.

[Atkins et al. 96] Derek Atkins, William Stallings, Philip Zimmermann. "PGP Message Exchange Formats". *RFC 1991*, August 1996. (`ftp://ftp.isi.edu/in-notes/rfc1991.txt`)

[Atkinson 94] Randall Atkinson. "Default IP MTU for use over ATM AAL5". *RFC 1626*, Mai 1994. (`ftp://ftp.isi.edu/in-notes/rfc1626.txt`)

[Atkinson 95a] Randall Atkinson. "Security Architecture for the Internet Protocol". *RFC 1825*, August 1995. (`ftp://ftp.isi.edu/in-notes/rfc1825.txt`)

[Atkinson 95b] Randall Atkinson. "IP Authentication Header". *RFC 1826*, August 1995. (`ftp://ftp.isi.edu/in-notes/rfc1826.txt`)

[Atkinson 95c] Randall Atkinson. "IP Encapsulating Security Payload (ESP)". *RFC 1827*, August 1995. (`ftp://ftp.isi.edu/in-notes/rfc1827.txt`)

[ATM Forum 94] ATM Forum Technical Committee. "User-Network Interface (UNI) Specification Version 3.1". *ATM Forum* (AF-UNI-0010.002), September 1994.

[ATM Forum 95a] ATM Forum Technical Committee. "LAN Emulation Over ATM Version 1.0". *ATM Forum* (AF-LANE-0021.000), Januar 1995.

[ATM Forum 95b] ATM Forum Technical Committee. "155.52 Mbps Physical Layer Specification for Category-3 Unshielded Twisted Pair". *ATM Forum* (AF-PHY-0047.000), November 1995.

[ATM Forum 96a] ATM Forum Technical Committee. "622.08 Mbps Physical Layer Specification". *ATM Forum* (AF-PHY-0046.000), Januar 1996.

[ATM Forum 96b] ATM Forum Technical Committee. "Native ATM Services: Semantic Description Version 1.0". *ATM Forum* (AF-SAA-0048.000), Februar 1996.

[ATM Forum 96c] ATM Forum Technical Committee. "Private Network-Network Interface Specification Version 1.0 (PNNI 1.0)". *ATM Forum* (AF-PNNI-0055.000), März 1996.

[ATM Forum 96d] ATM Forum Technical Committee. "ATM User-Network Interface (UNI) Signalling Specification Version 4.0". *ATM Forum* (AF-SIG-0061.000), Juli 1996.

[ATM Forum 96e] ATM Forum Technical Committee. "Integrated Local Management Interface (ILMI) Specification Version 4.0". *ATM Forum* (AF-ILMI-0065.000), September 1996.

[ATM Forum 97a] ATM Forum Technical Committee. "Audiovisual Multimedia Services: Video on Demand Specification 1.1". *ATM Forum* (AF-SAA-0049.001), März 1997.

[ATM Forum 97b] ATM Forum Technical Committee. "Voice and Telephony Over ATM to the Desktop Specification". *ATM Forum* (AF-VTOA-0083.000), Mai 1997.

[ATM Forum 97c] John D. Keene, Editor. "LAN Emulation Over ATM Version 2 – LUNI Specification". *ATM Forum* (AF-LANE-0084.000), Juli 1997.

[ATM Forum 97d] Andre N. Fredette, Editor. "Multi-Protocol Over ATM Version 1.0". *ATM Forum* (AF-MPOA-0087.000), Juli 1997.

[Badach et al. 94] Anatol Badach, Erwin Hoffmann, Olaf Knauer. "High Speed Internetworking: Grundlagen und Konzepte für den Einsatz von FDDI und ATM". Addison-Wesley, 1994.

[Balabanian et al. 96] Vahe Balabanian, Liam Casey, Nancy Greene, Chris Adams. "An Introduction to Digital Storage Media - Command and Control (DSM-CC)". *IEEE Communication*, November 1996.
(`http://drogo.cselt.stet.it/mpeg/documents/dsmcc.htm`)

[Bellovin 89] Steven M. Bellovin. "Security Problems in the TCP/IP Protocol Suite". Computer Communication Review, Vol. 19, No. 2, pp. 32-48, April 1989.
(`ftp://ftp.cert.dfn.de/pub/docs/net/ipext.ps.gz`)

[Bellovin et al. 91] Steven M. Bellovin, Michael Merrit. "Limitations of the Kerberos Authentication System". *Proceedings of the Winter 1991 USENIX Conference, Dallas, TX*, 1991.
(`ftp://ftp.cert.dfn.de/pub/docs/kerberos/kerblimit.usenix.ps.gz`)

[Bellovin 92] Steven M. Bellovin, "There be Dragons". *Proceedings: 5th International Computer Virus & Security Conference 92"*, 1992.
(`ftp://ftp.cert.dfn.de/pub/firewalls/att/dragon.ps.gz`)

[Bellovin 96] Steven M. Bellovin. "Defending Against Sequence Number Attacks". *RFC 1948*, Mai 1996. (`ftp://ftp.isi.edu/in-notes/rfc1948.txt`)

[Benecke et al. 97a] Carsten Benecke, Uwe Ellermann. "Firewall-Test-Labor für Hochgeschwindigkeitsnetze: Quartalsbericht für den Zeitraum vom 1. April 1997 bis 30. Juni 1997". *DFN-FWL*, Juli 1997.

[Benecke et al. 97b] Carsten Benecke, Uwe Ellermann. "Firewall-Test-Labor für Hochgeschwindigkeitsnetze: Quartalsbericht für den Zeitraum vom 1. Juli 1997 bis 30. September 1997". *DFN-FWL*, Oktober 1997.

[Benecke et al. 98a] Carsten Benecke, Uwe Ellermann. "Securing 'Classical IP over ATM Networks' ". *Proceedings of 7th USENIX Security Symposium, 26.-29. Januar 1998, San Antonio, Texas*, Januar 1998.
(`http://www.fwl.dfn.de/eng/team/cb/eng_natm/`)

[Benecke et al. 98b] Carsten Benecke, Uwe Ellermann. "Tools für Durchsatzmessungen an Proxies: Entwurf und Realisierung am Beispiel von HTTP-Client und -Server". *MMB-Arbeitsgespräche, 5.-6. März 1998, Hamburg*, März 1998.
(`http://www.fwl.dfn.de/team/cb/00tools/`)

[Benecke et al. 98c] Carsten Benecke, Uwe Ellermann. "Nutzung von Kryptographie im Zusammenhang mit Firewalls". *Studie des DFN-FWL, DFN-Bericht Nr. 86*, November 1998.

[Berge et al. 96] Nils Harald Berge, Jon Ølnes. "Firewalls in an OSI-environment". *Computer Communications, Vol. 19, No. 12, S. 982-989*, Oktober 1996.

[Berners-Lee et al. 94] Tim Berners-Lee, Larry Masinter, Mark McCahill. "Uniform Resource Locators (URL)". *RFC 1738*, Dezember 1994.
(`ftp://ftp.isi.edu/in-notes/rfc1738.txt`)

[Berners-Lee et al. 95] Tim Berners-Lee, Daniel W. Connolly. "Hypertext Markup Language — 2.0". *RFC 1866*, November 1995. (ftp://ftp.isi.edu/in-notes/rfc1866.txt)

[Berners-Lee et al. 96] Tim Berners-Lee, Roy T. Fielding, Henrik Frystyk Nielsen. "Hypertext Transfer Protocol – HTTP/1.0". *RFC 1945*, Mai 1996. (ftp://ftp.isi.edu/in-notes/rfc1945.txt)

[Bianco 94] Andrea Bianco. "Performance of the TCP Protocol over ATM Networks". *Proceedings of the 3rd International Conference on Computer Communications and Networks*, San Francisco, CA, S. 170-177, September 1994.

[Blaze et al. 96] Matt Blaze, Whitfield Diffie, Ronald L. Rivest, Bruce Schneier, Tsutomu Shimomura, Eric Thompson, Michael Wiener. "Minimal Key Lengths for Symmetric Ciphers to Provide Adequate Commercial Security". Januar 1996. (ftp://ftp.cert.dfn.de/pub/docs/crypt/keylength.ps.gz)

[Blumenthal et al. 98] Uri Blumenthal, Bert Wijnen. "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)". *RFC 2274*, Januar 1998. (ftp://ftp.isi.edu/in-notes/rfc2274.txt)

[Braden 89] Bob Braden (Hg.). "Requirements for Internet Hosts – Communication Layers". *RFC 1122*, Oktober 1989. (ftp://ftp.isi.edu/in-notes/rfc1122.txt)

[Braden 92] Bob Braden. "Extending TCP for Transactions – Concepts". *RFC 1379*, November 1992. (ftp://ftp.isi.edu/in-notes/rfc1379.txt)

[Braden 94] Bob Braden. "T/TCP – TCP Extensions for Transactions: Functional Specification". *RFC 1644*, Juli 1994. (ftp://ftp.isi.edu/in-notes/rfc1644.txt)

[Braden et al. 97] Bob Braden, Lixia Zhang, Steve Berson, Shai Herzog, Sugih Jamin. "Resource ReSerVation Protocol (RSVP) — Version 1 Functional Specification". *RFC 2205*, September 1997. (ftp://ftp.isi.edu/in-notes/rfc2205.txt)

[Bradner et al. 96] Scott Bradner, Jim McQuaid. "Benchmarking Methodology for Network Interconnect Devices". *RFC 1944*, Mai 1996. (ftp://ftp.isi.edu/in-notes/rfc1944.txt)

[Brisco 95] Thomas P. Brisco. "DNS Support for Load Balancing". *RFC 1794*, April 1995. (ftp://ftp.isi.edu/in-notes/rfc1794.txt)

[Callaghan et al. 95] Brent Callaghan, Brian Pawlowski, Peter Staubach. "NFS Version 3 Protocol Specification". *RFC 1813*, Juni 1995. (ftp://ftp.isi.edu/in-notes/rfc1813.txt)

[Callon et al. 97] Ross Callon, Paul Doolan, Nancy Feldman, Andre Fredette, George Swallow, Arun Viswanathan. "A Framework for Multiprotocol Label Switching". *Internet Draft*, November 1997.
(ftp://ftp.isi.edu/internet-drafts/draft-ietf-mpls-framework-02.txt)

[Campanella et al. 96a] Mauro Campanella, Tiziana Ferrari. "TCP-UDP/IP Performance over ATM". *TERENA TF-TEN Report*, November 1996.
(http://dxcnaf.cnaf.infn.it/~ferrari/papers/myarticles/
tf-tenD11.2-report.ps.gz)

[Cansever 98] Derya H. Cansever. "NHRP Protocol Applicability Statement". *RFC 2333*, April 1998. (ftp://ftp.isi.edu/in-notes/rfc2333.txt)

[Carl-Mitchell et al. 92] Smoot Carl-Mitchell, John S. Quarterman. "Building Internet Firewalls". *UNIX World*, Februar 1992

[Case et al. 90] Jeffrey D. Case, Mark Fedor, Martin Lee Schoffstall, James R. Davin. "A Simple Network Management Protocol (SNMP)". *RFC 1157*, Mai 1990.
(ftp://ftp.isi.edu/in-notes/rfc1157.txt)

[Case et al. 96a] Jeffrey D. Case, Keith McCloghrie, Marshall T. Rose, Steven Waldbusser. "Introduction to Community-based SNMPv2". *RFC 1901*, Januar 1996. (ftp://ftp.isi.edu/in-notes/rfc1901.txt)

[Case et al. 96b] Jeffrey D. Case, Keith McCloghrie, Marshall T. Rose, Steven Waldbusser. "Structure of Management Information for Version 2 of the Simple Network Management Protocol (SNMPv2)". *RFC 1902*, Januar 1996.
(ftp://ftp.isi.edu/in-notes/rfc1902.txt)

[Case et al. 96c] Jeffrey D. Case, Keith McCloghrie, Marshall T. Rose, Steven Waldbusser. "Textual Conventions for Version 2 of the Simple Network Management Protocol (SNMPv2)". *RFC 1903*, Januar 1996.
(ftp://ftp.isi.edu/in-notes/rfc1903.txt)

[Case et al. 96d] Jeffrey D. Case, Keith McCloghrie, Marshall T. Rose, Steven Waldbusser. "Conformance Statements for Version 2 of the Simple Network Management Protocol (SNMPv2)". *RFC 1904*, Januar 1996.
(ftp://ftp.isi.edu/in-notes/rfc1904.txt)

[Case et al. 96e] Jeffrey D. Case, Keith McCloghrie, Marshall T. Rose, Steven Waldbusser. "Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2)". *RFC 1905*, Januar 1996.
(ftp://ftp.isi.edu/in-notes/rfc1905.txt)

[Case et al. 96f] Jeffrey D. Case, Keith McCloghrie, Marshall T. Rose, Steven Waldbusser. "Transport Mappings for Version 2 of the Simple Network Management Protocol (SNMPv2)". *RFC 1906*, Januar 1996.
(ftp://ftp.isi.edu/in-notes/rfc1906.txt)

[Case et al. 96g] Jeffrey D. Case, Keith McCloghrie, Marshall T. Rose, Steven Waldbusser. "Management Information Base for Version 2 of the Simple Network Management Protocol (SNMPv2)". *RFC 1907*, Januar 1996.
(`ftp://ftp.isi.edu/in-notes/rfc1907.txt`)

[Case et al. 96h] Jeffrey D. Case, Keith McCloghrie, Marshall T. Rose, Steven Waldbusser. "Coexistence between Version 1 and Version 2 of the Internet-standard Network Management Framework". *RFC 1908*, Januar 1996.
(`ftp://ftp.isi.edu/in-notes/rfc1908.txt`)

[Case et al. 98] Jeffrey Case, Dave Harrington, Randy Presuhn, Bert Wijnen. "Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)". *RFC 2272*, Januar 1998.
(`ftp://ftp.isi.edu/in-notes/rfc2272.txt`)

[CERT 88:01] CERT Coordination Center. "CERT Advisory CA-88:01 Ftpd Vulnerability". CERT Advisory, December 1988.
(`ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/CA-88:01.ftpd.hole`)

[CERT 90:01] CERT Coordination Center. "CERT Advisory CA-90:01 Sun Sendmail Vulnerability". CERT Advisory, 29. January 1990.
(`ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/CA-90:01.sun.sendmail.vulnerability`)

[CERT 92:11] CERT Coordination Center. "CERT Advisory CA-92:11 SunOS Environment Variables and setuid/setgid Vulnerability". CERT Advisory, 27. Mai 1992.
(`ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/CA-92:11:SunOS.Environment.vulnerability`)

[CERT 93:15] CERT Coordination Center. "CERT Advisory CA-93:15 `/usr/lib/sendmail`, `/bin/tar`, and `/dev/audio` Vulnerabilities". CERT Advisory, 21. Oktober 1993.
(`ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/CA-93:15.SunOS.and.Solaris.vulnerabilities`)

[CERT 93:16] CERT Coordination Center. "CERT Advisory CA-93:16 Sendmail Vulnerability". CERT Advisory, 4. November 1993.
(`ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/CA-93:16.sendmail.vulnerability`)

[CERT 94:01] CERT Coordination Center. "CERT Advisory CA-94:01 Ongoing Network Monitoring Attacks". CERT Advisory, 3. Februar 1994.
(`ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/CA-94:01.network.monitoring.attacks`)

[CERT 94:12] CERT Coordination Center. "CERT Advisory CA-94:12 Sendmail Vulnerabilities". CERT Advisory, 14. Juli 1994.

(ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/
CA-94:12.sendmail.vulnerabilities)

[CERT 95:01] CERT Coordination Center. "CERT Advisory CA-95:01 IP Spoofing Attacks and Hijacked Terminal Connections". CERT Advisory, 23. Januar 1995.
(ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/
CA-95:01.IP.spoofing)

[CERT 95:05] CERT Coordination Center. "CERT Advisory CA-95:05 Sendmail Vulnerabilities". CERT Advisory, 22. Februar 1995.
(ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/
CA-95:05.sendmail.vulnerabilities)

[CERT 95:08] CERT Coordination Center. "CERT Advisory CA-95:08 Sendmail v.5 Vulnerability". CERT Advisory, 17. August 1995.
(ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/
CA-95:08.sendmail.v.5.vulnerability)

[CERT 95:11] CERT Coordination Center. "CERT Advisory CA-95:11 Sun Sendmail -oR Vulnerability". CERT Advisory, 19. September 1995.
(ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/
CA-95:11.sun.sendmail-oR.vul)

[CERT 95:13] CERT Coordination Center. "CERT Advisory CA-95:13 Syslog Vulnerability - A Workaround for Sendmail". CERT Advisory, 19. Oktober 1995.
(ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/
CA-95:13.syslog.vul)

[CERT 96:01] CERT Coordination Center. "CERT Advisory CA-96.01 UDP Port Denial-of-Service Attack". CERT Advisory, 8. Februar 1996.
(ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/
CA-96.01.UDP_service_denial)

[CERT 96:04] CERT Coordination Center. "CERT Advisory CA-96:04 Corrupt Information from Network Servers". CERT Advisory, 22. Februar 1996.
(ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/
CA-96.04.corrupt_info_from_servers)

[CERT 96:06] CERT Coordination Center. "CERT Advisory CA-96:06 Vulnerability in NCSA/Apache CGI example code". CERT Advisory, 20. März 1996.
(ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/
CA-96.06.cgi_example_code)

[CERT 96:20] CERT Coordination Center. "CERT Advisory CA-96.20 Sendmail Vulnerabilities". CERT Advisory, 26. August 1996.
(ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/
CA-96.20.sendmail_vul)

[CERT 96:21] CERT Coordination Center. "CERT Advisory CA-96.21 TCP SYN Flooding and IP Spoofing Attacks". CERT Advisory, 19. September 1996.
(ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/
CA-96.21.tcp_syn_flooding)

[CERT 96:24] CERT Coordination Center. "CERT Advisory CA-96.24 Sendmail Daemon Mode Vulnerability". CERT Advisory, 24. September 1996.
(ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/
CA-96.24.sendmail.daemon.mode)

[CERT 96:25] CERT Coordination Center. "CERT Advisory CA-96.25 Sendmail Group Permissions Vulnerability". CERT Advisory, 10. Dezember 1996.
(ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/
CA-96.25.sendmail_groups)

[CERT 96:26] CERT Coordination Center. "CERT Advisory CA-96.26 Denial-of-Service Attack via ping". CERT Advisory, 18. Dezember 1996.
(ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/
CA-96.26.ping)

[CERT 97:04] CERT Coordination Center. "CERT Advisory CA-97.04 talkd Vulnerability". CERT Advisory, 27. Januar 1997.
(ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/
CA-97.04.talkd)

[CERT 97:05] CERT Coordination Center. "CERT Advisory CA-97.05 MIME Conversion Buffer Overflow in Sendmail Versions 8.8.3 and 8.8.4". CERT Advisory, 28. Januar 1997.
(ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/
CA-97.05.sendmail)

[CERT 97:09] CERT Coordination Center. "CERT Advisory CA-97.09 Vulnerability in IMAP and POP". CERT Advisory, 7. April 1997.
(ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/
CA-97.09.imap_pop)

[CERT 97:28] CERT Coordination Center. "CERT Advisory CA-97.28 IP Denial-of-Service Attacks". CERT Advisory, 16. Dezember 1997.
(ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/
CA-97.28.Teardrop_Land)

[CERT 98:01] CERT Coordination Center. "CERT Advisory CA-98.01 'smurf' IP Denial-of-Service Attacks". CERT Advisory, 5. Januar 1998.
(ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/
CA-98.01.smurf)

[CERT 98:05] CERT Coordination Center. "CERT Advisory CA-98.05 Multiple Vulnerabilities in BIND". CERT Advisory, 8. April 1998.

(ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/
CA-98.05.bind_problems)

[CERT 98:09] CERT Coordination Center. "CERT Advisory CA-98.09 Buffer Overflow in Some Implementations of IMAP Servers". CERT Advisory, 20. Juli 1998.
(ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/
CA-98.09.imapd)

[Chapman 92] D. Brent Chapman. " Network (In)Security Through IP Packet Filtering". Baltimore, MD, 1992
(ftp://ftp.cert.dfn.de/pub/firewalls/discussion/pkt_filtering.ps.gz)

[Chapman et al. 95] D. Brent Chapman, Elizabeth D. Zwicky. "Building Internet Firewalls". O'Reilly & Associates, September 1995.

[Chatel 96] Marc Chatel. "Classical versus Transparent IP Proxies". *RFC 1919*, März 1996. (ftp://ftp.isi.edu/in-notes/rfc1919.txt)

[Cheswick 90] William R. Cheswick. "The design of a secure Internet gateway". AT&T Bell Laboratories, 1990.
(ftp://ftp.cert.dfn.de/pub/firewalls/att/gateway.ps.gz)

[Cheswick et al. 94] William R. Cheswick, Steven M. Bellovin. "Firewalls and Internet Security: Repelling the wily Hacker". Addison-Wesley, 1994.

[Chuang 95] Shaw-Cheng Chuang. "Securing ATM Networks". 18. Oktober 1995.
(ftp://ftp.cl.cam.ac.uk/public/papers/ATM/docs-95-10/12.ps.gz)

[Cisco 97] Cisco Systems. "LocalDirector — Reference Guide". 1997.
(http://www.cisco.com/warp/public/751/lodir/lodir_rg.htm)

[Clark 82] David D. Clark. "Window and Acknowledgement Strategy in TCP". *RFC 813*, Juli 1982. (ftp://ftp.isi.edu/in-notes/rfc813.txt)

[Colajanni et al. 97] Michele Colajanni, Philip S. Yu, Daniel M. Dias. "Scheduling Algorithms for Distributed Web Servers". *17th International Conference on Distributed Computer Systems (ICDCS 1997)*, 1997.

[Cole et al. 96] Robert G. Cole, David H. Shur, Curtis Villamizar. "IP over ATM: A Framework Document". *RFC 1932*, April 1996.
(ftp://ftp.isi.edu/in-notes/rfc1932.txt)

[Comer 91] Douglas E. Comer. " Internetworking with TCP/IP, Vol 1". *Prentice Hall*, Englewood Cliffs, NJ, 1991

[Comer et al. 94] Douglas E. Comer, John C. Lin. " TCP Buffering and Performance over an ATM Network". *Purdue Technical Report CSD-TR 94-026*, März 1994.

[Coulouris et al. 94] George Coulouris, Jean Dollimore, Tim Kindberg. "Distributed Systems: Concepts and Design". Addison-Wesley, 1994.

[Cowan et al. 98] Crispin Cowan, Calton Pu, Dave Maier, Heather Hinton, Jonathan Walpole, Peat Bakke, Steve Beattie, Aaron Grier, Perry Wagle, Qian Zhang. "StackGuard: Automatic Adaptive Detection and Prevention of Buffer-Overflow Attacks". *Proceedings of 7th USENIX Security Symposium, 26.-29. Januar 1998, San Antonio, Texas*, Januar 1998.

[Danthine et al. 95] A. Danthine, O. Bonaventure. "Is ATM a Continuity or a Discontinuity for the LAN Environment?". *In Proceedings: Highspeed Networks for Multimedia Applications*, Juni 1995.

[Deering 89] Stephen E. Deering. "Host Extensions for IP Multicasting". *RFC 1112*, August 1989. (`ftp://ftp.isi.edu/in-notes/rfc1112.txt`)

[Deering et al. 95] Stephen E. Deering, Robert M. Hinden. "Internet Protocol, Version 6 (IPv6) Specification". *RFC 1883*, Dezember 1995.
(`ftp://ftp.isi.edu/in-notes/rfc1883.txt`)

[Deng et al. 95] Robert H. Deng, Li Gong, Aurel A. Lazar. "Secure Data Transfer in Asynchronous Transfer Mode Networks". *In: Proceedings of IEEE Globecom '95, Singapore*, November 1995.
(`http://www.csl.sri.com/~gong/papers/globecom95atm.ps.Z`)

[Dennis 96] Ronald Michael Dennis. "Internetworking: Integrating IP/ATM LAN/WAN Security". *Naval Postgraduate School*, Monterey, CA, September 1996.
(`http://www.stl.nps.navy.mil/~iirg/dennis/thesis.htm`)

[DFN-CERT 98] DFN-CERT. "Projektbericht des DFN-NSCC". 1998.

[Dias et al. 96] Daniel M. Dias, William Kish, Rajat Mukherjee, Renu Tewari. "A Scalable and Highly Available Web Server". *41st IEEE Computer Society International Conference (COMPCON)*, 1996.

[Dierks et al. 97] Tim Dierks, Alan O. Freier, Philip L. Karlton, Paul C. Kocher. "The TLS Protocol: Version 1.0". *Internet Draft*, 12. November 1997.

[DoD 85] Hg.: Department of Defense. "Trusted Computer System Evaluation Criteria". DoD Standard 5200.28, ("Orange Book"), 1985.
(`ftp://ftp.cert.dfn.de/pub/docs/rainbow-series/std001.txt.gz`)

[Dorling et al. 96] Brian Dorling, Daniel Freedman, Chris Metz, Jaap Burger. "Internetworking over ATM: An Introduction". Prentice Hall, 1996.

[Dusse et al. 98a] Steve Dusse, Paul Hoffman, Blake Ramsdell, Laurence Lundblade, Lisa Repka. "S/MIME Version 2 Message Specification". *RFC 2311*, März 1998. (`ftp://ftp.isi.edu/in-notes/rfc2311.txt`)

[Dusse et al. 98b] Steve Dusse, Paul Hoffman, Blake Ramsdell, Jeff Weinstein. "S/MIME Version 2 Certificate Handling". *RFC 2312*, März 1998. (`ftp://ftp.isi.edu/in-notes/rfc2312.txt`)

[Eastlake et al. 97] Donald E. Eastlake 3rd, Charles W. Kaufman. "Domain Name System Security Extensions". *RFC 2065*, Januar 1997. (`ftp://ftp.isi.edu/in-notes/rfc2065.txt`)

[Eberle 92] Hans Eberle. "A High-speed DES Implementation for Network Applications". DEC Systems Research Center (SRC) Report # 090, 23. September 1992. (`ftp://ftp.cert.dfn.de/pub/docs/crypt/SRC-090.ps.gz`)

[Egevang et al. 94] Kjeld Borch Egevang, Paul Francis. "The IP Network Address Translator (NAT)". *RFC 1631*, Mai 1994. (`ftp://ftp.isi.edu/in-notes/rfc1631.txt`)

[Ellermann et al. 91] Uwe Ellermann, Klaus-Peter Kossakowski, Antonio Liu, Daniel Löffler, Astrid Ritscher, Gerald Schrod, Patrick Stöckmann. "Bericht der Arbeitsgruppe Computer-Würmer". *AGN Bericht, Universität Hamburg*, 1991.

[Ellermann 93] Uwe Ellermann. "Ein Firewall am Fachbereich Informatik". Universität Hamburg, Dezember 1993.

[Ellermann 94] Uwe Ellermann. "Firewalls: Isolations- und Audittechniken zum Schutz von lokalen Computer-Netzen". *DFN-Bericht Nr. 76*, September 1994.

[Ellermann 95] Uwe Ellermann. "Netzabsicherung durch Firewalls". *4. IT-Sicherheitskongreß, 8. bis 11. Mai 1995, Bonn*, Mai 1995. (`http://www.cert.dfn.de/team/ue/fw/fire/`)

[Ellermann 96] Uwe Ellermann. "IPv6 and Firewalls". Proceedings of SECURICOM — 14th International Congress on Computer and Communications Security Protection, Paris, Juni 1996. (`http://www.fwl.dfn.de/eng/~ue/fw/ipv6fw/`)

[Ellermann et al. 98a] Uwe Ellermann, Carsten Benecke. "Firewalls in ATM-Netzen: Vorbericht". *DFN-FWL Forschungsbericht*, Hamburg, Februar 1998.

[Ellermann et al. 98b] Uwe Ellermann, Carsten Benecke. "Parallele Firewalls: Skalierbare Lösungen für Hochgeschwindigkeitsnetze". *5. DFN-CERT Workshop 1998 "Sicherheit in vernetzten Systemen", 4. und 5. März 1998, Hamburg, In: DFN-Bericht Nr. 85*, Mai 1998. (`http://www.fwl.dfn.de/team/ue/fw/par-fire/`)

[Ellermann et al. 98c] Uwe Ellermann, Carsten Benecke. "Firewalls für Hochgeschwindigkeitsnetze". *Deutscher Internet Kongress 1998, 5. und 6. Mai 1998, CongressCenter Messe Frankfurt.*, Mai 1998. (`http://www.fwl.dfn.de/team/ue/fw/fire-hsn/`)

[Ellermann et al. 98d] Uwe Ellermann, Carsten Benecke. "Firewalls for ATM Networks". *INFOSECcom, 4. und 5. Juni 1998, CNIT Paris–La Défense, Frankreich.*, Juni 1998. (`http://www.fwl.dfn.de/eng/team/ue/fw/fire-atm/`)

[Engelschall 98] Ralf S. Engelschall. "Load Balancing Your Web Site: Practical Approaches for Distributing HTTP Traffic". *Web Techniques Magazine*, Vol. 3, Issue 5, Mai 1998.
(`http://www.webtechniques.com/features/1998/05/engelschall/engelschall.shtml`)

[Ferguson et al. 98] Paul Ferguson, Daniel Senie. "Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing". *RFC 2267*, Januar 1998. (`ftp://ftp.isi.edu/in-notes/rfc2267.txt`)

[Fielding et al. 97] Roy T. Fielding, Jim Gettys, Jeffrey C. Mogul, Henrik Frystyk Nielsen, Tim Berners-Lee. "Hypertext Transfer Protocol – HTTP/1.1". *RFC 2068*, Januar 1997. (`ftp://ftp.isi.edu/in-notes/rfc2068.txt`)

[FreeBSD-SA-98:03] FreeBSD, Inc. "FreeBSD Security Advisory SA-98:03: Problems with TTCP". FreeBSD Security Advisory, 18. Mai 1998.
(`ftp://ftp.cert.dfn.de/pub/vendor/freebsd/advisories/FreeBSD-SA-98:03.ttcp.asc`)

[Freier et al. 96] Alan O. Freier, Philip Karlton, Paul C. Kocher. "The SSL Protocol: Version 3.0". *Internet Draft*, November 1996.
(`http://www.netscape.com/eng/ssl3/draft302.txt`)

[Gage 98] Chris Gage. "IBM eNetwork Dispatcher Version 2.0: Olympic-scale TCP/IP Load-balancing and Availability". *IBM Whitepaper*, Juni 1998.
(`ftp://ftp.software.ibm.com/software/enetwork/dispatcher/whitepapers/end20wp.pdf`)

[Garfinkel 95] Simpson Garfinkel. *PGP: Pretty Good Privacy.* O'Reilly & Associates, Inc., 1995.

[Garfinkel et al. 96] Simpson Garfinkel, Gene Spafford. *Practical UNIX & Internet Security.* O'Reilly & Associates, Inc., 1996.

[Gettys et al. 98] James Gettys, Henrik Frystyk Nielsen. "The WebMUX Protocol". *Internet Draft / W3C Working Draft*, August 1998.
(`http://www.w3.org/TR/WD-mux`)

[Gilmore 97] John Gilmore. "Security for the Domain Name System". *In: Web Security: A Matter of Trust*, World Wide Web Journal: Vol. 2, #3, O'Reilly, 1997.
(`http://www.w3journal.com/7/s3.gilmore.wrap.html`)

[Harkins et al. 98] Dan Harkins, Dave Carrel. "The Internet Key Exchange (IKE)". *RFC 2409*, November 1998. (`ftp://ftp.isi.edu/in-notes/rfc2409.txt`)

[Heidemann 97] John Heidemann. "Performance Interactions Between P-HTTP and TCP Implementations". *ACM Computer Communication Review*, 27 2, S. 65-73, April, 1997. (`http://www.isi.edu/~johnh/PAPERS/Heidemann97a.html`)

[Hein 97] Mathias Hein. "ATM: Konzepte, Trends, Migration". International Thomson Publishing, 1997.

[Hein et al. 98] Mathias Hein, Theo Vollmer (Hrsg.). "Bay Networks ATM LAN Guide". *FOSSIL-Verlag*, 2. erweiterte und aktualisierte Auflage, 1998.

[Heinanen 93] Juha Heinanen. "Multiprotocol Encapsulation over ATM Adaptation Layer 5". *RFC 1483*, Juli 1993.
(`ftp://ftp.isi.edu/in-notes/rfc1483.txt`)

[Hinden et al. 95] Robert M. Hinden, Stephen E. Deering. "IP Version 6 Addressing Architecture". *RFC 1884*, Dezember 1995.
(`ftp://ftp.isi.edu/in-notes/rfc1884.txt`)

[Hoffman 90] Lance J. Hoffman. *Rogue Programs: Viruses, Worms and Trojan Horses.* Van Nostrand Reinhold, 1990.

[Howard 97] John D. Howard. "An Analysis Of Security Incidents On The Internet". *Carnegie Mellon University*, April 1997.
(`http://www.cert.org/research/JHThesis/Start.html`)

[Hudson et al. 95] Tim J. Hudson, Eric A. Young. "SSLeay: Programmer Reference". *Draft*, July 1995.
(`ftp://ftp.cert.dfn.de/pub/tools/net/ssleay/SSLeay.doc-1.2.tar.gz`)

[Hughes 96] James Hughes. "A High Speed Firewall Architecture for ATM/OC-3c". *Interop Engineering Conference, Spring 1996, Las Vegas*, 9. Februar 1996.
(`http://www.network.com/~hughes/Interop-eng96-1.0.ps`)

[Huitema 96] Christian Huitema. "IPv6: The new Internet Protocol". Prentice Hall, 1996.

[ISO 89] International Organization for Standardization (ISO). "Information Processing Systems — Open Systems Interconnection — Basic Reference Model — Part 2: Security Architecture". *International Organization for Standardization, ISO 7498-2,* First Edition, 1989.

[ISO 97] International Organization for Standardization (ISO). "Answers to the DSM-CC FAQ Version 1.0". *International Organization for Standardization*, Februar 1997 (`http://drogo.cselt.stet.it/mpeg/faq/faq_dsm-cc.htm`)

[ITU 93] International Telecommunication Union (ITU). "General Aspects of Digital Transmission Systems, Terminal Equipments: Pulse Code Modulation (PCM) of Voice Frequencies". *ITU-T Recommendation G.711*, 1993

[ITU 97] International Telecommunication Union (ITU). "Series H: Audiovisual and Multimedia Systems, Infrastructure of Audiovisual Services – Communication Procedures: Control Protocol for Multimedia Communication". *ITU-T Recommendation H.245*, Juli 1997

[Jacobson et al. 92] Van Jacobson, Bob Braden, Dave Borman. "TCP Extensions for High Performance". *RFC 1323*, Mai 1992.
(ftp://ftp.isi.edu/in-notes/rfc1323.txt)

[Joncheray 95] Laurent Joncheray. "A Simple Active Attack Against TCP". Proceedings of the Fifth USENIX UNIX Security Symposium, Salt Lake City, Utah, 5.-7. Juni 1995.

[Kaliski et al. 95] Burt Kaliski, Matt Robshaw. "Message Authentication with MD5". *RSA Laboratories' CryptoBytes*, 1(1), 1995.
(http://www.rsa.com/rsalabs/cryptobytes/spring95/md5.htm)

[Kantor 91] Brian Kantor. "BSD Rlogin". *RFC 1282*, Dezember 1991.
(ftp://ftp.isi.edu/in-notes/rfc1282.txt)

[Kaufman et al. 95] Charlie Kaufman, Radia Perlman, Mike Speciner. "Network Security: Private Communication in a Public World". Prentice Hall, 1995.

[Kent et al. 87] Christopher A. Kent, Jeffrey C. Mogul. "Fragmentation Considered Harmful". DEC Western Research Laboratory (WRL) Research Report 87/3, Dezember 1987.
(ftp://gatekeeper.dec.com/pub/DEC/WRL/research-reports/WRL-TR-87.3.ps)

[Keshav 97] Srinivasan Keshav. "An engineering Approach to Computer Networking: ATM Networks, the Internet, and the Telephone Network". Addison-Wesley, 1997.

[Koblas et al. 92] David Koblas, Michelle R. Koblas. "SOCKS". 1992.
(ftp://ftp.cert.dfn.de/pub/tools/net/socks/socks_usenix_paper.ps.gz)

[Kohl et al. 93] John Kohl, Clifford Neuman. "The Kerberos Network Authentication Service (V5)". *RFC 1510*, September 1993.
(ftp://ftp.isi.edu/in-notes/rfc1510.txt)

[Kossakowski 92] Klaus-Peter Kossakowski. "Klassifikation und Abwehr von Computer-Würmern in Netzwerken". *Diplomarbeit am Fachbereich Informatik der Universität Hamburg*, 1992.

[Kossakowski 98] Klaus-Peter Kossakowski. "Virtuelle Private Netzwerke: Einsatzmöglichkeiten und Techniken". *5. DFN-CERT Workshop 1998 "Sicherheit in vernetzten Systemen", 4. und 5. März 1998, Hamburg, In: DFN-Bericht Nr. 85*, Mai 1998.

[Kostick et al. 95] Chris Kostick, Matt Mancuso. "Firewall Performance Analysis Report". *Computer Sciences Corporation*, 10. August 1995.

[Kowalski 96] BJ Kowalski. "ATLAS Policy Cache Architecture". *StorageTek White Paper*, November 1996.
(http://www.network.com/Products/Security/ATLAS/TechBriefs/)

[Krawczyk et al. 97] Hugo Krawczyk, Mihir Bellare, Ran Canetti. "HMAC: Keyed-Hashing for Message Authentication". *RFC 2104*, Februar 1997. (ftp://ftp.isi.edu/in-notes/rfc2104.txt)

[Laubach 94] Mark Laubach. "Classical IP and ARP over ATM". *RFC 1577*, Januar 1994. (ftp://ftp.isi.edu/in-notes/rfc1577.txt)

[Laubach et al. 98] Mark Laubach, Joel Halpern. "Classical IP and ARP over ATM". *RFC 2225*, April 1998. (ftp://ftp.isi.edu/in-notes/rfc2225.txt)

[Laurent 96] Maryline Laurent. "Security Flows Analysis of the ATM Emulated LAN Architecture". *In: Proceedings of IFIP Conference on Communications and Multimedia Security 1996*, Essen, 1996. (ftp://ftp.rennes.enst-bretagne.fr/pub/security/ml_IFIP96.ps.gz)

[Laurent et al. 97a] Maryline Laurent, O. Paul, Pierre Rolin. "Securing communications over ATM networks". *In: Proceedings of IFIPSEC 13th International Information Security Conference 1997*, Kopenhagen, Dänemark, 1997. (ftp://ftp.rennes.enst-bretagne.fr/pub/security/ml_IFIPSEC97.ps.gz)

[Laurent et al. 97b] Maryline Laurent, Pierre Rolin, Laurent Stoffel. "Security Mechanisms within Control Plane". *ATM Forum, Technical Committee*, ATM Forum/97-0040, Draft vom Februar 1997. (ftp://ftp.rennes.enst-bretagne.fr/pub/security/ml_AF97-0040.ps.gz)

[Laurent et al. 97c] Maryline Laurent, Laurent Stoffel. "Signaling Confidentiality". *ATM Forum, Technical Committee*, ATM Forum/97-0803, Draft vom September 1997. (ftp://ftp.rennes.enst-bretagne.fr/pub/security/ml_AF97-0803.ps.gz)

[Laurent et al. 97d] Maryline Laurent, Laurent Stoffel. "Anonymity within the Control Plane". *ATM Forum, Technical Committee*, ATM Forum/97-0804, Draft vom Februar 1997. (ftp://ftp.rennes.enst-bretagne.fr/pub/security/ml_AF97-0804.ps.gz)

[Leech et al. 96] Marcus Leech, Matt Ganis, Ying-Da Lee, Ron Kuris, David Koblas, LaMont Jones. "SOCKS Protocol Version 5". *RFC 1928*, März 1996. (ftp://ftp.isi.edu/in-notes/rfc1928.txt)

[Levi et al. 98] David B. Levi, Paul Meyer, Bob Stewart. "SNMPv3 Applications". *RFC 2273*, Januar 1998. (ftp://ftp.isi.edu/in-notes/rfc2273.txt)

[Lipinski 98] Klaus Lipinski (Hrsg.). "Lexikon ATM". *International Thomson Publishing*, 1998

[Luciani et al. 98a] James V. Luciani, Dave Katz, David Piscitello, Bruce Cole, Naganand Doraswamy. "NBMA Next Hop Resolution Protocol (NHRP)". *RFC 2332*, April 1998. (ftp://ftp.isi.edu/in-notes/rfc2332.txt)

[Luciani et al. 98b] James V. Luciani, Grenville Armitage, Joel M. Halpern, Naganand Doraswamy. "Server Cache Synchronization Protocol (SCSP)". *RFC 2334*, April 1998. (ftp://ftp.isi.edu/in-notes/rfc2334.txt)

[Luciani 98] James V. Luciani. "Classical IP to NHRP Transition". *RFC 2336*, Juli 1998. (ftp://ftp.isi.edu/in-notes/rfc2336.txt)

[Mastercard et al. 97a] Mastercard, Visa. "SET Secure Electronic Transaction Specification: Book 1: Business Description". 31. Mai 1997. (http://www.setco.org/download.html)

[Mastercard et al. 97b] Mastercard, Visa. "SET Secure Electronic Transaction Specification: Book 2: Programmer's Guide". 31. Mai 1997. (http://www.setco.org/download.html)

[Mastercard et al. 97c] Mastercard, Visa. "SET Secure Electronic Transaction Specification: Book 3: Formal Protocol Definition". 31. Mai 1997. (http://www.setco.org/download.html)

[Maughan et al. 98] Douglas Maughan, Mark Schertler, Mark Schneider, Jeff Turner. "Internet Security Association and Key Management Protocol (ISAKMP)". *RFC 2408*, November 1998. (ftp://ftp.isi.edu/in-notes/rfc2408.txt)

[McCloghrie 96] Keith McCloghrie. "An Administrative Infrastructure for SNMPv2". *RFC 1909*, Februar 1996. (ftp://ftp.isi.edu/in-notes/rfc1909.txt)

[Metcalfe 73] Bob Metcalfe. "The Stockings were hung by the Chimney with Care". *RFC 602*, December 1973. (ftp://ftp.isi.edu/in-notes/rfc602.txt)

[Metzger et al. 95a] Perry Metzger, William Allen Simpson. "IP Authentication using Keyed MD5". *RFC 1828*, August 1995. (ftp://ftp.isi.edu/in-notes/rfc1828.txt)

[Metzger et al. 95b] Perry Metzger, William Allen Simpson. "IP Authentication using Keyed SHA". *RFC 1852*, September 1995. (ftp://ftp.isi.edu/in-notes/rfc1852.txt)

[Minoli et al. 97] Daniel Minoli, Thomas W. Golway, Norris P. Smith. "Planning and Managing ATM Networks". Manning, 1997.

[Mockapetris 87a] Paul Mockapetris "Domain Names - Concepts and Facilities". *RFC 1034*, November 1987. (ftp://ftp.isi.edu/in-notes/rfc1034.txt)

[Mockapetris 87b] Paul Mockapetris. "Domain Names - Implementation and Specification". *RFC 1035*, November 1987. (ftp://ftp.isi.edu/in-notes/rfc1035.txt)

[Mogul 89] Jeffrey C. Mogul. "Simple and Flexible Datagram Access Controls for Unix-based Gateways". DEC Western Research Laboratory (WRL) Research Report 89/4, März 1995. (ftp://gatekeeper.dec.com/pub/DEC/WRL/research-reports/WRL-TR-89.4.ps)

[Mogul 95] Jeffrey C. Mogul. "The Case for Persistent-Connection HTTP". DEC Western Research Laboratory (WRL) Research Report 95/4, Mai 1995. (ftp://gatekeeper.dec.com/pub/DEC/WRL/research-reports/WRL-TR-95.4.ps)

[Moldeklev et al. 95] Kjersti Moldeklev, Per Gunningberg. "How a large ATM MTU causes deadlocks in TCP data transfers". *IEEE/ACM Transactions on Networking*, Vol. 3, No. 4 (Aug. 1995), pp. 409-422. August 1995. (http://cs.weber.edu/htdocs/class-docs/CSx71/tcpdeadlock.ps)

[Molitor] Andrew Molitor. "Measuring Firewall Performance". (http://www.clark.net/pub/mjr/pubs/fwperf/molitor.htm)

[Morris 85] Robert T. Morris. "A Weakness in the 4.2BSD Unix TCP/IP Software". AT&T Bell Laboratories, Feb 25, 1985. (ftp://ftp.cert.dfn.de/pub/docs/net/weakTCP.ps.gz)

[Nagle 84] John Nagle. "Congestion Control in IP/TCP Internetworks". *RFC 896*, Januar 1984. (ftp://ftp.isi.edu/in-notes/rfc896.txt)

[Nahum et al. 94] Erich M. Nahum, David J. Yates, James F. Kurose, Don Towsley. "Performance Issues in Parallelized Network Protocols". *In: USENIX Symposium on Operating Systems Design and Implementation, Monterey, Ca.*, November 1994. (ftp://gaia.cs.umass.edu/pub/Nahu94:Performance.ps.Z)

[Narten et al. 96] Thomas Narten, Erik Nordmark, William Allen Simpson. "Neighbor Discovery for IP Version 6 (IPv6)". *RFC 1970*,August 1996. (ftp://ftp.isi.edu/in-notes/rfc1970.txt)

[Nessett et al. 98] Dan Nessett, Polar Humenn. "The Multilayer Firewall". *In: Symposium on Network and Distributed System Security, San Diego, Ca. März 1998*, 1998.

[Newman et al. 96] Peter Newman, W. L. Edwards, Robert M. Hinden, Eric Hoffman, Fong Ching Liaw, Tom Lyon, Greg Minshall. "Transmission of Flow Labelled IPv4 on ATM Data Links Ipsilon Version 1.0". *RFC 1954*, Mai 1996. (ftp://ftp.isi.edu/in-notes/rfc1954.txt)

[Newman et al. 97a] Peter Newman, Greg Minshall, Tom Lyon. "IP Switching: ATM under IP". *Ipsilon Networks Inc.*, 1997. (http://www.iprg.nokia.com/about/technology/papers/newman0001.htm)

[Newman et al. 97b] Peter Newman, Greg Minshall, Tom Lyon, Larry Huston. "IP Switching and Gigabit Routers". *IEEE Communications*, Januar 1997. (http://www.iprg.nokia.com/about/technology/papers/ieee_comm96.htm)

[Nielsen et al. 97] Henrik Frystyk Nielsen, Jim Gettys, Anselm Baird-Smith, Eric Prud'hommeaux, Håkon Wium Lie Chris Lilley. "Network Performance Effects of HTTP/1.1, CSS1, and PNG". *W3 Note*, 14. Februar 1997. (http://www.w3.org/pub/www/protocols/HTTP/Performance/Pipeline.html)

[NIST 95] National Institute of Standards and Technology (NIST). "Secure Hash Standard". *Federal Information Processing Standards Publication 180-1*, April 1995. (ftp://ftp.cert.dfn.de/pub/csir/nist/fips/fip180-1.txt)

[Nowicki 91] Bill Nowicki. "NFS: Network File System Protocol Specification". *RFC 1094*, März 1989. (ftp://ftp.isi.edu/in-notes/rfc1094.txt)

[NSF 95] Hg. National Science Foundation. "Internet Statistics". 1995. (ftp://nis.nsf.net/statistics/nsfnet)

[Orman 98] Hilarie K. Orman. "The OAKLEY Key Determination Protocol". *RFC 2412*, November 1998. (ftp://ftp.isi.edu/in-notes/rfc2412.txt)

[Perez et al. 95] Maryann Perez Maher, Fong-Ching Liaw, Allison Mankin, Eric Hoffman, Dan Grossman, Andrew G. Malis. "ATM Signaling Support for IP over ATM". *RFC 1755*, Februar 1995. (ftp://ftp.isi.edu/in-notes/rfc1755.txt)

[Peterson et al. 96] Larry L. Peterson, Bruce S. Davie. "Computer Networks: A Systems Approach". Morgan Kaufman Publishers, 1996.

[Plummer 82] David C. Plummer. "An Ethernet Address Resolution Protocol". *RFC 826*, November 1982. (ftp://ftp.isi.edu/in-notes/rfc826.txt)

[Postel 81a] Jonathan B. Postel. "Internet Control Message Protocol". *RFC 792*, September 1981. (ftp://ftp.isi.edu/in-notes/rfc792.txt)

[Postel 81b] Jonathan B. Postel. "Transmission Control Protocol". *RFC 793*, September 1981. (ftp://ftp.isi.edu/in-notes/rfc793.txt)

[Postel 82] Jonathan B. Postel. "Simple Mail Transfer Protocol". *RFC 821*, August 1982. (ftp://ftp.isi.edu/in-notes/rfc821.txt)

[Postel 83a] Jonathan B. Postel. "Echo Protocol". *RFC 862*, Mai 1983. (ftp://ftp.isi.edu/in-notes/rfc862.txt)

[Postel 83b] Jonathan B. Postel. "Character Generator Protocol". *RFC 864*, Mai 1983. (ftp://ftp.isi.edu/in-notes/rfc864.txt)

[Postel 83c] Jonathan B. Postel. "Daytime Protocol". *RFC 867*, Mai 1983. (ftp://ftp.isi.edu/in-notes/rfc867.txt)

[Postel et al. 85] Jonathan B. Postel, Joyce Reynolds. "File Transfer Protocol (FTP)". *RFC 959*, Oktober 1985. (ftp://ftp.isi.edu/in-notes/rfc959.txt)

[Ranum 92a] Marcus J. Ranum. "A Network Firewall". Digital Equipment Corporation, 1992 (ftp://ftp.cert.dfn.de/pub/firewalls/dec/firewall.ps.gz)

[Ranum 92b] Marcus J. Ranum. "Thinking about Firewalls". TIS, Greenbelt, MD, 1992 (ftp://ftp.cert.dfn.de/pub/firewalls/tis/thinking.about.firewalls.ps.gz)

[Rathgeb et al. 97] Erwin Rathgeb, Eugen Wallmeier. "ATM-Infrastruktur für die Hochleistungskommunikation". *Springer*, 1997.

[Rekhter et al. 96] Yakov Rekhter, Robert G. Moskowitz, Daniel Karrenberg, Geert Jan de Groot, Eliot Lear. "Address Allocation for Private Internets". *RFC 1918*, Februar 1996. (`ftp://ftp.isi.edu/in-notes/rfc1918.txt`)

[Rekhter et al. 97] Yakov Rekhter, Bruce Davie, Dave Katz, Eric Rosen, George Swallow. "Cisco Systems' Tag Switching Architecture Overview". *RFC 2105*, Februar 1997. (`ftp://ftp.isi.edu/in-notes/rfc2105.txt`)

[Rekhter et al. 98] Yakov Rekhter, Joel Halpern. "NHRP for Destinations off the NBMA Subnetwork". *Internet Draft*, Juni 1998.

[Reynolds 89] Joyce K. Reynolds. "The Helminthiasis of the Internet". *RFC 1135*, Dezember 1989. (`ftp://ftp.isi.edu/in-notes/rfc1135.txt`)

[Rivest 92] Ronald L. Rivest. "The MD5 Message-Digest Algorithm". *RFC 1321*, April 1992. (`ftp://ftp.isi.edu/in-notes/rfc1321.txt`)

[Rochlis et al. 89] Jon A. Rochlis, Mark W. Eichin. "With Microscope and Tweezers: An Analysis of the Internet Virus of November 1988". *IEEE Symposium on Research in Security and Privacy*, 1989.
(`ftp://ftp.cert.dfn.de/pub/docs/worm/internet/mit.ps.gz`)

[Rosen et al. 98] Eric C. Rosen, Arun Viswanathan, Ross Callon. "Multiprotocol Label Switching Architecture". *Internet Draft*, Juli 1998.
(`ftp://ftp.isi.edu/internet-drafts/draft-ietf-mpls-arch-02.txt`)

[Sachs 92] L. Sachs. "Angewandte Statistik. Anwendung statistischer Methoden". 7. Auflage, Springer, 1992

[Saltzer et al. 76] Jerome H. Saltzer, Michael D. Schroeder. "The Protection of Information in Computer Systems". *Proceedings of the IEEE, 63, 9*, 1975.
(`http://web.mit.edu/Saltzer/www/publications/protection/Basic.html`)

[Salus 95] Peter H. Salus. "Early insecurity". *Network Security 1995, Washington D.C.*, 1995.

[Scharf 96] Achim Scharf. "ATM ohne Geheimnis". Heise, 1996.

[Schill et al. 97] Alexander Schill, Robert Hess, Sascha Kümmel, Dieter Hege, Holger Lieb. "ATM-Netze in der Praxis: Einsatzszenarien, Produktkategorien, aktuelle Standards". Addison-Wesley, 1997.

[Schneier 96] Bruce Schneier. *Applied Cryptography Second Edition: Protocols, Algorithms, and Source Code in C.* John Wiley & Sons, 1996.

[Schuba et al. 97] Christoph L. Schuba, Ivan V. Krsul, Markus G. Kuhn, Eugene H. Spafford, Aurobindo Sundaram, Diego Zamboni. "Analysis of a Denial of Service Attack on TCP". *1997 IEEE Symposium on Security and Privacy*, 1997. (`ftp://ftp.cert.dfn.de/pub/docs/net/synkill.ps.gz`)

[Schulzrinne et al. 96] Henning Schulzrinne, Stephen L. Casner, Ron Frederick, Van Jacobson. "RTP: A Transport Protocol for Real-Time Applications". *RFC 1889*, Januar 1996. (ftp://ftp.isi.edu/in-notes/rfc1889.txt)

[Schulzrinne et al. 98] Henning Schulzrinne, Anup Rao, Robert Lanphier. "Real Time Streaming Protocol (RTSP)". *RFC 2326*, April 1998. (ftp://ftp.isi.edu/in-notes/rfc2326.txt)

[Spafford 88] Eugene H. Spafford. "The Internet Worm Program: An Analysis". *Purdue Technical Report CSD-TR-823*, 1988. (ftp://ftp.cert.dfn.de/pub/docs/worm/internet/spafford.ps.gz)

[Spafford 90] Eugene H. Spafford. "The Internet Worm Incident". *In: Rogue Programs [Hoffman 90]*, 1990. (ftp://ftp.cert.dfn.de/pub/docs/worm/internet/IWorm2.ps.gz)

[Srisuresh et al. 98] Pyda Srisuresh, Der-hwa Gan. "Load Sharing using IP Network Address Translation (LSNAT)". *RFC 2391*, August 1998. (ftp://ftp.isi.edu/in-notes/rfc2391.txt)

[Steiner et al. 88] Jennifer G. Steiner, B. Clifford Neuman, Jeffrey I. Schiller. "Kerberos : An Authentication Service for Open Network Systems". *Proceedings of the Winter 1988 USENIX Conference, Dallas, TX*, 1988. (ftp://ftp.cert.dfn.de/pub/docs/kerberos/kerberos-article.ps.gz)

[Stevens 97] W. Richard Stevens. "TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms". *RFC 2001*, Januar 1997. (ftp://ftp.isi.edu/in-notes/rfc2001.txt)

[Stevens 98] W. Richard Stevens. " UNIX Network Programming". *Prentice Hall*, Englewood Cliffs, NJ, 1998

[Stevenson et al. 95a] Daniel Stevenson, Nathan Hillery, Greg Byrd, Fengmin Gong, Dan Winkelstein. "Design of a Key Agile Cryptographic System for OC-12c Rate ATM". *In: Symposium on Network and Distributed System Security, San Diego, Ca. 16.-17. Februar 1995*, 1995.

[Stevenson et al. 95b] Daniel Stevenson, Nathan Hillery, Greg Byrd. "Secure Communications in ATM Networks". *Communications of the ACM, Vol. 38, No. 2*, Februar 1995.

[St. Johns 93] Michael C. St. Johns. "Identification Protocol". *RFC 1413*, Februar 1993. (ftp://ftp.isi.edu/in-notes/rfc1413.txt)

[Tarman 96] Thomas D. Tarman (Ed.). "Phase I ATM Security Specification". *ATM Forum, Technical Committee*, ATM Forum/95-1473R2, Draft vom 15. April 1996.

[Touch 97] Joseph D. Touch. "TCP Control Block Interdependence". *RFC 2140*, April 1997. (ftp://ftp.isi.edu/in-notes/rfc2140.txt)

[Treml 98] Uli Treml. Private E-Mail Kommunikation. *B-WiN Labor*, Februar 1998.

[US GAO 89] United States General Accounting Office. "Computer Security: Virus Highlights Need for improved Internet Management". *GAO/IMTEC-89-57*, 1989.
(`ftp://ftp.cert.dfn.de/pub/docs/worm/internet/gao-imtec87-57.txt.gz`)

[Varadharajan et al. 97] Vijay Varadharajan, Rajan Shankaran, Michael Hitchens. "Security Issues in Asynchronous Transfer Mode". *Proceedings of Second Australasian Conference, ACISP'97, Sydney July 1997*, Springer, 1997.

[Vilela 98] José Vieira Vilela. "Security in ATM Networks". *TERENA TF-TEN Report*, 1998.
(`http://www.terena.nl/task-forces/tf-ten/hypermail/1998-04/0023.html`)

[Wagner et al. 96] David Wagner, Bruce Schneier. "Analysis of the SSL 3.0 protcol". *2nd USENIX Workshop on Electronic Commerce*, Revised 19. November 1996.
(`ftp://ftp.cert.dfn.de/pub/docs/crypt/ssl3.0.ps.gz`)

[Waters 96] Glenn W. Waters. "User-based Security Model for SNMPv2". *RFC 1910*, Februar 1996. (`ftp://ftp.isi.edu/in-notes/rfc1910.txt`)

[Wijnen et al. 98] Bert Wijnen, Randy Presuhn, Keith McCloghrie. "View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)". *RFC 2275*, Januar 1998.
(`ftp://ftp.isi.edu/in-notes/rfc2275.txt`)

[Xu et al. 98a] Jun Xu, Mukesh Singhal. "Design of a High-Performance ATM Firewall". In Vorbereitung für: *5th ACM Conference on Computer and Communication Security, San Francisco, 2.-5. November 1998*, 1998.
(`ftp://ftp.cis.ohio-state.edu/pub/tech-report/1998/TR13.ps.gz`)

[Xu et al. 98b] Jun Xu, Mukesh Singhal. "A Firewalling Scheme for Securing MPOA-based Corporate Intranets". In Vorbereitung für: *3rd IEEE High-Assurance Systems Engineering Symposium, Washington, DC, 13.-14. November 1998*, 1998. (`ftp://ftp.cis.ohio-state.edu/pub/tech-report/1998/TR19.ps.gz`)

[Ylönen 96a] Tatu Ylönen. "SSH — Secure Login Connections over the Internet". *The Sixth USENIX Security Symposium Proceedings, San Jose, Ca. 22.-25. Juli 1996*, 7. Juni 1996.

[Zitterbart 91] Martina Zitterbart. "Funktionsbezogene Parallelität in transportorientierten Kommunikationsprotokollen". *Fortschritt-Berichte VDI, Reihe 10, Nr. 183*, 1991.

B Zusätzliche Abbildungen

Dieser Anhang enthält die zusätzliche Abbildungen, für die im Fließtext kein Platz war oder die in den einzelnen Abschnitten nur verkleinert wiedergegeben werden konnten.

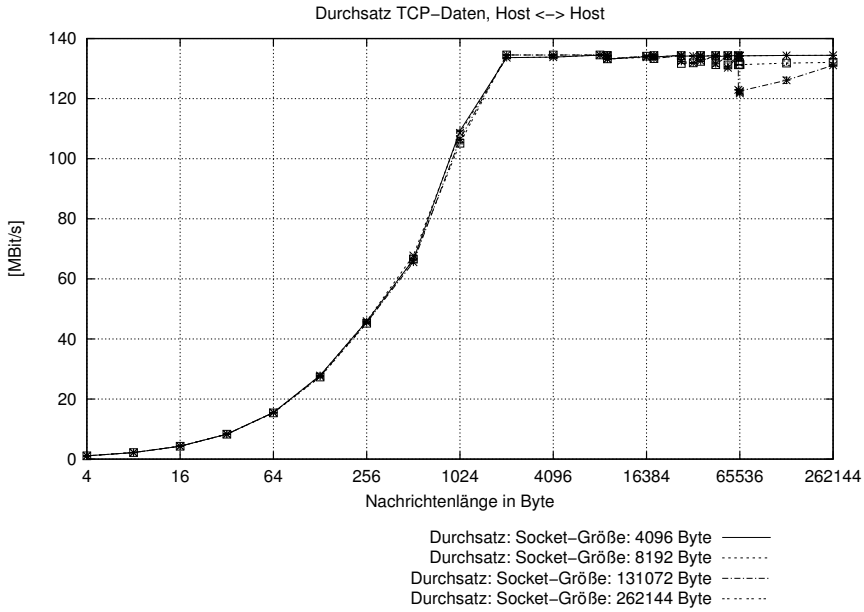


Abbildung B.1: Maximaler Durchsatz über TCP (1)

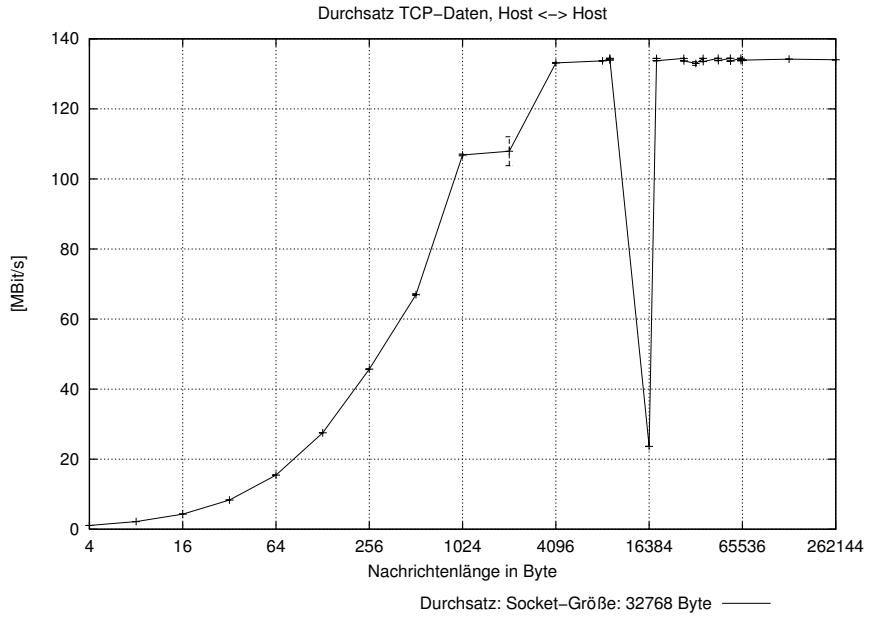


Abbildung B.2: Maximaler Durchsatz über TCP (2)

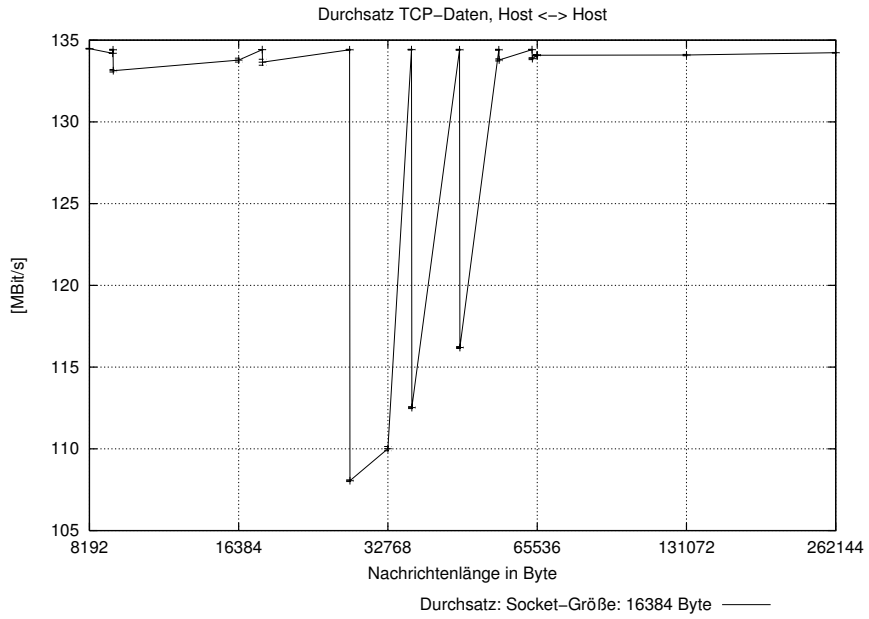


Abbildung B.3: Maximaler Durchsatz über TCP (3)

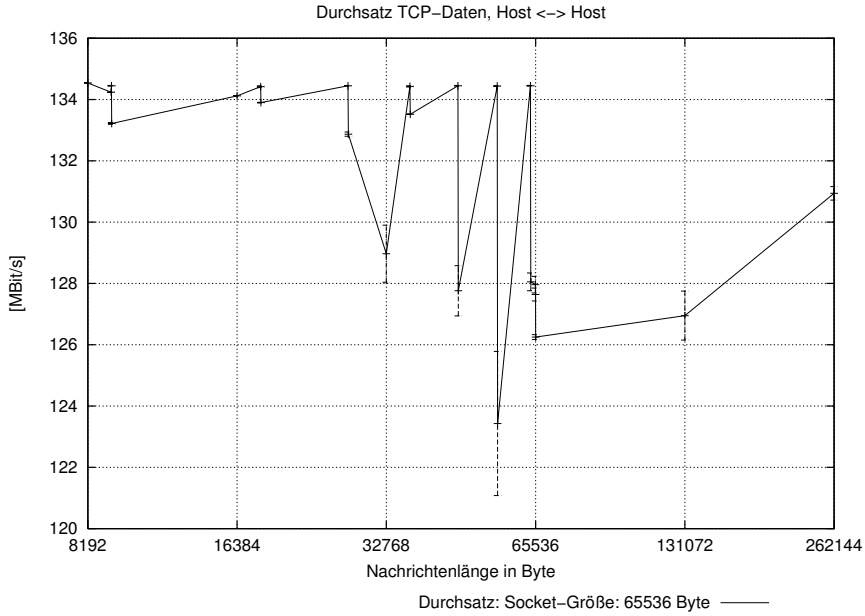


Abbildung B.4: Maximaler Durchsatz über TCP (4)

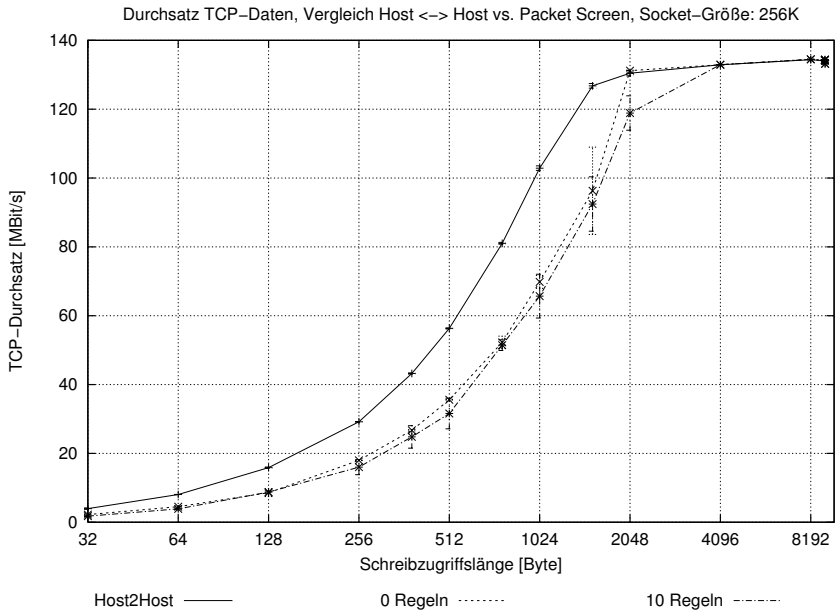


Abbildung B.5: Durchsatzmessungen TCP über Packet Screen, Socket-Größe 256 KByte (1)

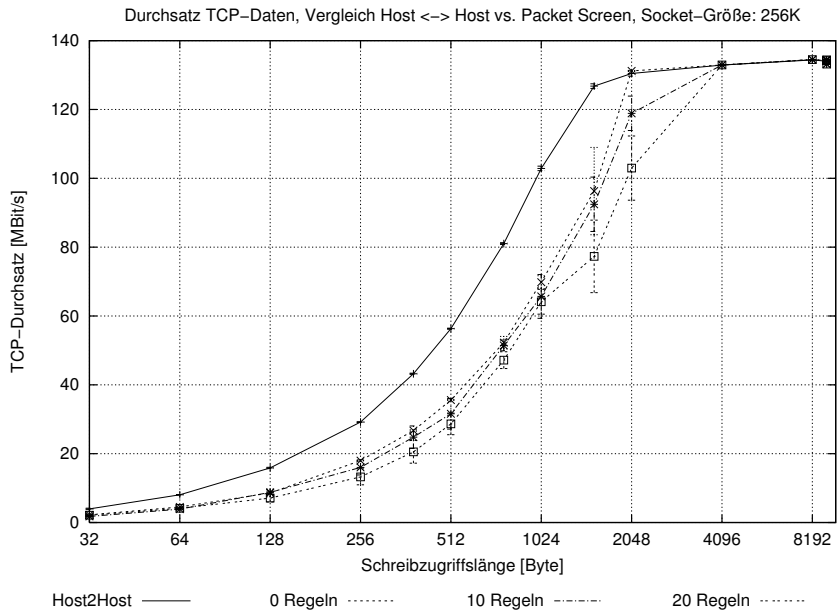


Abbildung B.6: Durchsatzmessungen TCP über Packet Screen, Socket-Größe 256 KByte (2)

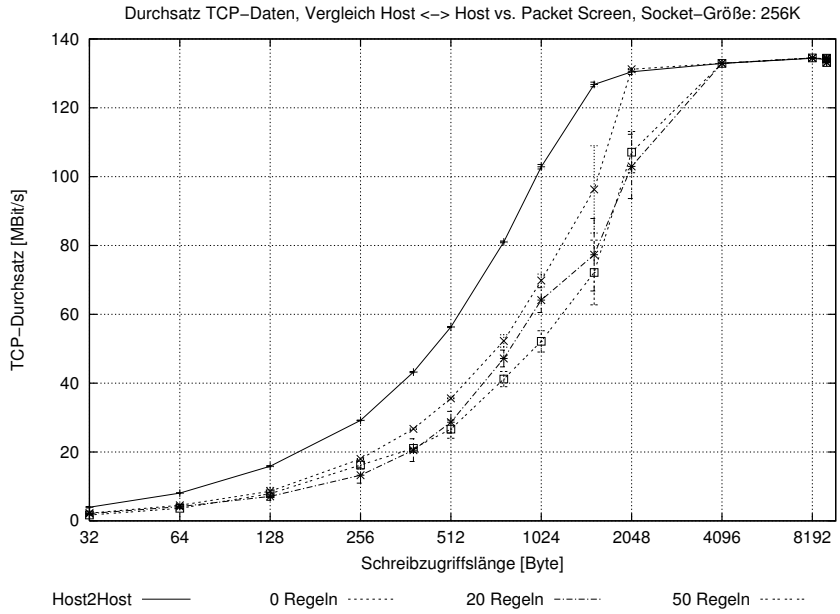


Abbildung B.7: Durchsatzmessungen TCP über Packet Screen, Socket-Größe 256 KByte (3)

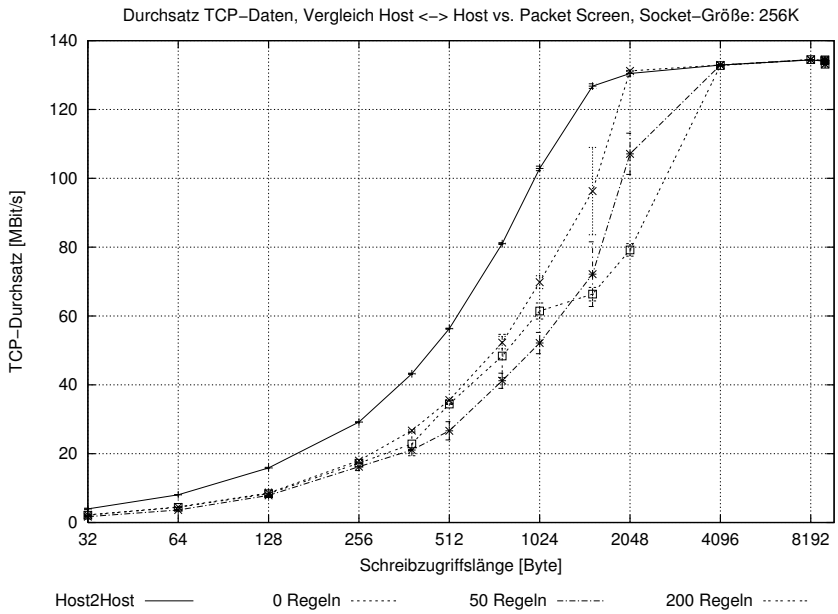


Abbildung B.8: Durchsatzmessungen TCP über Packet Screen, Socket-Größe 256 KByte (4)

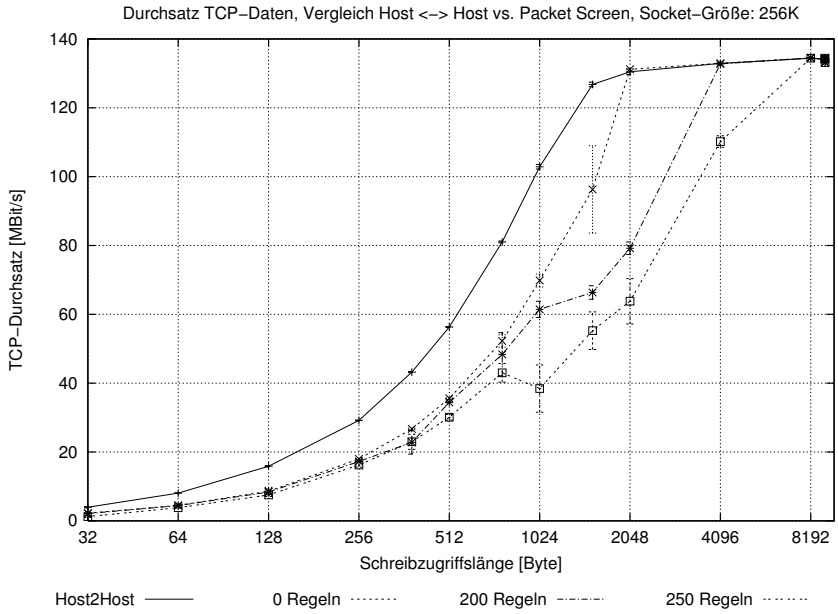


Abbildung B.9: Durchsatzmessungen TCP über Packet Screen, Socket-Größe 256 KByte (5)

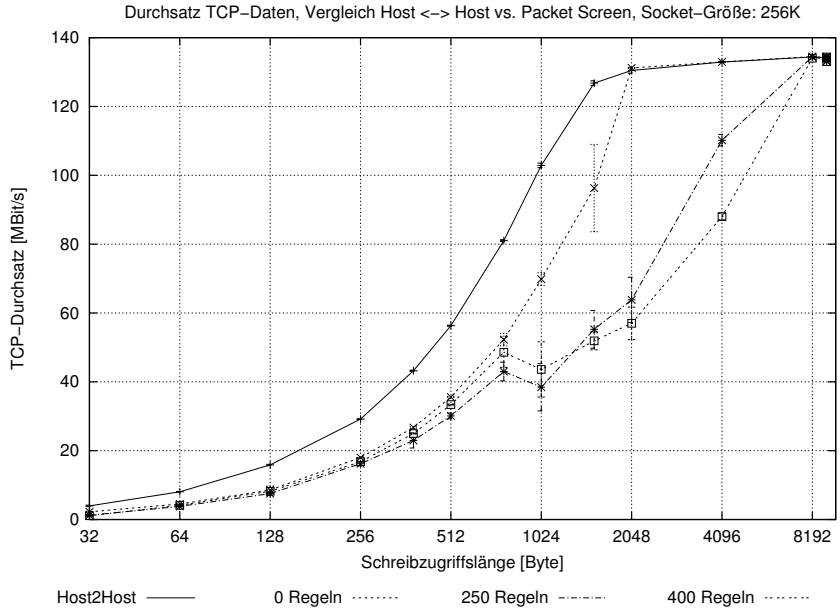


Abbildung B.10: Durchsatzmessungen TCP über Packet Screen, Socket-Größe 256 KByte (6)

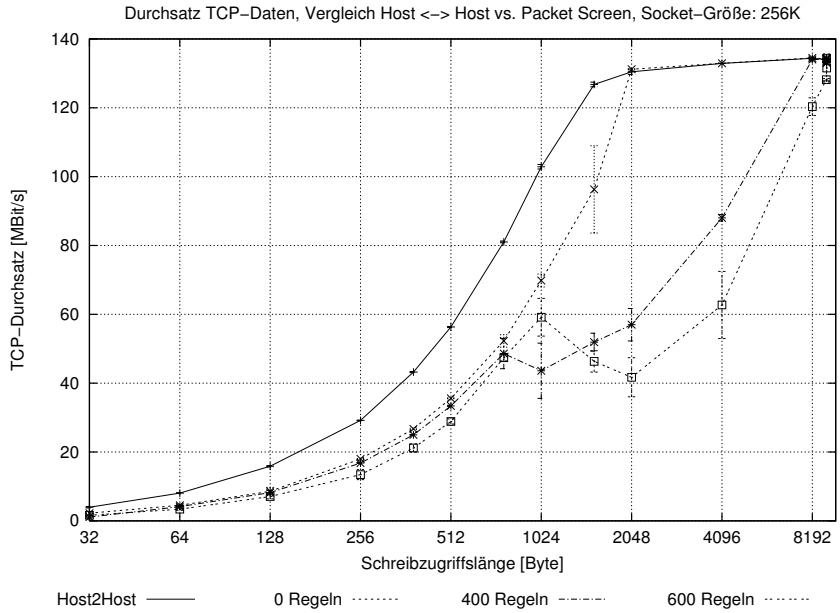


Abbildung B.11: Durchsatzmessungen TCP über Packet Screen, Socket-Größe 256 KByte (7)

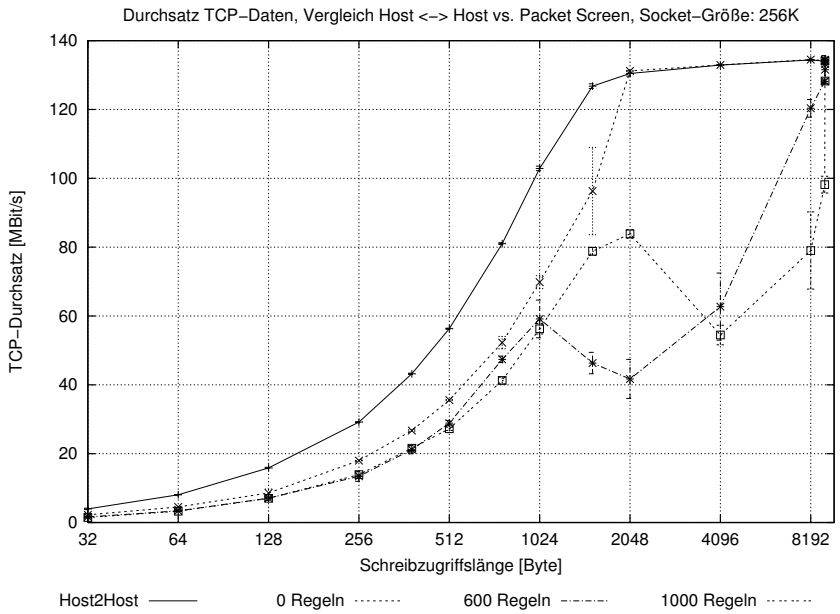


Abbildung B.12: Durchsatzmessungen TCP über Packet Screen, Socket-Größe 256 KByte (8)

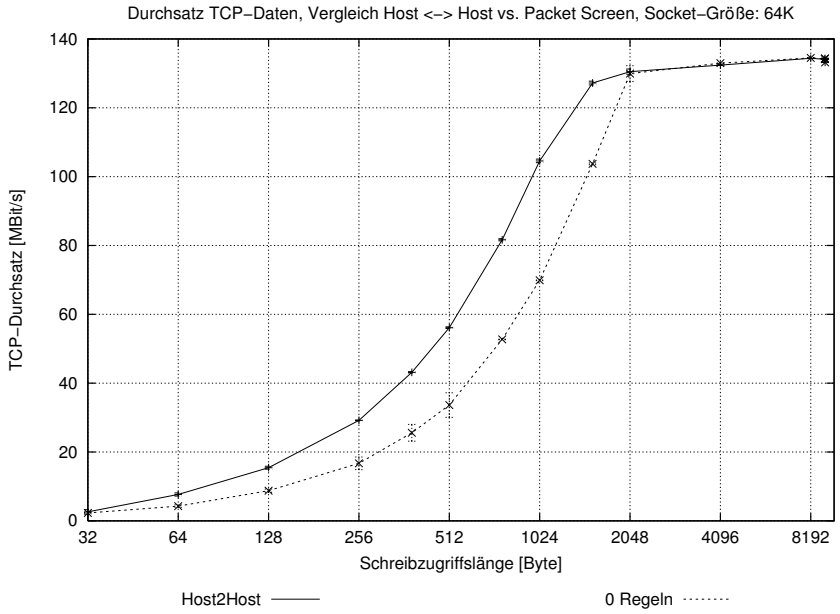


Abbildung B.13: Durchsatzmessungen TCP über Packet Screen, Socket-Größe 64 KByte (1)

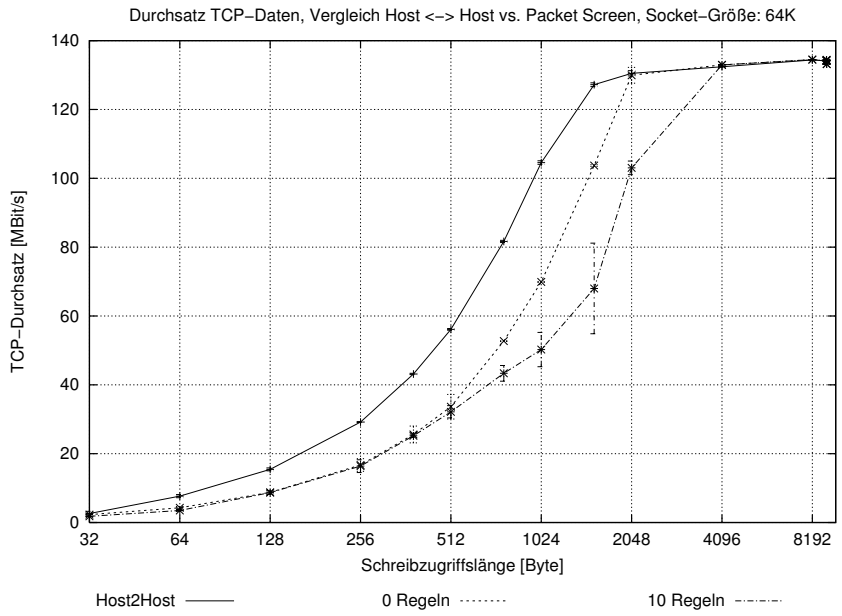


Abbildung B.14: Durchsatzmessungen TCP über Packet Screen, Socket-Größe 64 KByte (2)

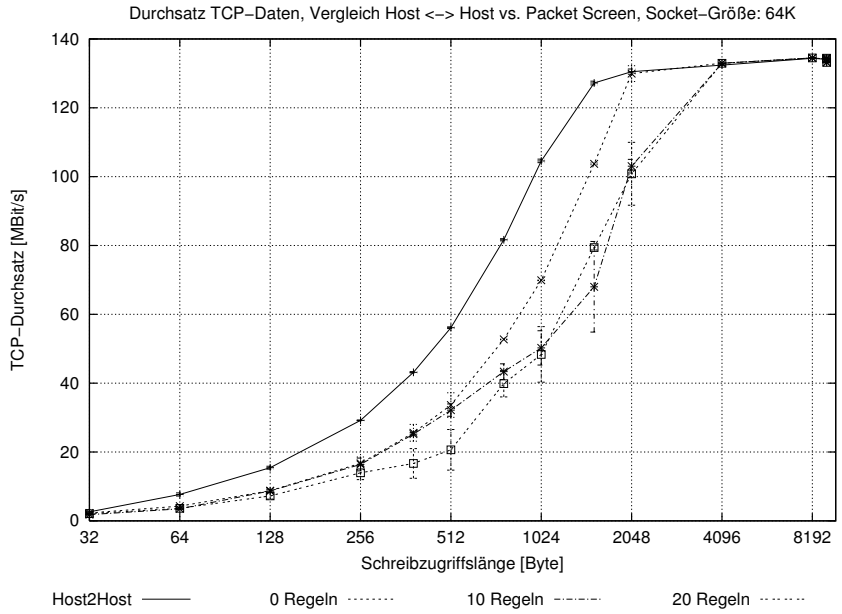


Abbildung B.15: Durchsatzmessungen TCP über Packet Screen, Socket-Größe 64 KByte (3)

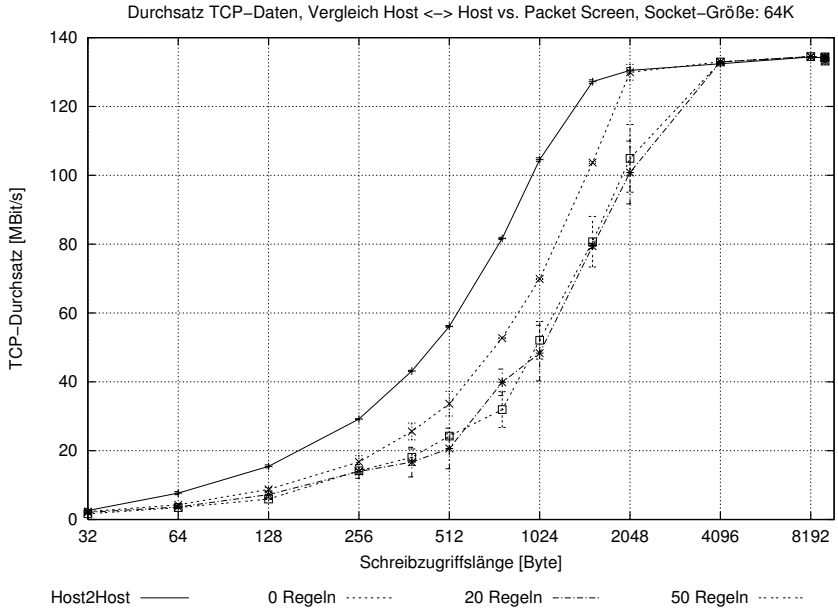


Abbildung B.16: Durchsatzmessungen TCP über Packet Screen, Socket-Größe 64 KByte (4)

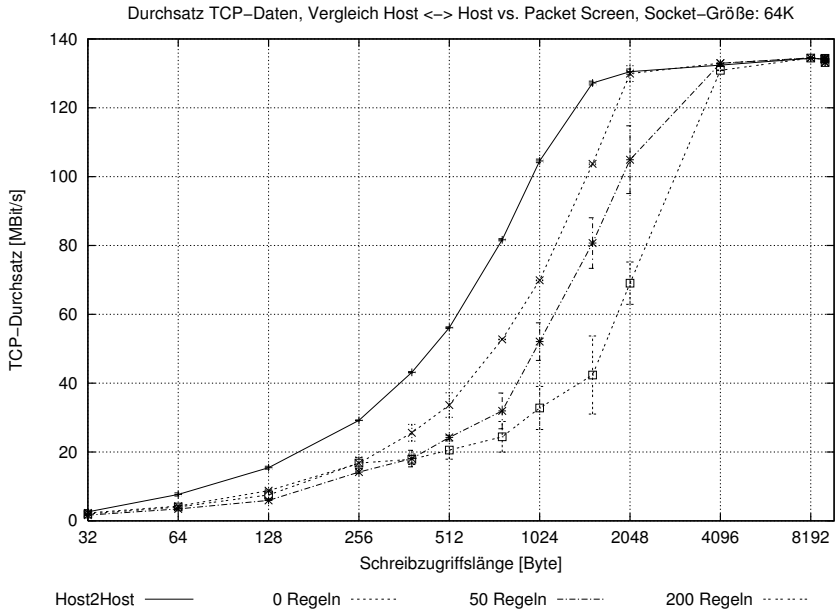


Abbildung B.17: Durchsatzmessungen TCP über Packet Screen, Socket-Größe 64 KByte (5)

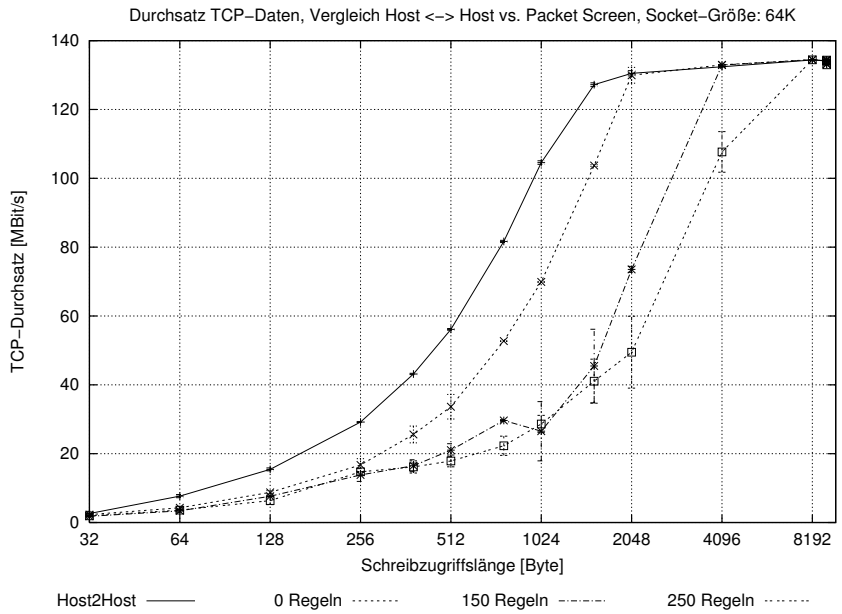


Abbildung B.18: Durchsatzmessungen TCP über Packet Screen, Socket-Größe 64 KByte (6)

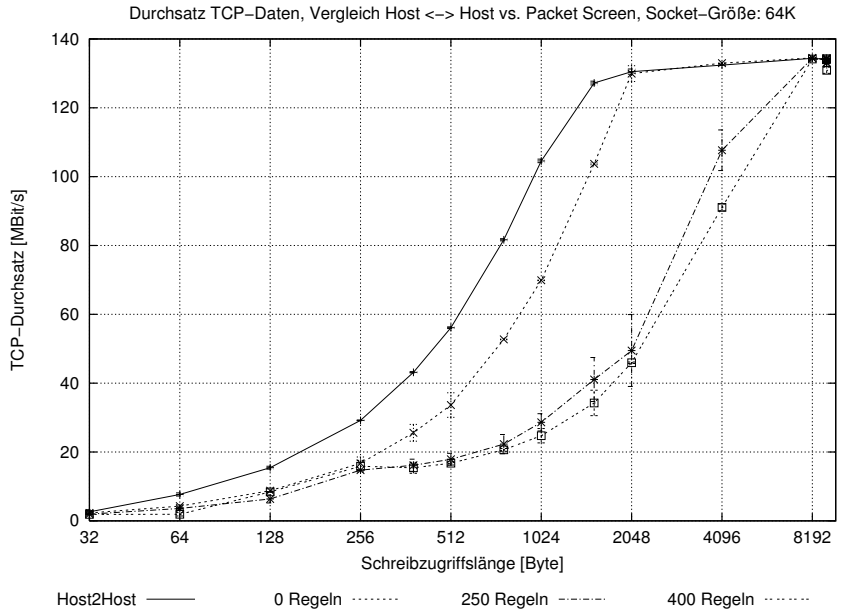


Abbildung B.19: Durchsatzmessungen TCP über Packet Screen, Socket-Größe 64 KByte (7)

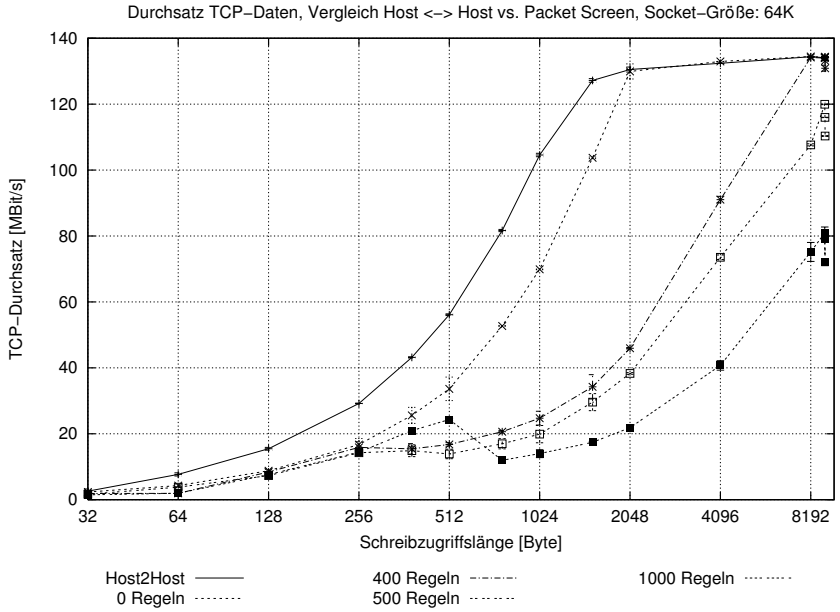


Abbildung B.20: Durchsatzmessungen TCP über Packet Screen, Socket-Größe 64 KByte (8)

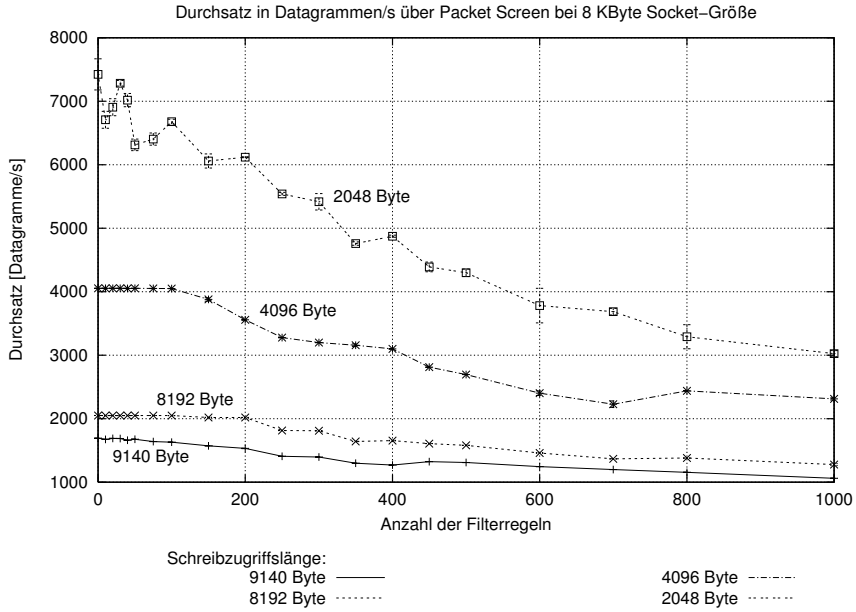


Abbildung B.21: Datagrammdurchsatz abhängig von der Schreibzugriffslänge (1)

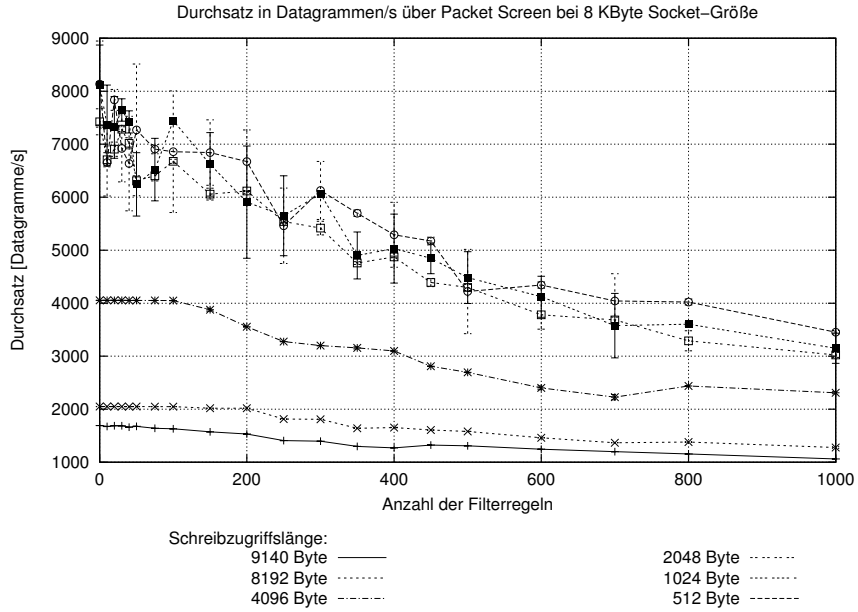


Abbildung B.22: Datagrammdurchsatz abhängig von der Schreibzugriffslänge (2)

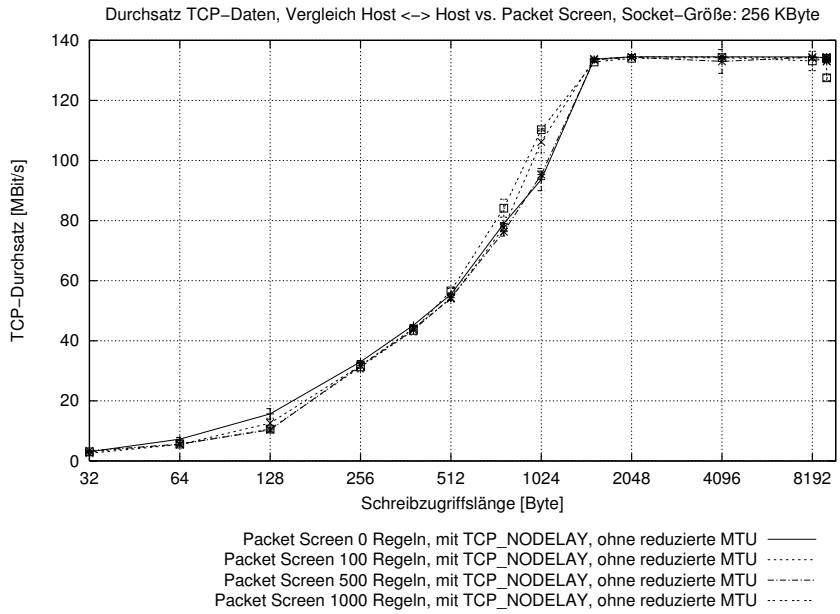


Abbildung B.23: Durchsatzmessungen TCP über Packet Screen ohne Reduzierung der MTU-Größe (1)

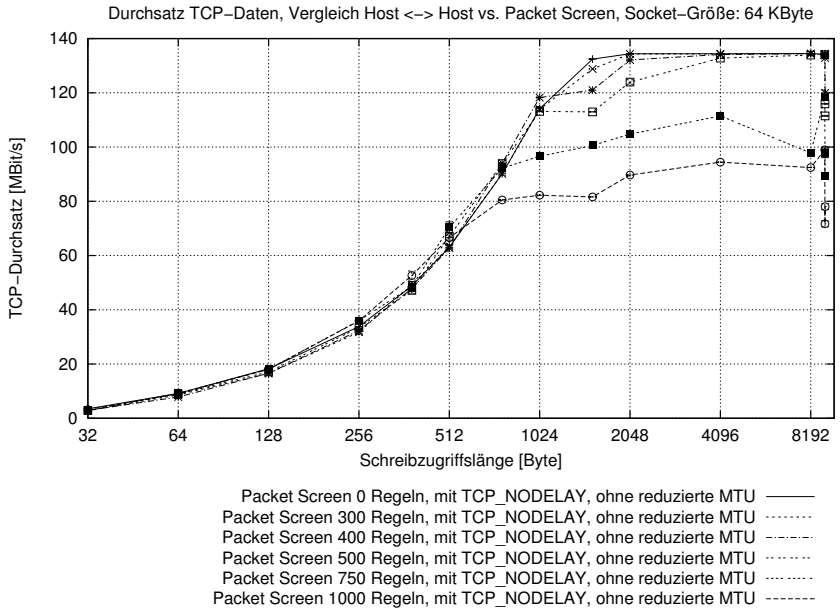


Abbildung B.24: Durchsatzmessungen TCP über Packet Screen ohne Reduzierung der MTU-Größe (2)

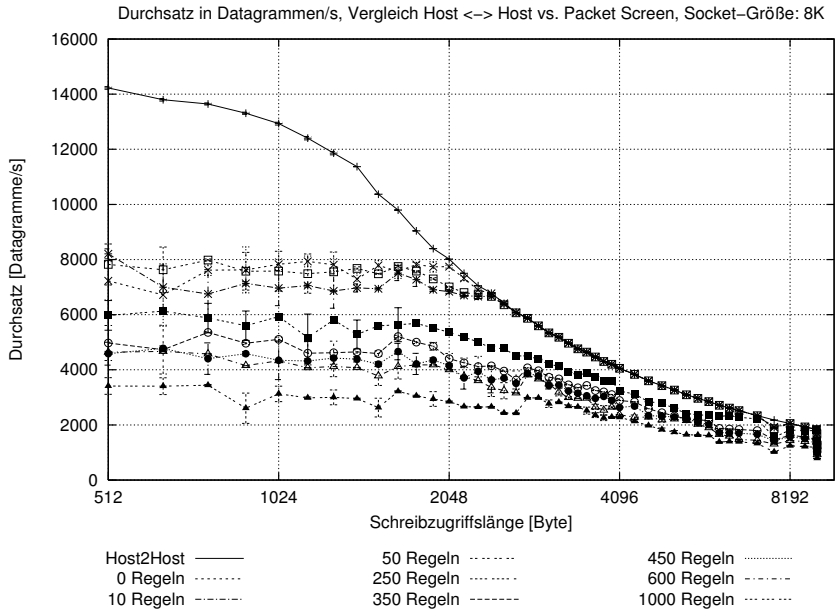


Abbildung B.25: Datagrammdurchsatz über Packet Screen

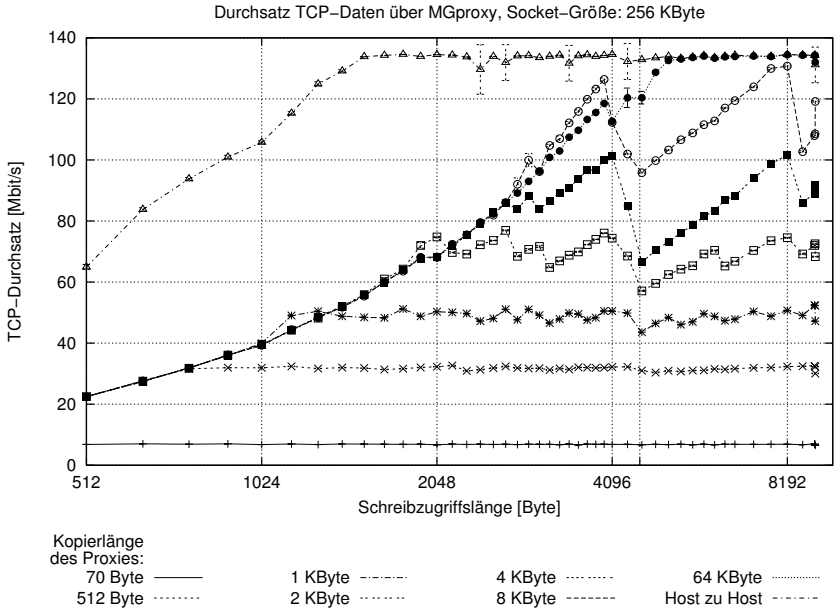


Abbildung B.26: Durchsatz über "Minimalen Generischen Proxy" ("MGproxy")

C Ergänzende Tabellen und Daten

C.1 Durchschnittliche Paket- und Nachrichtenlängen

Die beiden folgenden Tabellen (C.1 und C.2) zu den durchschnittlichen Paket- und Nachrichtenlängen wurden dem dritten Quartalsbericht des DFN-FWL [Benecke et al. 97b, S. 13-14] entnommen:

Proto-koll	Source Pkt.-Size [Byte]	(Std.)	Dest. Pkt.-Size [Byte]	(Std.)
HTTP	48	(29)	316	(301)
SMTP	217	(177)	29	(10)
FTP	6	(7)	81	(46)
FTP-Data	0	(10)	581	(443)
SSH	62	(214)	137	(266)
DNS	49	(6)	205	(74)
AUTH	1	(0)	7	(2)

Tabelle C.1: Verkehrsstatistik: Paketlängen

Proto-koll	Source Verb.-Länge [Byte]	(Std.)	Dest. Verb.-Länge [Byte]	(Std.)
HTTP	332	(389)	6698	(140547)
SMTP	6407	(35759)	415	(146)
FTP	260	(3247)	1567	(5443)
FTP-Data	97	(8150)	224009	(1682687)
SSH	39637	(433306)	289618	(2779742)
DNS	64	(201)	235	(2020)
AUTH	10	(0)	37	(11)

Tabelle C.2: Verkehrsstatistik: pro Verbindung übertragene Byte

C.2 Packet Screen Konfigurationen

Die im folgenden wiedergegebene Filterkonfiguration mit 100 Regeln beginnt mit 23 Filterregeln, die "IP-Spoofing"-Angriffe abwehren sollen. Anschließend wird das "Default"-Verhalten auf Blockieren gesetzt und ein administrativer Zugang für die Systeme "`wind`" und "`wave`" konfiguriert. Um die gewünschte Anzahl von Filterregeln zu erreichen, werden Blöcke mit je zehn Zeilen eingefügt, von denen sich jeweils ein Block immer auf eine an der Messung nicht beteiligte IP-Adresse bezieht. Für die weiteren Messungen müssen nur jeweils Filterregeln für andere im Testnetz nicht vorhandene IP-Adressen hinzugefügt werden. Abschließend muß der Zugriff über SSH (Port 22) für Konfigurationsaufgaben geöffnet werden und die Erreichbarkeit des Meßwerkzeuges (Port 5000) gewährleistet sein. Der Port 5000 dient dabei nur für Steuernachrichten (Öffnen und Schließen von Verbindungen, Übertragung von Meßergebnissen). Für die eigentliche Messung trifft die letzte (hier 101.) Regel zu. Kommentare beginnen mit einem "`#`" und werden ignoriert.

```
# When checking a packet, the packet is compared to the entire list from
# top to bottom, the last matching line being effective.
#
# It currently defaults to pass on all packets.
#
#
block in log quick from any to any with ipopts
block in log quick proto tcp from any to any with short
block in on le0 from 127.0.0.0/8 to any
block out on le0 from 127.0.0.0/8 to any
block out on le0 from any to 127.0.0.0/8
block in on le0 from 192.168.16.76/32 to any
```

```
block out on le0 from any to 192.168.16.76/32
block in on le0 from 192.168.14.76/0xffffff00 to any
block in on le0 from 192.168.15.76/0xffffff00 to any
block in on ba0 from 127.0.0.0/8 to any
block out on ba0 from 127.0.0.0/8 to any
block out on ba0 from any to 127.0.0.0/8
block in on ba0 from 192.168.14.76/32 to any
block out on ba0 from any to 192.168.14.76/32
block in on ba0 from 192.168.16.76/0xffffff00 to any
block in on ba0 from 192.168.15.76/0xffffff00 to any
block in on ba1 from 127.0.0.0/8 to any
block out on ba1 from 127.0.0.0/8 to any
block out on ba1 from any to 127.0.0.0/8
block in on ba1 from 192.168.15.76/32 to any
block out on ba1 from any to 192.168.15.76/32
block in on ba1 from 192.168.16.76/0xffffff00 to any
block in on ba1 from 192.168.14.76/0xffffff00 to any
# set default to block
block in log from any to any
#
# pass wind + wave
pass in on le0 from wind to any
pass in on le0 from wave to any
#
# Testnetzregeln:
# fuell blocks unnoetig (10er)
block in proto tcp from 134.100.14.1 to any port = echo
block in proto tcp from 134.100.14.1 to any port = telnet
block in proto tcp from 134.100.14.1 to any port = 25
block in proto tcp from 134.100.14.1 to any port = finger
block in proto tcp from 134.100.14.1 to any port = ftp
block in proto tcp from 134.100.14.1 to any port = 1
block in proto tcp from 134.100.14.1 to any port = 2
block in proto tcp from 134.100.14.1 to any port = 3
block in proto tcp from 134.100.14.1 to any port = 4
block in proto tcp from 134.100.14.1 to any port = 5
# fuell blocks unnoetig (10er)
block in proto tcp from 134.100.14.2 to any port = echo
block in proto tcp from 134.100.14.2 to any port = telnet
block in proto tcp from 134.100.14.2 to any port = 25
block in proto tcp from 134.100.14.2 to any port = finger
block in proto tcp from 134.100.14.2 to any port = ftp
block in proto tcp from 134.100.14.2 to any port = 1
block in proto tcp from 134.100.14.2 to any port = 2
block in proto tcp from 134.100.14.2 to any port = 3
block in proto tcp from 134.100.14.2 to any port = 4
block in proto tcp from 134.100.14.2 to any port = 5
# fuell blocks unnoetig (10er)
block in proto tcp from 134.100.14.3 to any port = echo
block in proto tcp from 134.100.14.3 to any port = telnet
block in proto tcp from 134.100.14.3 to any port = 25
```

```
block in proto tcp from 134.100.14.3 to any port = finger
block in proto tcp from 134.100.14.3 to any port = ftp
block in proto tcp from 134.100.14.3 to any port = 1
block in proto tcp from 134.100.14.3 to any port = 2
block in proto tcp from 134.100.14.3 to any port = 3
block in proto tcp from 134.100.14.3 to any port = 4
block in proto tcp from 134.100.14.3 to any port = 5
# fuell blocks unnoetig (10er)
block in proto tcp from 134.100.14.4 to any port = echo
block in proto tcp from 134.100.14.4 to any port = telnet
block in proto tcp from 134.100.14.4 to any port = 25
block in proto tcp from 134.100.14.4 to any port = finger
block in proto tcp from 134.100.14.4 to any port = ftp
block in proto tcp from 134.100.14.4 to any port = 1
block in proto tcp from 134.100.14.4 to any port = 2
block in proto tcp from 134.100.14.4 to any port = 3
block in proto tcp from 134.100.14.4 to any port = 4
block in proto tcp from 134.100.14.4 to any port = 5
# fuell blocks unnoetig (10er)
block in proto tcp from 134.100.14.5 to any port = echo
block in proto tcp from 134.100.14.5 to any port = telnet
block in proto tcp from 134.100.14.5 to any port = 25
block in proto tcp from 134.100.14.5 to any port = finger
block in proto tcp from 134.100.14.5 to any port = ftp
block in proto tcp from 134.100.14.5 to any port = 1
block in proto tcp from 134.100.14.5 to any port = 2
block in proto tcp from 134.100.14.5 to any port = 3
block in proto tcp from 134.100.14.5 to any port = 4
block in proto tcp from 134.100.14.5 to any port = 5
# fuell blocks unnoetig (10er)
block in proto tcp from 134.100.14.6 to any port = echo
block in proto tcp from 134.100.14.6 to any port = telnet
block in proto tcp from 134.100.14.6 to any port = 25
block in proto tcp from 134.100.14.6 to any port = finger
block in proto tcp from 134.100.14.6 to any port = ftp
block in proto tcp from 134.100.14.6 to any port = 1
block in proto tcp from 134.100.14.6 to any port = 2
block in proto tcp from 134.100.14.6 to any port = 3
block in proto tcp from 134.100.14.6 to any port = 4
block in proto tcp from 134.100.14.6 to any port = 5
# fuell blocks unnoetig (10er)
block in proto tcp from 134.100.14.7 to any port = echo
block in proto tcp from 134.100.14.7 to any port = telnet
block in proto tcp from 134.100.14.7 to any port = 25
block in proto tcp from 134.100.14.7 to any port = finger
block in proto tcp from 134.100.14.7 to any port = ftp
block in proto tcp from 134.100.14.7 to any port = 1
block in proto tcp from 134.100.14.7 to any port = 2
block in proto tcp from 134.100.14.7 to any port = 3
block in proto tcp from 134.100.14.7 to any port = 4
block in proto tcp from 134.100.14.7 to any port = 5
```

```
## fuell blocks unnoetig (10er)
#block in proto tcp from 134.100.14.8 to any port = echo
#block in proto tcp from 134.100.14.8 to any port = telnet
#block in proto tcp from 134.100.14.8 to any port = 25
#block in proto tcp from 134.100.14.8 to any port = finger
#block in proto tcp from 134.100.14.8 to any port = ftp
#block in proto tcp from 134.100.14.8 to any port = 1
#block in proto tcp from 134.100.14.8 to any port = 2
#block in proto tcp from 134.100.14.8 to any port = 3
#block in proto tcp from 134.100.14.8 to any port = 4
#block in proto tcp from 134.100.14.8 to any port = 5
#
# Fuer Konfig+Login notwendig
pass in proto tcp from any to any port = 22
pass in proto tcp from any port = 22 to any
#
# Fuer Tests notwendig
pass in proto tcp from any to any port = 5000
pass in proto tcp from any port = 5000 to any
pass in proto tcp from any port > 32764 to any port > 32764
```

Der wiedergegebene Filterregelsatz wurde für Durchsatzmessungen (Abschnitt 4.3.1) eingesetzt, bei denen ausschließlich Portnummern im Bereich oberhalb von 32764 verwendet werden. Für die Nachrichtenlaufzeit- (Abschnitt 4.3.4) und Transaktionsmessungen (Abschnitt 4.3.5) wird der gesamte Portbereich benötigt, so daß die letzte Filterregel für Nachrichtenlaufzeit- und Transaktionsmessungen in

```
pass in proto tcp from any to any
```

geändert werden muß.

D Abkürzungsverzeichnis

AAL	**A**TM **A**daptation **L**ayer
ABR	**A**vailable **B**it **R**ate
ACK	**ACK**nowledgment
ARP	**A**ddress **R**esolution **P**rotocol
ASN.1	**A**bstract **S**yntax **N**otation **1**
ATM	**A**synchronous **T**ransfer **M**ode
ATMARP	**A**synchronous **T**ransfer **M**ode **A**ddress **R**esolution **P**rotocol
BDP	**B**andwidth x **D**elay **P**roduct
BHLI	**B**roadband **H**igh **L**ayer **I**nformation
BLLI	**B**roadband **L**ow **L**ayer **I**nformation
BSD	**B**erkeley **S**oftware **D**istribution
BUS	**B**roadcast and **U**nknown **S**erver
CAM	**C**ontent **A**ddressable **M**emory
CBR	**C**onstant **B**it **R**ate
CDV	**C**ell **D**elay **V**ariation
CERT	**C**omputer **E**mergency **R**esponse **T**eam
CLIP	**CL**assical **IP** and ARP over ATM
CLR	**C**ell **L**oss **R**atio
CPU	**C**entral **P**rocessing **U**nit
CTD	**C**ell **T**ransfer **D**elay
DES	**D**ata **E**ncrytion **S**tandard
DFN	**D**eutsches **F**orschungs**n**etz
DNS	**D**omain **N**ame **S**ystem
DoD	(US) **D**epartment **o**f **D**efense
DSM-CC	**D**igital **S**torage **M**edia – **C**ommand and **C**ontrol
DTL	**D**esignated **T**ransit **L**ist
ESP	**E**ncapsulated **S**ecurity **P**ayload
FDDI	**F**iber **D**istribution **D**ata **I**nterface
FTP	**F**ile **T**ransfer **P**rotocol
FWL	**F**ire**w**all **L**abor
FWTK	**F**ire**w**all **T**ool**K**it
GFC	**G**eneric **F**low **C**ontrol
HSN	**H**igh-**S**peed **N**etwork

HTML	**H**yper**t**ext **M**arkup **L**anguage
HTTP	**H**yper**t**ext **T**ransfer **P**rotocol
IBM	**I**nternational **B**usiness **M**achines
ICMP	**I**nternet **C**ontrol **M**essage **P**rotocol
IDEA	**I**nternational **D**ata **E**ncryption **A**lgorithm
IE	**I**nformation **E**lement
IETF	**I**nternet **E**ngineering **T**ask **F**orce
IISP	**I**nterim **I**nter-Switch **S**ignaling **P**rotocol
ILMI	**I**ntegrated **L**ocal **M**anagement **I**nterface
IOS	**I**nternetwork **O**perating **S**ystem
IP	**I**nternet **P**rotocol
IPSEC	**I**nternet **P**rotocol **SEC**urity Protocol Working Group
ISAKMP	**I**nternet **S**ecurity **A**ssociation **K**ey **M**anagement **P**rotocol
ISDN	**I**ntegrated **S**ervices **D**igital **N**etwork
ISO	**I**nternational **S**tandards **O**rganization
ITU	**I**nternational **T**elecommunication **U**nion
LAN	**L**ocal **A**rea **N**etwork
LANE	**LAN E**mulation
LCH	**L**ast **C**ell **H**ostage
LEC	**L**AN **E**mulation **C**lient
LECS	**L**AN **E**mulation **C**onfiguration **S**erver
LES	**L**AN **E**mulation **S**erver
LIJ	**L**eaf **I**nitiated **J**oin
LIS	**L**ogical **I**P **S**ubnet
LUNI	**L**ANE **U**ser **N**etwork **I**nterface
MAN	**M**etropolitan **A**rea **N**etwork
MARS	**M**ulticast **A**ddress **R**esolution **S**erver
MCTD	**M**aximum **C**ell **T**ransfer **D**elay
MD5	**M**essage **D**igest **#5**
MPLS	**M**ulti-**P**rotocol **L**abel **S**witching
MPOA	**M**ulti-**P**rotocol **o**ver **A**TM
MPS	**MP**OA **S**erver
MS-DOS	**M**icro**S**oft **D**isk **O**perating **S**ystem
MSS	**M**aximum **S**egment **S**ize
MTU	**M**aximum **T**ransfer **U**nit
NAT	**N**etwork **A**ddress **T**ranslation
NBMA	**N**on-**B**roadcast, **M**ultiple-**A**ccess
NFS	**N**etwork **F**ile **S**ystem
NHC	**N**ext **H**op **C**lient
NHRP	**N**ext **H**op **R**esolution **P**rotocol
NHS	**N**ext **H**op **S**erver
NNTP	**N**etwork **N**ews **T**ransfer **P**rotocol
OC	**O**ptical **C**arrier

OSI	**O**pen **S**ystems **I**nterconnection
OSPF	**O**pen **S**hortest **P**ath **F**irst
PCR	**P**eak **C**ell **R**ate
PDU	**P**rotocol **D**ata **U**nit
PGL	**P**eer **G**roup **L**eader
PGP	**P**retty **G**ood **P**rivacy
PNNI	**P**rivate **N**etwork-**N**etwork **I**nterface
PVC	**P**ermanent **V**irtual **C**ircuit
QoS	**Q**uality **o**f **S**ervice
RC4	**R**ivest **C**ipher No. **4**
RFC	**R**equest **F**or **C**omments
ROLC	**R**outing **O**ver **L**arge **C**louds
RPC	**R**emote **P**rocedure **C**all
RSA	**R**ivest, **S**hamir, **A**dleman
RST	**R**ese**t**
RSVP	**R**esource Re**S**er**V**ation **P**rotocol
RTT	**R**ound-**T**rip **T**ime
SCR	**S**ustainable **C**ell **R**ate
SCSP	**S**erver **C**ache **S**ynchronization **P**rotocol
SDU	**S**ervice **D**ata **U**nit
SET	**S**ecure **E**lectronic **T**ransaction
SHA	**S**ecure **H**ash **A**lgorithm
SMTP	**S**imple **M**ail **T**ransfer **P**rotocol
SNAP	**S**ub-**N**etwork **A**ttachment **P**oint
SNMP	**S**imple **N**etwork **M**anagement **P**rotocol
SOA	**S**tart **O**f **A**uthority
SONET	**S**ynchronous **O**ptical **NET**work
SSH	**S**ecure **Sh**ell
SSL	**S**ecure **S**ocket **L**ayer
STS	**S**ynchronous **T**ransport **S**ignal
SVC	**S**witched **V**irtual **C**ircuit
SYN	**SYN**chronizing Segment
TCP	**T**ransmission **C**ontrol **P**rotocol
TIS	**T**rusted **I**nformation **S**ystems, Inc.
TLS	**T**ransport **L**ayer **S**ecurity
TNS	**T**ransit **N**etwork **S**election
TTL	**T**ime **T**o **L**ive
UBR	**U**nspecified **B**it **R**ate
UDP	**U**ser **D**atagram **P**rotocol
UNI	**U**ser-**N**etwork **I**nterface
UNIX	**UN**iplexed **I**nformation and **C**omputing **S**ystem
URL	**U**niform **R**esource **L**ocator
UTP	**U**nshielded **T**wisted-**P**air

VBR	**V**ariable **B**it **R**ate
VCI	**V**irtual **C**hannel **I**dentifier
VoD	**V**ideo **o**n **D**emand
VPI	**V**irtual **P**ath **I**dentifier
VPN	**V**irtual **P**rivate **N**etworks
WAN	**W**ide **A**rea **N**etwork
WWW	**W**orld **W**ide **W**eb

E. Abbildungsverzeichnis

2.1 "Sniffer"-Angriff 12
2.2 TCP-Verbindungsaufbau ("3-Way-Handshake") 14
2.3 Einsatz kryptographischer Verfahren in Protokollen 24
2.4 Von Packet Screens ausgewertete Informationen 30
2.5 Konzeptionelle Sicht auf einen Proxy 33
2.6 Packet Screen: konzeptionelle Sicht 37
2.7 Kombination von Packet Screen und Bastion: konzeptionelle Sicht – Bastion innen 40
2.8 Kombination von Packet Screen und Bastion: konzeptionelle Sicht – Bastion mittig 40
2.9 Kombination von Packet Screen und Bastion: konzeptionelle Sicht – Bastion außen 41
2.10 Kombination von Packet Screen und Bastion: konzeptionelle Sicht – beidseitig abgeschirmte Bastion 42
2.11 Gateway-Firewall: konzeptionelle Sicht 42
2.12 Aufbau einer ATM-Zelle 48
2.13 Virtuelle Verbindung 49
2.14 Signalisierung zum Auf- und Abbau einer virtuellen Verbindung . 50
2.15 Überblick über die Verkehrsklassen bei ATM 53
2.16 "Classical IP"-Protokollstapel 56

3.1 "Sniffer"-Angriff in einem ATM-Netz 82
3.2 ATM-Netz mit zwei PNNI-Hierarchieebenen 107
3.3 Alternative Positionen für Signalisierungsfilter im Switch B 118
3.4 PVCs für UNI-Signalisierung in einem Switch 119
3.5 Anschluß eines Prototyps eines Signalisierungsfilters 120
3.6 Sicherheitsdienstgüte-basierte Wegwahl durch "Einschleifen" von Sicherheitskomponenten 129
3.7 Sicherheitsdienstgüte-basierte Wegwahl bei hereinkommender Signalisierung 134

3.8 Sicherheitsdienstgüte-basierte Wegwahl bei herausgehender Signalisierung . . . 135
3.9 Aufteilung von IP-Dateneinheiten auf ATM-Zellen bei CLIP und NHRP . . . 147
3.10 Aufteilung von IP-Dateneinheiten auf ATM-Zellen bei LANE . . . 150
3.11 Aufteilung von IPv6-Dateneinheiten auf ATM-Zellen . . . 150
3.12 Packet Screen zwischen zwei Switches . . . 152
3.13 "Video on Demand" [ATM Forum 97a] . . . 161
3.14 Einbindung eines IP-basierten Proxies durch einen Signalisierungskontroller . . . 167
3.15 Datenfluß über einen IP-basierten Proxy in einem ATM-Netz . . . 169
3.16 Testnetz-Konfiguration für erstes Experiment mit transparenten Proxies . . . 175
3.17 Testnetz-Konfiguration für zweites Experiment mit transparenten Proxies . . . 175
3.18 Packet Screen in einem CLIP-Netz . . . 181
3.19 Beidseitig abgeschirmte Bastion für ein CLIP-Netz . . . 183
3.20 Kombination von Packet Screen mit Signalisierungsfilter . . . 183
3.21 Beidseitig abgeschirmte Bastion mit Packet Screens und Signalisierungsfilter . . . 186
3.22 Kombination von Bastion und Signalisierungsfilter . . . 188
3.23 Aufteilung eines Netzes in virtuelle Netze . . . 193
3.24 Kontrolle virtueller Netze durch dezentrale Firewalls . . . 197
3.25 Kontrolle virtueller Netze durch einen zentralen Firewall . . . 198
3.26 Gateway-Firewall mit einer virtuellen Schnittstelle . . . 201
3.27 Trennung von "Native"-ATM und IP durch virtuelle Netze . . . 203
3.28 Ein Firewall für "Native"-ATM und IP . . . 205

4.1 Das Testnetz: flexible Infrastruktur . . . 216
4.2 Das Testnetz: zweite Ausbaustufe . . . 217
4.3 Maximaler Datendurchsatz über TCP . . . 219
4.4 Durchsatz mit reduzierter MTU-Größe und `TCP_NODELAY` . . . 222
4.5 Maximaler Durchsatz über TCP . . . 224
4.6 Struktur eines SONET STS-1 Rahmens . . . 227
4.7 Theoretisch erreichbarer Durchsatz . . . 231
4.8 Theoretisch erreichbarer Durchsatz (Ausschnitt) . . . 235
4.9 Effekt von Segmentierung und Fragmentierung auf den Durchsatz 235
4.10 Vergleich mit theoretisch erreichbarem Durchsatz . . . 237
4.11 Vergleich mit theoretisch erreichbarem Durchsatz (Ausschnitt) . . . 238

4.12 Datagrammdurchsatz 239
4.13 Datagrammdurchsatz bei unterschiedlichen Netzen 240
4.14 Durchsatz bei leistungsfähigeren Workstations 242
4.15 Datagrammdurchsatz bei "Ultra-2/300" 243
4.16 Durchsatz über eine 622 MBit/s ATM-Schnittstelle 244
4.17 Vergleich mit theoretisch erreichbarem Durchsatz (Ausschnitt) . . 245
4.18 Maximaler Durchsatz auf Verbindungen an "Localhost" 246
4.19 Maximaler Durchsatz über UDP 247
4.20 Umlaufzeitmessung 249
4.21 Transaktionsmessung 252
4.22 Transaktionsmessung über "Cross-Over"-Kabel 254
4.23 Transaktionsmessung über unterschiedliche Netze 255
4.24 Anteil des Verbindungsauf- und -abbaus an der Übertragung . . . 258
4.25 Synchronisation paralleler Anwendungsmessungen 262
4.26 Vergleich von Einzel- zu Gesamttransaktionsdurchsatz 266
4.27 Transaktionen pro Thread während der Meßphase 267
4.28 Aufsummierter Datendurchsatz über alle Threads 268
4.29 Transaktionsdurchsatz im Vergleich zum Datendurchsatz 271
4.30 Durchschnittliche Anzahl der Transaktionen pro Thread 272
4.31 Transaktionsdurchsatz über SSL 274
4.32 CPU-Last auf dem Meß-Server bei Transaktionsdurchsatzmessungen über SSL 277
4.33 Gesamtdatendurchsatz bei längeren Transaktionen 278
4.34 Gesamttransaktionsdurchsatz bei 1, 2 und 10 Threads 280
4.35 Datendurchsatz bei längeren Transaktionen 281
4.36 Auswirkungen von TCP-Parametern auf den Datendurchsatz: "Deferred-Acknowledge-Intervall" 284
4.37 Auswirkungen von TCP-Parametern auf den Datendurchsatz: "Deferred Acknowledge" 285
4.38 Auswirkungen von TCP-Parametern: "Deferred Acknowledge" bei 256 KByte Socket-Größe 286
4.39 Auswirkungen von TCP-Parametern: Socket-Größe 287
4.40 Testnetz-Konfiguration für Packet Screen Messungen 288
4.41 Durchsatzmessungen TCP über Packet Screen ohne Filterregeln . 290
4.42 Durchsatzmessungen TCP über Packet Screen ohne Filterregeln mit zusätzlichen Meßpunkten 292
4.43 Durchsatzmessungen TCP über Packet Screen mit 600 Filterregeln mit zusätzlichen Meßpunkten 293

4.44 Durchsatzmessungen TCP über Packet Screen, Socket-Größe 256 KByte 294
4.45 Durchsatzmessungen TCP über Packet Screen, Socket-Größe 256 KByte (Fortsetzung) 295
4.46 Durchsatzmessungen TCP über Packet Screen, Socket-Größe 64 KByte 296
4.47 Durchsatzmessungen TCP über Packet Screen, Socket-Größe 64 KByte (Fortsetzung) 297
4.48 Datagrammdurchsatz über Packet Screen 298
4.49 Datagrammdurchsatz über Packet Screen 299
4.50 Datagrammdurchsatz abhängig von der Schreibzugriffslänge . . . 299
4.51 Durchsatzmessungen TCP über Packet Screen ohne Reduzierung der MTU-Größe 301
4.52 Durchsatzmessungen TCP über Packet Screen ohne Reduzierung der MTU-Größe 302
4.53 Nachrichtenumläufe über Packet Screen 304
4.54 Nachrichtenlaufzeit über Packet Screen 306
4.55 Nachrichtenumläufe vs. Transaktionen über Packet Screen 307
4.56 Vergleich des theoretischen Datendurchsatzes mit gemessenen Durchsätzen über eine Packet Screen 311
4.57 Testnetz-Konfiguration für Proxy-Server Einsatz bei einem Gateway-Firewall (Gateway-Setup) 313
4.58 Testnetz-Konfiguration für Proxy-Server Einsatz auf einer Bastion (Bastion-Setup) 314
4.59 Datendurchsatz über Proxy-Server “Socks” im Vergleich zu einer direkten Host-zu-Host-Verbindung 316
4.60 Transaktionsmessung über Proxy-Server 319
4.61 Durchsatz über “Minimalen Generischen Proxy” (“MGproxy”) . . 324
4.62 Durchsatz über “MGproxy” (Ausschnitt) 326
4.63 Durchsatz bei “MGproxy”: Kopierlänge überschreitet Socket-Größe des Meßwerkzeuges 327
4.64 Durchsatz über “MGproxy”: Vergleich zwischen Bastion und Gateway Setup 329
4.65 Gesamtdatendurchsatz über “MGproxy” 334
4.66 Anzahl von Transaktionen während der Meßphase 336
4.67 Einzel- und Gesamttransaktionsdurchsatz über “MGproxy” 337
4.68 Transaktionsdurchsatz über “MGproxy” 338
4.69 Transaktionsdurchsatz: Verbindungsannahme-Alternativen 340

4.70 Auswirkungen der Verbindungsannahme-Alternativen auf CPU-Last der Bastion 342
4.71 Gesamtdatendurchsatz über "MGproxy" bei längeren Transaktionen 343
4.72 Vergleich Datendurchsatz über Host-zu-Host und "MGproxy" . . 344
4.73 Vergleich Datendurchsatz über Host-zu-Host und "MGproxy" . . 345
4.74 Gesamtdatendurchsatz über "MGproxy" bei längeren Transaktionen 346
4.75 Gesamtdatendurchsatz und CPU-Last bei längeren Transaktionen 347
4.76 Transaktionsdurchsatz über "MGproxy" bei längeren Transaktionen 348
4.77 Datendurchsatz bei Verschlüsselung und Zugriffskontrolle 351
4.78 CPU-Last auf der Bastion durch "Scanner" oder Verschlüsselung . 352
4.79 Transaktionsdurchsatz von "MGproxy" beim Einsatz von SSL . . 355
4.80 Last auf der Bastion bei Transaktionsdurchsatzmessung über "MGproxy" mit SSL 356
4.81 Anteil des Verbindungsaufbaus an der Übertragung 359
4.82 Anteil der Verbindungsauf- und -abbauzeiten an der Gesamttransaktionszeit 360
4.83 Transaktionslängen bei unterschiedlichen Datendurchsätzen 361
4.84 Testnetz-Konfiguration für Messungen bei Kombinationen von Packet Screen und Bastion 363
4.85 Durchsatz über Kombination von Packet Screen und Bastion . . . 364
4.86 Durchsatz über Kombination von Packet Screen und Bastion im Vergleich zu Messungen über Packet Screen 365
4.87 Nachrichtenlaufzeit über Kombination von Packet Screen und Bastion 366
4.88 Transaktionsmessung über Kombination von Packet Screen und Bastion 367

5.1 Konfiguration für "Round-Robin"-DNS 378
5.2 Lastverteilung über "Network Address Translation" (NAT) 383
5.3 Lastverteilung über NAT auf parallele Bastionen 384
5.4 Lastverteilung über "TCP-Router" 387
5.5 Lastverteilung über "TCP-Router" auf parallele Bastionen 388
5.6 Lastverteilung über virtuelle Verbindungen auf parallele Bastionen 389
5.7 Lastverteilung mit NHRP 391
5.8 Lastverteilung über Signalisierungskontroller 394
5.9 Verbindungsaufbau über "Socks" 399
5.10 Umleiten eines Zugriffs zur Lastverteilung 400
5.11 Parallele Packet Screen für Ethernet 408

5.12 Gleichzeitiger Zugriff eines Clients auf verschiedene Server über dieselbe TCP-Verbindung zum Proxy 416

B.1 Maximaler Durchsatz über TCP (1) 456
B.2 Maximaler Durchsatz über TCP (2) 457
B.3 Maximaler Durchsatz über TCP (3) 457
B.4 Maximaler Durchsatz über TCP (4) 458
B.5 Durchsatzmessungen TCP über Packet Screen, Socket-Größe 256 KByte (1) 458
B.6 Durchsatzmessungen TCP über Packet Screen, Socket-Größe 256 KByte (2) 459
B.7 Durchsatzmessungen TCP über Packet Screen, Socket-Größe 256 KByte (3) 459
B.8 Durchsatzmessungen TCP über Packet Screen, Socket-Größe 256 KByte (4) 460
B.9 Durchsatzmessungen TCP über Packet Screen, Socket-Größe 256 KByte (5) 460
B.10 Durchsatzmessungen TCP über Packet Screen, Socket-Größe 256 KByte (6) 461
B.11 Durchsatzmessungen TCP über Packet Screen, Socket-Größe 256 KByte (7) 461
B.12 Durchsatzmessungen TCP über Packet Screen, Socket-Größe 256 KByte (8) 462
B.13 Durchsatzmessungen TCP über Packet Screen, Socket-Größe 64 KByte (1) 462
B.14 Durchsatzmessungen TCP über Packet Screen, Socket-Größe 64 KByte (2) 463
B.15 Durchsatzmessungen TCP über Packet Screen, Socket-Größe 64 KByte (3) 463
B.16 Durchsatzmessungen TCP über Packet Screen, Socket-Größe 64 KByte (4) 464
B.17 Durchsatzmessungen TCP über Packet Screen, Socket-Größe 64 KByte (5) 464
B.18 Durchsatzmessungen TCP über Packet Screen, Socket-Größe 64 KByte (6) 465
B.19 Durchsatzmessungen TCP über Packet Screen, Socket-Größe 64 KByte (7) 465
B.20 Durchsatzmessungen TCP über Packet Screen, Socket-Größe 64 KByte (8) 466
B.21 Datagrammdurchsatz abhängig von der Schreibzugriffslänge (1) . 466

B.22 Datagrammdurchsatz abhängig von der Schreibzugriffslänge (2) . 467
B.23 Durchsatzmessungen TCP über Packet Screen ohne Reduzierung der MTU-Größe (1) 467
B.24 Durchsatzmessungen TCP über Packet Screen ohne Reduzierung der MTU-Größe (2) 468
B.25 Datagrammdurchsatz über Packet Screen 468
B.26 Durchsatz über "Minimalen Generischen Proxy" ("MGproxy") . . 469

F. Tabellenverzeichnis

4.1 SONET-Übertragungsraten 228
4.2 "Protokoll-Overhead" in "Classical IP"-Netzen 229
4.3 Zusammenfassung der Umlaufmessungen 250
4.4 Zusammenfassung der Transaktionsmessungen 256
4.5 Transaktionsdurchsatz bei RSA-Authentisierung 276
4.6 Datagrammdurchsatz über Packet Screen 297
4.7 Zusammenfassung der Umlaufmessungen über eine Packet Screen 305
4.8 Nachrichtenumläufe über "Socks" 317
4.9 Transaktionen über "Socks" (Bastion-Setup) 320
4.10 Nachrichtenumläufe über "MGproxy" 330
4.11 Transaktionen über "MGproxy" 331

C.1 Verkehrsstatistik: Paketlängen 470
C.2 Verkehrsstatistik: pro Verbindung übertragene Byte 471

G Glossar

AAL: Die "ATM Adaption Layer" (AAL) sorgt für das Zerlegen der Nutzdaten in Zellen und nach der Übertragung wieder für das Zusammenfügen der Zellen. Diese Verarbeitung findet in der "Segmentation and Reassembly Sublayer" statt. Zusätzlich stellt AAL mit der "Convergence Sublayer" eine dienstabhängige Anpassungsschicht bereit.

ARP: Dieses Protokoll wird zum Auflösen von IP-Adressen in physikalische Netzadressen benötigt. ARP kann nur auf "Broadcast"-Netzen eingesetzt werden.

ATM: Eine Netzwerktechnologie für Hochgeschwindigkeitsnetze, die sowohl für LANs als auch für WANs eingesetzt werden kann. ATM basiert auf einem verbindungsorientierten Konzept, bei dem Daten in Paketen fester Länge (Zellen) übertragen werden.

ATMARP: Ein Client-Server-basiertes Protokoll, das für das Auflösen von IP-Adressen in ATM-Adressen verwendet wird. Im Gegensatz zum herkömmlichen ARP ist keine "Broadcast"-Funktionalität erforderlich.

ATM Forum: Eine Organisation, die wichtige ATM-Protokolle standardisiert.

Bastion: Ein besonders abgesichertes System, das überwunden werden muß, um Endsysteme im jeweils anderen Netz zu erreichen, wird als Bastion bezeichnet. Auf einer Bastion werden Proxies und Anwendungs-Gateways eingesetzt, um eine Zugriffskontrolle auf den zwischen internen und externen Systemen übertragenen Daten zu ermöglichen.

Broadcast: Das gleichzeitige Senden einer Dateneinheit an alle Systeme eines Netzes wird als "Broadcast" bezeichnet.

Classical IP and ARP over ATM: (kurz "Classical IP") ist eine Lösung zur Übertragung von IP-Dateneinheiten über ein ATM-Netz. Das herkömmliche ARP wird bei CLIP durch eine Client-Server-Lösung (ATMARP) ersetzt.

Client: Ein Client ist ein Prozeß, der auf die Dienste eines Servers zugreift.

CLIP: siehe "Classical IP and ARP over ATM"

CPU-Last: Die CPU-Last gibt die über einen bestimmten Zeitraum gemittelte Anzahl von Prozessen oder Threads an, die gleichzeitig die CPU beanspruchen. Dabei wird davon ausgegangen, daß jeder einzelne Prozeß die CPU etwa gleich stark belastet.

Datagramm: Eine Dateneinheit eines verbindungslosen Protokolls.

Datendurchsatz: Die Menge der Daten, die in einer bestimmten Zeit über ein Netz übertragen wird.

Dateneinheit: Eine Menge von Daten, die eine syntaktische Einheit im Rahmen eines Kommunikationsprotokolls bildet und zwischen Kommunikationspartnern übertragen wird. Eine Dateneinheit enthält unter anderem Adreßinformationen, die Sender und Empfänger spezifizieren.

"Denial of Service"-Angriff: Ein "Denial of Service"-Angriff ist ein Angriff auf die Verfügbarkeit eines Systems, eines Dienstes oder auf die auf einem System gespeicherten Daten.

Dienstgüte: Die Dienstgüte ("Quality of Service") definiert die Qualität einer Diensterbringung an einer wohldefinierten Schnittstelle. In dieser Arbeit wird die Dienstgüte in erster Linie als Eigenschaften betrachtet, die ein Netz für die Kommunikation zwischen Endsystemen zur Verfügung stellt.

DNS: Eine verteilte Datenbank, die für die Abbildung zwischen IP-Adressen und einer hierarchischen Namenstruktur verwendet wird.

Durchsatz: Die Anzahl der Aufträge (Dateneinheiten, Verbindungen, Berechnungen, Kunden, Patienten usw.), die in einer bestimmten Zeit bearbeitet wird.

Edge-Device: Als "Edge-Device" wird ein System bezeichnet, das einen Übergang zwischen Ethernet und ATM-Netz bildet. Das "Edge-Device" überträgt hierzu Ethernet-Pakete über eine virtuelle Verbindung an LANE- oder MPOA-Clients im ATM-Netz. [ATM Forum 97d]

Endsystem: Ein Endsystem ist ein Gerät, das an ein Netzwerk angeschlossen ist. Auf Endsystemen werden die Anwendungen von Benutzern ausgeführt und/oder Dienste erbracht.

Ethernet: Eine sehr weit verbreitete Netzwerktechnologie für lokale Netzwerke. In der ursprünglichen Form ist das Ethernet lediglich ein passives Coaxial-Kabel, an das aktive, direkt mit einem Endsystem verbundene Komponenten angeschlossen werden. Heute ersetzen "Multiport-Repeater" (Hubs) oder (Ethernet-)Switches zusammen mit kostengünstigen UTP-Kabeln zunehmend die bisher verwendeten Coaxial-Kabel.

Firewall: "Ein Firewall ist eine Schwelle zwischen zwei Netzen, die überwunden werden muß, um Systeme im jeweils anderen Netz zu erreichen. Es wird dafür gesorgt, daß jede Kommunikation zwischen den beiden Netzen über den Firewall geführt werden muß. Auf dem Firewall sorgen Zugriffskontrolle und Audit dafür, daß das Prinzip der geringsten Berechtigung durchgesetzt wird und potentielle Angriffe schnellstmöglich erkannt werden." [Ellermann 93]

Firewall-Komponente: Firewall-Komponenten sind Bausteine, aus denen Firewalls aufgebaut werden. Firewall-Komponenten können auf unterschiedlichen Ebenen des Protokollstapels realisiert werden. Die erreichte Granularität der Zugriffskontrolle hängt von den auf der jeweiligen Ebene zur Verfügung stehenden Informationen ab.

Gateway: Ein Gateway ist ein System, das an mindestens zwei Netze angeschlossen ist und das auf Anwendungsebene eine Kommunikation zwischen beiden Netzen ermöglicht.[1]

Gateway-Firewall: Ein Gateway-Firewall ist ein besonders abgesichertes Gateway, das Zugriffskontrolle und Audit der übertragenen Daten durchführt.

IISP: Ein Vorgänger von PNNI.

IP: Ein Protokoll zur verbindungslosen Übertragung von Datagrammen über unterschiedliche Netzwerktechnologien.

LAN: Ein lokales Netzwerk (LAN) bezeichnet ein Netz, daß eine maximale Ausdehnung von wenigen Kilometern nicht überschreitet.

LAN Emulation: LANE ist eine Lösung zur Übertragung von Dateneinheiten herkömmlicher Netze (z. B. Ethernet oder Token-Ring) über ein ATM-Netz.

LIS: siehe "Logisches IP-Subnetz"

Localhost: "Localhost" (IP-Adresse: `127.0.0.1`) ist eine Pseudo-IP-Adresse, über die jeder IP-Knoten auf seine eigenen Netzwerkdienste zugreifen kann,

[1] In anderen Arbeiten wird der Begriff Gateway oft weiter gefaßt und allgemein für Systeme verwendet, die zur Kopplung von Netzen mit unterschiedlicher Netzarchitekturen dienen.

ohne daß Datagramme über eine physikalische Netzwerkschnittstelle gesendet werden müssen.

Logisches IP-Subnetz: Alle Systeme, die zu einem logischen IP-Subnetze (LIS) gehören, müssen an dasselbe ATM-Netz angeschlossen sein und dieselbe IP-Subnetz-Adresse verwenden. Die Systeme eines LIS kommunizieren direkt miteinander. Auf Systeme außerhalb des LIS wird über einen Router zugegriffen.

Maximum Transfer Unit: ist die Länge der größten IP-Dateneinheit, die über ein Netz übertragen werden kann.

MTU: siehe "Maximum Transfer Unit"

Nachricht: Eine Nachricht sind beliebige, von einer Anwendung generierte Daten, die an einen Kommunikationspartner gesendet werden.

Nachrichtenlaufzeit: Die Nachrichtenlaufzeit ist definiert als die Zeit, die für die Übertragung einer Nachricht vom Sender zum Empfänger benötigt wird.

"Native"-ATM-Anwendung: Anwendungen, die ein ATM-Netz zur Übertragung von Daten nutzen, ohne die zu übertragenden Daten zuerst in Dateneinheiten herkömmlicher Protokolle (z. B. IP) zu verpacken.

Nutzdaten: Die Nutzdaten ("Service Data Unit") sind die von einer Protokollschicht transportierten Daten.

Nutzdatenlänge: Die Nutzdatenlänge wird im ISO-OSI Modell als "Service Data Unit Size" kurz "SDU-Size" bezeichnet. Die Nutzdatenlänge ergibt zusammen mit der Header- und Trailer-Länge die PDU-Länge ("Protocol Data Unit Size").

Packet Screen: Eine Packet Screen ist eine Firewall-Komponente, die zwischen zwei oder mehreren Netzen installiert wird und eine Zugriffskontrolle auf IP-Datagrammen durchführt.

PDU-Länge: Die "Protocol Data Unit Size" gibt die Länge einer Dateneinheit einer Protokollschicht an und ergibt sich aus der Summe von Nutzdatenlänge, Header- und Trailer-Länge.

Für die unterschiedlichen Protokollschichten werden die PDU-Längen bezeichnet als TCP-Segmentlänge, UDP-Datagrammlänge oder IP-Datagrammlänge.

Performanz: Die Bezeichnung "Performanz" ist ein Oberbegriff für die verschiedenen Leistungsaspekte von Firewall-Komponenten. Zu diesen Leistungsaspekten zählen Datendurchsatz, Datagrammdurchsatz, Nachrichtenlaufzeit, Verbindungsaufbauzeit, maximale Anzahl paralleler Verbindungen und Transaktionsdurchsatz.

PNNI: Ein Protokoll zum Austausch von Routing-Informationen und zum Aufbau von SVCs.

Prüfsumme: Eine Prüfsumme ist eine Funktion, die über einer Nachricht berechnet wird. Der Empfänger einer Nachricht kann durch das Überprüfen der Prüfsumme Übertragungsfehler erkennen.

PVC: Eine manuell konfigurierte virtuelle Verbindung.

"Quality of Service": siehe Dienstgüte

Schreibzugriffslänge: Die Schreibzugriffslänge gibt die Datenmenge an, die eine Anwendung mit einem Schreibzugriff auf einem Socket dem Kommunikationssystem zur Übertragung übergibt. Die anschließend in einer Dateneinheit übertragene Datenmenge kann sich von der Schreibzugriffslänge unterscheiden.

Server: Ein Prozeß, der einen Dienst anbietet. Häufig werden auch Endsysteme stellvertretend als Server bezeichnet, wenn auf diesen Endsystemen einer oder mehrere Server (Prozesse) ausgeführt werden.

Sicherheitsdomäne: Als (Sicherheits-)Domäne bezeichnet [DoD 85] eine Menge von Objekten, auf die ein Subjekt zugreifen kann.

Socket: Ein Socket ist ein spezieller Datentyp in BSD-UNIX-Systemen, der Anwendungen den Zugriff auf ein Netz ermöglicht.

Socket-Größe: Die Socket-Größe ("Socket Size") gibt den für eine (TCP-)Verbindung in einem Endsystem reservierten Pufferspeicher an. Es wird unterschieden zwischen Sende- und Empfangspuffer.

"Spoofing": Das Fälschen von Adreßinformationen, mit dem der Sender einer Nachricht dem Empfänger eine andere Identität vortäuscht.

SVC: Eine virtuelle Verbindung, die von einem Endsystem bei Bedarf aufgebaut wird.

TCP: Das "Transmission Control Protocol" (TCP) ist ein Protokoll auf der Transportschicht, das Anwendungen eine verbindungsorientierte Übertragung von Daten ermöglicht. TCP enthält Mechanismen zur Wiederherstellung der Reihenfolge übertragener Dateneinheiten beim Empfänger, zum

wiederholten Senden verlorengegangener Dateneinheiten und zur Flußkontrolle.

UDP: Das "User Datagram Protocol" (UDP) ist ein Protokoll auf der Transportschicht, das Anwendungen ein verbindungsloses Übertragen von Daten ermöglicht.

UNI: Das vom ATM-Forum spezifizierte Protokoll, das zwischen Endsystem und Switch zum Aufbau einer virtuellen Verbindung verwendet wird.

URL: Ein "Uniform Resource Locator" beschreibt den Ort (Hostname, Portnummer, Dateiname, ...) und die Art des Zugriffs (Protokoll) auf eine Datei oder einen Dienst. [Berners-Lee et al. 94]

Verbindungsaufbauzeit: Die (TCP-)Verbindungsaufbauzeit bezeichnet die Zeit, die zum Aufbau einer (TCP-)Verbindung benötigt wird. Darin eingeschlossen ist die Dauer des "3-Way-Handshakes" und die Verarbeitung in den Endsystemen.

Virtuelles Netz: Ein "virtuelles Netz" ist ein Netz, das ein (physikalisches) ATM-Netz nicht exklusiv nutzen kann, aber über eine Zugriffskontrolle beim Aufbau von virtuellen Verbindungen gegenüber anderen Nutzern des Netzes abgeschottet wird.

Virtuelle Verbindung: Ein Pfad durch ein ATM-Netz, auf dem Dateneinheiten zwischen zwei Endsystemen übertragen werden. Übertragene Dateneinheiten kommen in der gesendeten Reihenfolge beim Empfänger an und brauchen keine global eindeutigen Adreßinformationen zu enthalten.

WAN: Ein Weitverkehrsnetz (WAN) erstreckt sich als überregionales bzw. globales Netz über größere geographische Entfernungen.

Zelle: Die in ATM-Netzen verwendete Dateneinheit fester Länge. Eine Zelle kann maximal 48 Byte Nutzdaten übertragen und enthält 5 Byte Header-Informationen.

Zugriffskontrolle: Das Einschränken des Zugriffs auf ein Objekt, so daß nur autorisierten Subjekten der Zugriff gewährt wird.

Zugriffskontrollmechanismus: Hardware, Software, organisatorische Maßnahmen oder eine Kombination hieraus, die eine Zugriffskontrolle durchführt.

Index

3-Way-Handshake, 13
3DES, 22, 273, 349

AAL, 53, 109, 142, 146, 229, 487
Aggregation, 221, 301, 308
Angriff
  Broadcast-Ping, 18
  Denial of Service, 17, 85, 93, 96, 98, 488
  IP-Spoofing, 12, 16, 18, 89, 95, 414
  Spoofing, 83, 98, 471
  SYN-Flooding, 20, 269
  TCP-Hijacking, 11, 16
  UDP Echo, 18
Anwendungs-Gateway, 35–36, 157, 375
ARP, 487
ATM, 48, 487
ATM-Anpassungsschicht, *siehe* AAL
ATMARP, 55, 88–94, 487
Authentisierung, 272, 354, 411
Authentizität, 12, 22, 83, 89, 95, 98, 126

Bastion, 39, 119, 185, 187, 202, 206, 362, 487
  parallele, 375–404
Benutzerschicht, 100
Block-Chiffre, 22

Cell Screen, 146–152
Classical IP, *siehe* CLIP
CLIP, 55, 88, 157, 180, 211, 226, 487

Datagrammdurchsatz, 213
  über Packet Screen, 294–303, 372
  Host-zu-Host, 237–240
Datendurchsatz, 212, 217, 356, 488
  über Kombination von Packet Screen und Bastion, 363–364
  über Packet Screen, 290–294
  über Proxy, 316, 322–329, 372
  Host-zu-Host, 218–247
  Host-zu-Host bei UDP, 246
  theoretisch erreichbarer, 226–237
DES, 22, 349
Designated Transit List, *siehe* DTL
Dienstgüte, 52, 109, 158, 488
DNS, 488
  Round-Robin, 377
DTL, 51, 106, 132
Durchsatz, 212, 488
  Einzel-, 213, 258
  Gesamt-, 213, 258

Empfangspuffer, 223
Executable Content, 348

Fenstergröße, 223
Filterregeln, 289, 305, 363, 471
Firewall
  Definition, 28
  dezentraler, 196
  virtuell zentraler, 198, 207
  zentraler, 197
Firewall-Architektur, 36
  für CLIP-Netze, 180
  für Native-ATM, 202
  Gateway-Firewall, 42–43, 183

Kombinationen von Packet Screen und Bastion, 39–42, 185
Packet Screen, 37–39, 183
Firewall-Komponente, 29
Anwendungs-Gateway, 35–36
Packet Screen, 29–33
Proxy, 33–35, *siehe auch* Proxy
Flußkontrolle, 282
fork, 318
Fragmentierung, 233

Gateway-Firewall, 42

Hackertools, 9
HTTP, 260

ILMI, 51, 178
Implementationsfehler, 19, 26, 87, 94, 97
Informationselement, 104
BHLI, 109
BLLI, 108
Called Party Number, 105, 130, 164
Calling Party Number, 105, 164
DTL, *siehe* DTL
Integrated Local Management Interface, *siehe* ILMI
Integrität, 15, 22, 83, 89, 95, 98, 126
IP, 229
IP-Filter, 288
IP-Spoofing, 12, 16, 18, 89, 95, 414
IP-Subnetz
logisches, *siehe* LIS
IP-Switching, 59
IPSEC, 26, 95, 385

Kerberos, 25
Konfigurationsfehler, 21, 26, 87, 94, 97
Kopierlänge, 322, 356
Kryptographie, 21

Label-Switching, 59
LAN Emulation, *siehe* LANE
LANE, 62, 98, 157, 180
LCH-Technik, 148
LIS, 55, 79, 89, 93, 95, 138, 180, 216, 490
Localhost, 243

Managementschicht, 100
MARS, 59, 97
MBit/s, 212
Meßdauer, 218
Meßwerkzeug
für HTTP, 258
Netperf, 217
MGproxy, 322
MPLS, 59
MPOA, 63, 99, 140, 143, 157
MSS, 221, 242
MTU, 220, 231, 301, 490
Multi-Protocol over ATM, *siehe* MPOA
Multicast, 59
Multicast Address Resolution Server, *siehe* MARS
Multiplexing, 415

Nachrichtenlänge, 219, 470
Nachrichtenlaufzeit, 74, 214, 217, 474, 490
über Kombination von Packet Screen und Bastion, 365
über Packet Screen, 303–305
über Proxy, 316–317, 329–330
Host-zu-Host, 247–251
Nagle-Algorithmus, 220, 250, 283, 302, 322
NAT, 36, 168, 382
Native-ATM, 78
Network Address Translation, *siehe* NAT
Netz
logisches, *siehe* LIS
virtuelles, 182, 192, 207, 492

Netzwerkmanagement, 178
Next Hop Resolution Protocol, *siehe* NHRP
NHRP, 56, 94–97, 138, 143, 157
  Verteilung durch, 390

Packet Screen, 29–33, 37, 39, 78, 101, 109, 120, 123, 181, 187, 190, 404
  dynamische, 31
  für ATM-Netze, 142–156
  parallele, 404–411
Padding, 230
Paket-Klassifikation, 155
Paketlänge, 470
  durchschnittliche, 310
Paketparallelität, 373, 404–411
Parallelisierungskonzepte, 373
Performanz, 211
Permanent Virtual Circuit, *siehe* PVC
PGP, 25
Pipeline-Konzept, 373
Pipelining, 412
Plug-Gateway, 333
PNNI, 50, 125, 132, 179
Private Network-Network Interface, *siehe* PNNI
Proxy, 33–35, 39, 42
  für ATM-Netze, 156–177
  Native-ATM, 158–166
  Native-ATM-, 202, 206
  paralleler, 375–404
  Socks, 314, 398
  transparenter, 166–177, 207, 388
Puffer-Überlauf, 19
PVC, 49

Quality of Service, *siehe* Dienstgüte

RC4, 23, 350
Reassemblierung, 53, 143, 146, 180, 235, 308
RSA, 23, 273, 354

Scanner, 348
Schlüsselaustausch, 354
Schlüssellänge, 22, 353
Schreibzugriffslänge, 220, 491
Secure DNS, 25
Segmentierung, 53, 143, 308
Sendepuffer, 223
SET, 25
Sicherheitsdienstgüte, 125–138
Sicherheitsmechanismen, 99
Sicherheitsrichtlinien, 8, 27, 99, 115, 126, 155
Signalisierungsfilter, 101, 103–124, 206
Signalisierungskontroller, 102, 124–138, 187, 207, 393
Signalisierungsnachricht, 104
  ADD_PARTY, 112
  CONNECT, 110, 163
  CONNECT_ACKNOWLEDGE, 110
  RELEASE, 111
  RESTART, 113
  SETUP, 105, 163
  STATUS, 113
Signalisierungsproxy, 102
Skalierbarkeit, 395, 404, 418
SNAP, 56, 229
Sniffer, 10, 81
SNMP, 88, 178
Socket-Größe, 223, 271, 328, 341
SONET, 227
Spoofing, 83, 98, 471
SSH, 25
SSL, 25, 273, 353
Steuerungsschicht, 99
Strom-Chiffre, 22
SVC, 49, 491
Switched Virtual Circuit, *siehe* SVC
SYN-Flooding, 20
Synchronisation, 212, 260, 261

TCP, 229, 491

TCP-Hijacking, 11, 16
TCP-Segment, 232
TCP-Verbindungsaufbau, 13
Testnetz, 215
TLS, 25, 273
Transaction TCP, 413
Transaktion, 251, 264, 269, 319
Transaktionsdurchsatz, 214, 217, 396, 399, 474
  über Proxy, 337–342, 372
  Host-zu-Host, 269–277

UDP, 229, 492
UDP Echo, 18
Umlaufzeit, 248
UNI, 49, 86, 110, 160, 492
User-Network Interface, *siehe* UNI

Verbindungen
  parallele, 214
    über Proxy, 333–337
    Host-zu-Host, 263–269
Verbindungsannahme, 339
Verbindungsaufbau, 251
Verbindungsaufbauzeit, 214, 217, 397, 399, 411
  über Kombination von Packet Screen und Bastion, 366
  über Packet Screen, 305–307
  über Proxy, 317–321, 330–332, 372
  Host-zu-Host, 251–258
Verbindungsmanagement, 74
Verbindungsparallelität, 374–404
Verfügbarkeit, 17, 85, 93, 96, 98, 208
Verteilung
  über virtuelle Verbindungen, 388
  dezentrale, 395
  durch DNS, 377
  durch Filtern von Datagrammen, 401
  durch NAT, 382
  durch NHRP, 390
  durch Proxy, 381
  durch Redirect, 397
  durch Signalisierungskontroller, 393
  durch Socks, 398
  durch TCP-Router, 385
  durch zufällige Auswahl, 395
  zentrale, 377
Verteilungsfunktion, 374
Vertrauensintervalle, 217, 259
Vertraulichkeit, 10, 21, 81, 89, 94, 126
Video on Demand, 160
Virtual Channel Identifier, 48
Virtual Path Identifier, 48
Virtuelle Verbindung, 48, 391
Virtueller Kanal, 48
VPN, 44

Zelle, 48, 78, 83, 87, 143, 146, 152, 180, 229, 230, 235, 308