Universität Karlsruhe (TH)
Fakultät für Informatik
Institut für Telematik
76128 Karlsruhe

# Netzwerk-Management und Hochgeschwindigkeits-Kommunikation

# Teil XX

## Seminar SS 1999

Herausgeber:
**Roland Bless**
**Stefan Dresler**
**Daniel Müller**
**Klaus Wehrle**

*Universität Karlsruhe (TH)*
*Institut für Telematik*
http://www.telematik.informatik.uni-karlsruhe.de/

Interner Bericht 1999-11
ISSN 1432–7864

## Übersicht

Der vorliegende interne Bericht enthält die Beiträge zum Seminar „Netzwerk-Management und Hochgeschwindigkeits-Kommunikation“, das im Sommersemester 1999 zum zwanzigsten Mal stattgefunden hat.

Die Themenauswahl kann grob in folgende vier Blöcke gegliedert werden:

Ein Block ist der Leistungssteigerung im Internet gewidmet. Hier geht es weniger um das Wachstum der zur Verfügung stehenden Bandbreite als vielmehr um neuartige Ansätze zur Verbesserung von Diensten und ihrer Unterstützung durch das Netzwerk. Dies umfaßt neuartige Netzwerkarchitekturkonzepte wie „Programmierbare Netzwerke“ und die Erweiterung von Mechanismen zur Zwischenspeicherung von audio-visuellen Datenströmen sowie elementare Funktionen zur Unterstützung der Gruppenkommunikation und schließlich neuere Ansätze zur Dienstgüteunterstützung im Internet (Differentiated Services).

Ein zweiter Block beschäftigt sich mit speziellen Netzwerktechniken. Die Spanne reicht hier von nahezu rein optischen Netzwerken über dedizierte Kontrollnetzwerkarchitekturen bis hin zu den „Kleinst“-Netzwerktechniken USB und IrDA zur Kopplung von Rechnerkomponenten oder einigen Rechnern in unmittelbarer Umgebung.

Der dritte Block umfaßt den Themenbereich zur Anonymität und Vertraulichkeit im Internet.

Die Beiträge des vierten Blocks beschäftigen sich schließlich mit Themen der Kopplung von Netzwerken. In diesem Zusammenhang werden die Kopplung mehrerer privater Netzwerke über das Internet (Virtual Private Networks) sowie der Transport von Signalisierungsnachrichten des Telefonnetzes zur Rufsteuerung über das Internet behandelt.

## Abstract

This technical report includes student papers produced within small lessons called seminar on “Network Management and High Speed Communications”. For the twentieth time this seminar has attracted a large number of diligent students, proving the broad interest in topics of network management and high speed communications.

The topics of this report may be divided into four blocks:

One block is devoted to the aspects of how to increase performance in the Internet. Instead of concentrating on the permanently increasing growth of bandwidth, new approaches for enhancing current services and their support by the network are discussed. This includes novel concepts for network architectures such as “programmable networks” and enhancements of mechanisms for caching audio-visual data streams as well as basic functions for supporting group communication and new approaches for supporting quality of service in the Internet (Differentiated Services).

A second block deals with special network technologies. This comprises almost pure optical networks, dedicated control network architectures, and also very-small-network technologies USB and IrDA for interconnecting peripheral components and computers in the near range.

The third block deals with anonymity and privacy in the Internet.

The articles in the fourth block cover topics around the interconnection of networks. In this context, linking of several private networks together over the Internet (virtual private networks) as well as transport of signalling messages of the telephone network for call control over the Internet are addressed.

# Inhaltsverzeichnis

# Vorwort

Das Seminar „Netzwerk-Management und Hochleistungs-Kommunikation“ am Institut für Telematik erfreut sich weiterhin großer Beliebtheit. Die Telematik als Verbindung von Telekommunikation und Informatik entfaltet immer mehr von ihrer Dynamik. Dies zeigt sich an der breiten öffentlichen Diskussion über die zukünftige Bedeutung des Internet, das ja schon lange dem akademischen Bereich entwachsen ist, ebenso wie an der wachsenden Bedeutung der Mobilkommunikation, ob über Satellit oder terrestrisch. Unabhängig vom zugrundeliegenden Netzwerk wird die Betrachtung von Dienstgütefaktoren – und die Abrechnung der Nutzung eines Dienstes – angesichts knapper Ressourcen in Zukunft große Wichtigkeit bei der Ausrichtung der künftigen Netze auf die Bedürfnisse der Anwender besitzen. In diesem Umfeld existiert eine derartige Vielzahl von innovativen Forschungsergebnissen und Produktideen, daß die Behandlung in anderen Lehrveranstaltungen so detailliert nicht möglich ist.

Jetzt liegt auch der nunmehr zwanzigste Seminarband als Interner Bericht vor. Durch die engagierte Mitarbeit der beteiligten Studenten konnte so zumindest ein Ausschnitt aus dem komplexen und umfassenden Themengebiet klar und übersichtlich präsentiert werden. Für den Fleiß und das Engagement der Seminaristen sei daher an dieser Stelle recht herzlich gedankt.

Die weiterhin gute Resonanz bei den Studenten bestätigt uns darin, auch im kommenden Wintersemester 1999/2000 ein derartiges Seminar – natürlich mit geändertem aktuellem Inhalt – durchzuführen, so daß bald ein weiterer interner Bericht mit neuen Forschungsergebnissen aus innovativen Seminarbeiträgen erscheinen wird. Doch vorerst sollen im vorliegenden Band folgende Themengebiete vorgestellt werden:

## Programmierbare Netzwerke

Traditionelle Netzwerke bestehen aus Komponenten, deren Hauptaufgabe in der möglichst schnellen Weiterleitung von Daten besteht. An einer neueren Entwicklung, die auf einem anderen Ansatz basiert, wird jedoch in letzter Zeit intensiv geforscht: programmierbare bzw. aktive Netzwerke. Diese unterscheiden sich in zweierlei Hinsicht von herkömmlichen Netzwerken: zum einen nehmen die Komponenten im Netzwerk bereits eine Verarbeitung der zu transportierenden Daten vor, so daß sie eine gewisse Unterstützung von Anwendungen bereitstellen, zum anderen sind die Netzwerkkomponenten für Anwender programmierbar geworden. Dadurch kann die Funktionalität der Netzknoten beeinflußt oder geändert werden, was die Grundlage für die schnelle Umsetzung und Implementierung neuer Dienste bildet.

In dem Beitrag werden die grundlegenden Konzepte von programmierbaren Netzwerken vorgestellt und anhand von Beispielen illustriert. Anschließend werden einige aktuelle Forschungsprojekte kurz vorgestellt.

## Möglichkeiten des Cachings bei Audio- und Video-Datenströmen

Um Dienste wie Video on Demand (VoD) über bestehende Netzwerkstrukturen und Cache-Systeme nutzen zu können, müssen diese auf verschiedene Weise erweitert werden. Aktuelle Cache-Systeme berücksichtigen kontinuierliche Medien nur ungenügend und führen somit zu sehr hoher Netzlast, indem sie z.B. nur Unicast-Verbindungen erlauben. Ansätze zur dynamischen Selbstkonfiguration von Caches mit Hilfe von Helpern sowie der Implementierung von Staggered VoD (S-VoD) sind nur zwei Vorschläge, die versuchen, den Ansprüchen von heutigen multimedialen Anwendungen gerecht zu werden. Diese Ausarbeitung liefert neben einer kurzen Zusammenfassung über die Funktionsweise von statischen Caches Lösungsvorschläge für Implementierungen sowie Simulationsauswertungen einer bestehenden Testumgebung.

## Multicast Routing – Wegewahlverfahren für die Gruppenkommunikation

Der Bedarf an Gruppenkommunikation wächst zunehmend. Im Internet wird Gruppenkommunikation (IP-Multicast) seit geraumer Zeit unterstützt, obwohl sie noch nicht überall im Netz unterstützt wird. Seit der Einführung gibt es einige alternative Ansätze zur Unterstützung der Multicast-Infrastruktur. Der Kern solcher Mechanismen bilden die Wegewahlverfahren (Routing), die eine möglichst effiziente Verteilstruktur für die Gruppenkommunikation berechnen und verwalten sollen.

In diesem Beitrag werden zunächst die grundlegenden Mechanismen erläutert, welche die Gruppenkommunikation im Internet ermöglichen. Anschließend werden einige der Routing-Verfahren für IP-Multicast genauer vorgestellt, deren Entwicklung immer noch nicht abgeschlossen ist.

## Ressourcenreservierung für Differentiated Services

Der Bedarf an Dienstgüte im Internet wird immer deutlicher. Aus diesem Grund wurde von der Internet Engineering Task Force (IETF) die Differentiated-Services-Architektur entworfen, welche verschiedenste höherqualitative Dienste für das Internet anbieten soll. Die Skalierbarkeit dieser Architektur und die Garantien einzelner Dienste wurden bereits untersucht und festgestellt. Jedoch gibt es bei der Verwaltung und Reservierung von Netzwerkressourcen in diesem Kontext verschiedene Meinungen und Ansätze. Der Beitrag „Ressourcenreservierung in Differentiated Services Netzen“ gibt eine kurze Einführung in die neue QoS-Architektur und vergleicht die bestehenden Ansätze zur Ressourcenreservierung.

## Optische Netzwerke

Bereits seit geraumer Zeit werden optische Verfahren zur Datenübertragung in Telekommunikationssystemen genutzt. Allerdings werden optische Signale dabei in Zwischensystemen zunächst wieder in elektrische Signale zurückverwandelt, bevor eine weitere Bearbeitung stattfindet, wie etwa eine Wegbestimmung für die Weiterleitung der Daten. Der Beitrag behandelt neben diesen „herkömmlichen“ optischen Netzen auch neuere Entwicklungen auf dem Gebiet der optischen Übertragungsverfahren wie das Wellenlängenmultiplex, mit dessen Hilfe gleichzeitig mehrere unabhängige Kanäle über eine einzige Glasfaser übertragen werden können. Außerdem werden Komponenten zur Weiterleitung von optischen Signalen beschrieben, durch welche teilweise eine Umwandlung der optischen in elektrische Signale in Zwischensystemen vermieden werden kann.

## Das LonTalk-Protokoll für Kontrollnetzwerke

Neben dem Internet-Protokoll gibt es eine Vielzahl von Protokollen für Spezialaufgaben. Das LonTalk-Protokoll stellt nicht nur ein einziges Protokoll zur Verfügung, sondern ist eigentlich eine umfassende Architektur mit einer Reihe von Konzepten. In dem Beitrag werden zuerst Anwendungsgebiete für ein Kontrollprotokoll wie LonTalk dargelegt. Anschließend wird auf den von LonTalk zur Verfügung gestellten Dienst, die zugrundeliegende Architektur mit ihren Schichten, die Adressierung und das Namens- und Managementkonzept sowie natürlich auf die verwendeten Protokolle eingegangen.

### Universal Serial Bus – Funktionsweise und Protokolle

Der Personal Computer (PC) ist ein gutes Beispiel für dedizierte Kommunikationsschnittstellen. Ein handelsüberlicher PC verfügt über mindestens sieben verschiedene Schnittstellen (Maus, Tastatur, Modem, Monitor, Mikrophon, Lautsprecher, Erweiterungskarten, etc.) an denen sich jeweils nur ein bestimmtes Peripheriegerät anschließen läßt.

Mit dem Universal Serial Bus wurde nun eine gemeinsame Schnittstelle entworfen, welche die meisten der genannten Peripheriegeräte bedienen kann. Darüber hinaus bietet der Universal Serial Bus noch weitere Vorteile, wie Anschließen und Entfernen des Geräts zur Laufzeit, Stromsparmechanismen und Unterstützung für qualitätsbasierte Kommunikation. In dem Beitrag werden die grundlegenden Konzepte und Mechanismen des USB unter netzwerktechnischen Gesichtspunkten beschrieben.

### IrDA – Der Standard für Infrarotkommunikation

In der heutigen Zeit spielt Mobilität eine immer wichtigere Rolle. So wird dieser Trend auch deutlich durch zahllose leidtungsfähige Notebooks und digitale Assitenten (PDAs) unterstützt. Jedoch fehlte diesen Geräten bisher eine mobile Kommunikationsschnittstelle, um ad-hoc-Netzwerke aufbauen oder Daten mit anderen Geräten austauschen zu können. Die Infrared Data Association (IrDA) hat mit den IrDA-Standards diese Lücke geschlossen und verschiedene Protokolle zum Anschluß mobiler Geräte an LANs oder zum Austausch von Daten auf verschiedenste Art und Weise definiert. Der Beitrag stellt die Architektur und die Protokolle vor, die im Rahmen der IrDA entwickelt wurden.

### Anonymität und Vertraulichkeit im Internet

Mit dem Wachstum des Internet und dessen zunehmender Nutzung für private und geschäftliche Zwecke gewinnen auch Sicherheitsfragen an Bedeutung. Eine bisher weniger beachtete Ausprägung von Sicherheit ist Anonymität, also die Eigenschaft, daß das Netz selbst bzw. ein externer Beobachter nicht nachvollziehen können, wer mit wem kommuniziert – gegebenenfalls kennen nicht einmal die Kommunikationspartner selbst ihre gegenseitige wahre Identität. Der Beitrag stellt zwei Ansätze zur Bereitstellung von Anonymität im Internet, *MIXe* und *Crowds*, sowie eine Anwendung des ersten im sogenannten *Onion Routing* vor.

### Virtual Private Networks

Kaum ein größeres Unternehmen besitzt heutzutage nur einen Standort. Um nun aber trotz verteilter Standorte den Eindruck einer „lokalen“ Vernetzung mit den entsprechenden Dienstgarantien und Sicherheitsmechanismen bieten zu könnnen, wurden sogenannte Virtuelle Private Netze (VPN) entwickelt. Der Beitrag „Virtual Private Networks“ schildert die Möglichkeiten, VPNs zu errichten, und geht insbesondere auf Protokoll- und Sicherheitsmechanismen ein.

### Transport von Signalisierungsnachrichten über IP

Es ist zu erwarten, daß die traditionellen Telefonnetze und das Internet in Zukunft weiter zusammenwachsen werden. Somit müssen aber auch Übergänge zwischen beiden Netzen geschaffen werden. Dies betrifft vor allem den Austausch von Nachrichten zur Steuerung und

Abwicklung von Anrufen (Signalisierungsnachrichten). Die hohen Verfügbarkeitsanforderungen des Telefonnetzes können nicht ohne weiteres auf das Internet übertragen werden. Hierfür bedarf es einer besonderen Architektur und neuer Protokolle, welche die Anforderungen erfüllen können.

Der Beitrag führt in diese Thematik ein und gibt einen Überblick über die Entwicklungen und Arbeiten der Arbeitsgruppe „Sigtran“ der Internet Engineering Task Force (IETF).

# Programmierbare Netzwerke

Peter Ebinger

**Kurzfassung**

Ein programmierbares Netzwerk befördert Daten nicht nur, sondern verarbeitet diese auch „aktiv“ weiter. Es erlaubt noch weitergehend, diese Verarbeitungsprogramme innerhalb des Netzwerkes frei zu manipulieren, zu erweitern oder auch ganz auszutauschen. Daraus entsteht eine größere Flexibilität als in herkömmlichen Netzen und man kann neue Dienste schnell in bestehende Netzen einführen. Eine Vielzahl von Beispielanwendungen, für die programmierbare Netze von Vorteil sind, wurde bereits entwickelt, z.B. Online-Auktionen und verteiltes adaptives Caching. Man hat eine große Freiheit beim Entwurf eines programmierbaren Netzwerkes, doch ist eine „richtige“ Architektur die Grundlage für ein effizientes und trotzdem sicheres System. So stellt sich die Frage nach der richtigen Programmiersprache, dem Betriebssystem und der Knoten-Hardware, um dieses Ziel zu realisieren.

## 1 Einleitung

Durch die immer weitergehende Vernetzung der vorhandenen Rechnersysteme, die steigende Rechenleistung und die große Leistungsfähigkeit der Übertragungssysteme werden neue Netzwerkdienste denkbar, die zum Teil auch schon in der Praxis eingesetzt werden. Dabei wird den Rechnern und Übertragungssystemen immer mehr Leistung abverlangt. Programmierbare Netze bieten neben der weiter fortschreitenden Verbesserung der Technik die Möglichkeit, vorhandene Rechen- und Netzressourcen flexibler und besser zu nutzen [Orti98]: „Es kommt das Programm zu den Daten, anstatt die Daten zum Programm“.

### 1.1 Was sind programmierbare Netzwerke?

In „traditionellen“ Rechnernetzen werden die Daten vom Sender in Pakete verpackt und dann von Netzknoten zu Netzknoten über die einzelnen Übertragungsstrecken weitergereicht, bis sie am Zielrechner angekommen sind. Dabei wird ein Paket, das in einem Vermittlungsknoten ankommt, zwischengespeichert und dann über die ermittelte Übertragungsleitung weitergeschickt (Store-and-Forward).

In programmierbaren Netzwerken dagegen enthält das Datenpaket zusätzlich noch ein Programm, das im Netzknoten ausgeführt wird, woraus sich anschließend ergibt, wie der Netzknoten mit dem Paket im weiteren verfährt (Store-Compute-and-Forward), d.h. das Netzwerk überträgt nicht nur die Daten von A nach B, sondern es operiert unter Umständen direkt auf den Anwendungsdaten, manipuliert sie oder löst bestimmte Aktionen im Netzwerk aufgrund der übertragenen Daten aus. Je nach Ausprägung kann so ein Datenpaket auch ausschließlich aus Programmcode bestehen. Teilweise können auch Programmteile in den Netzknoten abgelegt werden, die von nachfolgenden Paketen aufgerufen werden (vgl. Abb. 1).

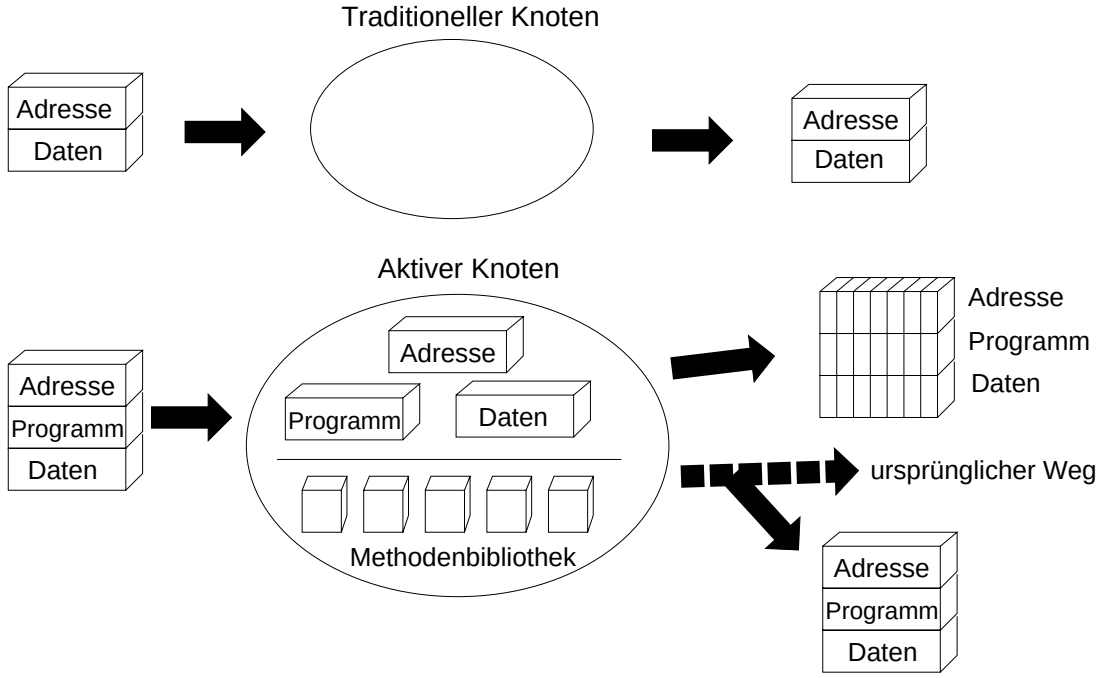


Abbildung 1: Traditionelle und programmierbare Netze

Es wird neben dem Begriff des programmierbaren Netzwerks auch der des aktiven Netzwerks verwendet. Dabei wird mit „aktiv“ ausgedrückt, daß das Netzwerk die Daten nicht nur befördert, sondern diese auch „aktiv“ weiterverarbeitet, z.B. kann dadurch ein Netzwerk Videoströme bearbeiten und verschiedenen Nutzern mit verschiedenen Anforderungen entsprechende Bildqualitäten zur Verfügung stellen. Ein programmierbares Netz erlaubt nun noch weitergehend, diese Verarbeitungsprogramme innerhalb des Netzwerkes frei zu manipulieren, zu erweitern oder auch ganz auszutauschen. Daraus entsteht eine viel größere Flexibilität. Es ist allerdings keine scharfe Trennung der beiden Begriffsdefinitionen vorhanden.

Üblicherweise wird der normale Benutzer nicht selbst Protokolle oder Programme für Netzknoten entwerfen, sondern auch weiterhin auf Programme von Drittanbietern zurückgreifen, wie er das auch bei herkömmlichen Anwendungsprogrammen tut.

## 1.2 Vorteile und Beispielanwendungen

Der Vorteil von programmierbaren Netzwerken liegt in der größeren Flexibilität. Man kann die Rechenlast flexibler auf die einzelnen Rechner im Netz verteilen und dadurch unter Umständen eine deutlich gesteigerte Leistungsfähigkeit des Gesamtsystems erreichen. So ist gerade z.B. im WWW die Frage des *Cachings* für die Leistung des Gesamtsystems enorm wichtig. Es ist hier entscheidend, die Zwischenspeicher an der richtigen Stelle im Netzwerk – zwischen der Datenquelle (WWW-Server) und den Benutzern – zu positionieren, um zwischen kurzen Antwortzeiten und geringem Gesamtspeicherbedarf einen optimalen Weg zu finden [CBZS98].

Es kann also teilweise eine bessere Funktionalität und Leistungsfähigkeit der Netzdienste erreicht werden, wenn sie von Knoten innerhalb des Netzwerks direkt unterstützt werden. Bei einem programmierbaren Netzwerk können sogar die Dienste und Protokolle selbst schnell und flexibel erweitert oder ausgetauscht werden. Dies kann auch zur Laufzeit passieren. Es können also neue Dienste auf einem bestehenden Netzwerk und seinen Netzknoten implementiert werden, an die man bei der Netzwerkplanung noch gar nicht gedacht hat.

Ein programmierbares Netzwerk bietet die Möglichkeit, jeder Anwendung maßgeschneiderte Dienste zur Verfügung zu stellen. Es können mehrere Protokolle parallel betrieben werden, was bei der Einführung von neuen Diensten und Protokollen eine einfachere Migration ermöglicht.

Für Forscher und Entwickler bieten programmierbare Netzwerke daher eine gute Plattform, um zu experimentieren. Dabei kann ein neues Protokoll ohne Zentralstelle bei gegenseitiger Übereinkunft im Netzwerk eingesetzt und getestet werden.

Im folgenden sollen nun ein paar Beispielanwendungen für programmierbare Netze vorgestellt werden.

- **Online-Auktion:** Ein WWW-Server, der Online-Auktionen anbietet, gehört zur Zeit zu den gefragtesten Angeboten im Internet. So ein Server sammelt und bearbeitet die Gebote der WWW-Benutzer. Er teilt auf Anfrage auch die aktuellen Höchstgebote für die angebotenen Objekte mit. Wenn der Server stark belastet ist, sind diese Preise aber wegen der großen Verzögerung unter Umständen nicht mehr gültig, wenn sie beim WWW-Benutzer ankommen. Dies führt zu sinnlosen Geboten, da bereits ein höherer Preis für das Objekt der Begierde geboten wurde, als man selbst in diesem Moment als Gebot abschickt.

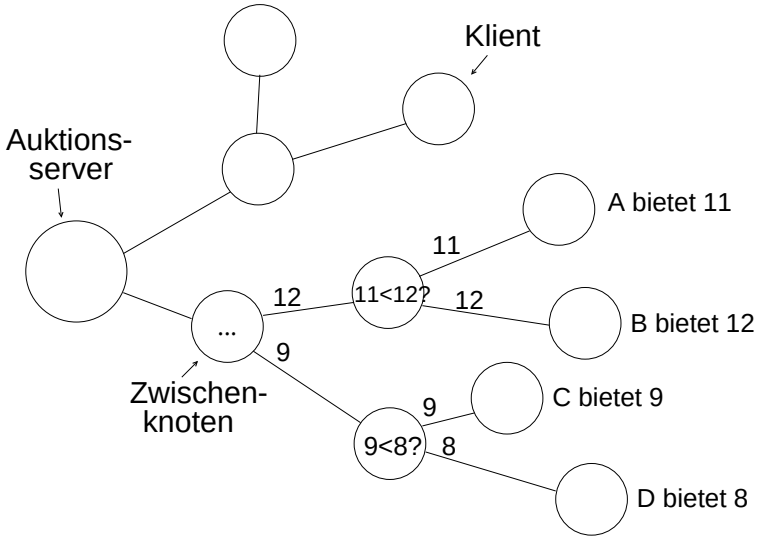


Abbildung 2: Online-Auktion

Bei einem programmierbaren Netzwerk kann man erreichen, daß nur die höchsten – und damit gültigen – Gebote bis zum Server vordringen und sich dadurch deutlich kürzere Antwortzeiten für den Benutzer ergeben. Er erhält sofort eine negative Rückmeldung, wenn sein „Gebot-Paket“ auf dem Weg auf ein anderes Paket mit höherem Gebot trifft. Dies ist in Abb. 2 illustriert: hier erreicht nur das Gebot von Klient B den Auktionsserver, da es das höchste ist – alle anderen Klienten (B, C und D) bekommen bereits von den Zwischenknoten eine negative Rückmeldung.

Zusätzlich können die Netzknoten noch das höchste Gebot, das sie passiert hat, für jedes Objekt zwischenspeichern, und von Zeit zu Zeit bekommen sie vom Auktionsserver die aktuellen Höchstpreise mitgeteilt. Durch dieses System wird die Last des Auktionsservers auf die verschiedenen aktiven Netzknoten verteilt und die Antwortzeiten für die Benutzer deutlich gesenkt [WeLG98].

- **Multimedia, Multicast:** Beim „traditionellen“ IP-Multicast bleiben die Details der Vermittlungstopologie und die Anzahl der Empfänger vor dem Benutzer verdeckt. Dies

macht für unzuverlässigen Multicast auch Sinn. Bei einem zuverlässigen Dienst kann das aber ein Problem darstellen.

Zur Entlastung des Netzwerkes ist es sinnvoll, Übertragungswiederholungen lokal von benachbarten Empfängern oder von Zwischenspeichern in den Netzknoten anzufordern. Hier besteht einer der Hauptvorteile von programmierbaren Netzen, da diese die nötige Flexibilität bieten, um dieses Verfahren zu implementieren. Die Verantwortung wird dabei über den Multicastbaum verteilt.

Bei der Videokodierung, z.B. MPEG, bestehen auch Abhängigkeiten zwischen den einzelnen Rahmen. Hier ist es wichtig, daß im Netzwerk die nötigen Informationen über die zu transportierenden Daten vorhanden sind, so daß, falls Pakete verworfen werden müssen, nicht ausgerechnet die wichtigen Basisrahmen verworfen werden.

Bei Stau oder langsamen Übertragungsleitungen kann z.B. bei Videokonferenzen die Qualität und damit die Datenrate reduziert werden, auch hierfür wird Rechenleistung innerhalb des Netzwerkes benötigt.

- **Mobile IP/schnurlose Übertragung:** Mobile IP ist dafür ausgelegt, mobile Benutzer zu unterstützen, die sich je nach Aufenthaltsort mit ihrem Notebook an unterschiedlichen Orten an das Internet anschließen. Dabei soll dies für andere Benutzer transparent bleiben. Das Netz soll also alle Daten automatisch an den neuen Standort weiterleiten.

  Da diese Geräte oft auch noch schnurlos an das Netz angebunden werden, entsteht ein weiteres Problem. Funkstrecken stellen verhältnismäßig fehlerhafte und langsame Übertragungsstrecken dar. Herkömmliche Transportprotokolle, die eine Ende-zu-Ende-Kontrolle durchführen, haben mit einer Verkettung von Übertragungsstrecken mit unterschiedlichen Charakteristiken oftmals ein Problem. Programmierbare Netzwerke können ein lokales, adaptives System bieten, das die speziellen Eigenschaften einer Funkstrecke beachtet und eventuell Daten mit Vorwärtsfehlerkorrektur (engl. Forward-Error-Correction) redundant kodiert, so daß, auch wenn nur ein Teil der Daten beim Empfänger ankommt, die ursprünglich zu übertragenden Daten wieder vollständig hergestellt werden können.

Im Zusammenhang mit programmierbaren Netzen muß man den Begriff der *Leistungsfähigkeit* neu überdenken. Bei Rechnernetzen versteht man darunter traditionell immer übertragungsspezifische Größen, wie z.B. die Datenübertragungsrate oder die Verzögerungszeit. In programmierbaren Netzen macht dies allerdings keinen Sinn, da die Menge der übertragenen Daten ja gerade durch die Verarbeitung reduziert werden soll. Es ist vielmehr sinnvoll einen Leistungsbegriff zu definieren, der mißt, was für den Benutzer bzw. die Netzwerkendanwendung sichtbar und wichtig ist. So bieten sich z.B. bei der Online-Auktion als Leistungsmaße an: Anzahl der Gebote, die pro Sekunde bearbeitet werden oder die tatsächlich gültigen neuen Höchstgebote pro Sekunde.

## 1.3 Mögliche Probleme

Durch die größere Flexibilität und vor allem die weitergehenden Möglichkeiten des Benutzers, auf das Netzwerk direkt einzuwirken, entstehen auch beträchtliche Probleme. So ist die Frage, inwieweit sich die Benutzer dadurch gegenseitig gewollt oder ungewollt beeinflussen können oder was beim Auftreten von fehlerhaften Netzwerkkomponenten passiert. Diese *Sicherheitsaspekte* werden im folgenden Kapitel (in Abschnitt 2.3) ausführlicher behandelt. Es müssen Mechanismen für die Warteschlangen und Richtlinien für die Zuteilung der Ressourcen gefunden werden, um mit den kombinierten Anforderungen an Rechenleistung und Übertragungsbandbreite der Ausgangsleitungen umzugehen.

Es besteht ein deutlich *größerer Bedarf an Rechenleistung* in den aktiven Netzknoten, da diese nicht nur bestimmen müssen, über welche Ausgangsleitung ein Datenpaket weitergeleitet werden soll, sie können unter Umständen auch noch beliebig komplexe andere Berechnungen ausführen.

Weitere Schwierigkeiten entstehen dadurch, daß in dieser potentiell sehr *heterogenen Umgebung* trotzdem sichergestellt werden muß, daß die Daten ihren Zielpunkt erreichen und die einzelnen Netzteile miteinander interagieren können. Hier ist es besonders kritisch, wenn selbst die Wegewahlverfahren der einzelnen Pakete und Netzknoten frei wählbar oder sogar frei programmierbar sind. Um die Funktion des Netzwerkes zu garantieren, muß die Flexibilität des Netzwerkprogrammierers in gewisser Weise eingeschränkt werden.

Ein weiteres Problem sind *längere Verzögerungszeiten*, da für jedes Paket in jedem Knoten eventuell noch Berechnungen und Datenverarbeitungsschritte ausgeführt werden müssen, bis klar ist, ob und wohin welche Pakete weitergeleitet werden sollen. In den höheren Schichten erreicht man dafür unter Umständen aber verbesserte Anwendungsverzögerungszeiten, d.h. für den Endbenutzer liegen die relevanten Daten im Endeffekt schneller vor, da die Informationen kürzere oder geschicktere Wege gehen.

Wenn man spezielle komplexe Funktionen in ein Netzwerk implizit einbaut, optimiert man es für bestimmte Funktionalitäten, während man vielleicht dabei die Kosten für andere wertvolle Dienste, die zur Entwurfszeit noch nicht bekannt oder voraussehbar sind, beträchtlich vergrößert. Es bleibt also zu klären, ob man ein programmierbares Netzwerk als stark spezialisiert und optimiert für besondere, komplexe Netzdienste anzusehen hat – oder man es im Gegenteil aufgrund seiner großen Flexibilität als den geforderten allgemeinen Ansatz sieht, der keine konkreten Annahmen über die Netzanwendungen macht, sondern dies dem Netzbenutzer frei läßt.

# 2 Ansätze und Architekturkonzepte

## 2.1 Kapseln/Switchlets

Beim Entwurf eines aktiven Netzwerkes hat man einen großen Spielraum, wie weitgehend der Benutzer das Verhalten des Netzwerks direkt beeinflussen kann. Schickt er Programme direkt in den Datenpaketen mit oder legt er Programme in den Netzknoten ab, die von nachfolgenden Paketen aufgerufen werden? Kann er Zustandsinformation in Netzknoten ablegen, so daß Informationen im Netzwerk selbst direkt gespeichert werden, auf die verschiedene Datenpakete zugreifen können? Wie wird die Verteilung und der Transport von Programmcode zu den Netzknoten durchgeführt?

Bei einem Ansatz, der am MIT entwickelt wurde (siehe Abschnitt 3.1), wird der Programmcode direkt im Datenpaket mitgeschickt. Diese aktiven Pakete werden dann *Kapseln (Capsules)* genannt. Beim SwitchWare-Projekt der University of Pennsylvania dagegen (siehe Abschnitt 3.2) kann Programmcode in den Netzwerkknoten installiert werden, der dann von den nachfolgenden Paketen aufgerufen wird. Doch dazu mehr in den entsprechenden Abschnitten.

Eine neue Systemkomponente bei programmierbaren Netzen ist ein Mechanismus, um Programmcode zu transportieren und zu installieren. Dies muß schnell und effizient passieren und hier ist ein besonderes Augenmerk auf den Sicherheitsaspekt zu richten.

## 2.2 Programmiersprachen, API

Ein Herzstück der programmierbaren Netze ist deren *Anwendungsprogrammierschnittstelle (Netzwerk-API, engl. Network Application Programming Interface)*. Hier werden Aspekte fest-

gelegt, die zentral für die Flexibilität der Programmierung und damit auch für die Sicherheit des Gesamtsystems, sowie für dessen Leistungsfähigkeit sind.

Man kann entweder eine *virtuelle Maschine (engl. Virtual Machine)* definieren, die eine abstrakte Plattform bietet, unabhängig vom Rechnersystem auf dem das Programm ausgeführt wird (Bsp. Java Virtual Machine) oder man kann die Programme direkt in einer *maschinenspezifischen Sprache (engl. Native Code)* über das Netz schicken. Eine virtuelle Maschine hat den Nachteil, daß das Programm nicht direkt ausgeführt werden kann, sondern immer erst zur Laufzeit interpretiert wird, was zu erheblichen Leistungseinbußen führt. Maschinenspezifischer Code wirft dagegen vor allem in heterogenen Systemen das Problem auf, daß jedes Programm in der gleichen Version für verschiedene Plattformen im Netz vorhanden sein muß, was zu einem großen Mehraufwand bei der Programmierung und der Verteilung des Programmcodes führt.

Ein anderes wichtiges Kriterium ist die *Mächtigkeit der Programmiersprache*. Dabei reicht das Spektrum von einer einfachen Liste von festgelegten Parametern, die aus einer vordefinierten Menge von Möglichkeiten auswählen, bis zu einer Turing-mächtigen Programmiersprache, die jede beliebige Berechnung durchführen kann.

Der Vorteil einer sehr einfachen Programmiersprache liegt darin, daß sie sehr effizient implementierbar und optimierbar ist. Durch die Einschränkungen des Knotenverhaltens kann leichter geprüft werden, ob ein Programm korrekt ist, und seine Auswirkungen auf das Netzwerk sind beschränkt.

Eine Turing-mächtige Programmiersprache läßt dem Entwickler dagegen viel mehr Freiraum. In diesem Fall läßt sich eventuell auch eine bereits vorhandene Hochsprache, z.B. Java, verwenden. Es kann aber sehr schwierig werden, zu prüfen, ob ein Programm korrekt ist und ob es terminiert.

Es besteht auch die Möglichkeit, eine spezielle Sprache zu entwerfen, die bestimmte, gewünschte Eigenschaften, z.B. Terminierung und Erhaltung der Sicherheit des aktiven Knoten, inhärent garantiert.

Ein weiterer Aspekt ist die *Granularität der Kontrolle*. Dabei gibt es zwei Extreme: ein Paket ändert das Verhalten eines Knoten für alle nachfolgenden Pakete oder ein Paket ändert nur das Verhalten des Knotens in bezug auf sich selbst. Dazwischen gibt es die Möglichkeit, daß ein Paket sich auf einen Teil der folgenden Pakete auswirkt, dies ist z.B. bei sogenannten *Flows* der Fall.

## 2.3 Sicherheit

Bei der (System-)Sicherheit unterscheidet man im allgemeinen *zwei Zuverlässigkeitseigenschaften*: zum einen, ob ein System gegen Fehler durch vertrauenswürdige Benutzer geschützt ist (engl. *Safety*) oder zum anderen, ob ein System sicher vor Fehlern ist, die von nicht vertrauenswürdigen Benutzer veranlaßt werden, d.h. ob „Angriffe“ von außen vereitelt werden (engl. *Security*) [AAKS98].

Bei den herkömmlichen Netzwerkmodellen setzt ein (Anwendungs-)Dienst an den Endpunkten auf einem vorhandenen minimalen Netzwerkdienst auf (Service Overlay Model). Damit sind Sicherheitsfragen an die Netzwerkgrenzen verlegt. Bei programmierbaren Netzwerken ist dies eine Aufgabe der Netzwerkinfrastruktur. Unter der Netzwerkinfrastruktur sind die Systeme zu verstehen, die der Datenübertragung und Paketvermittlung dienen (Zwischensysteme), im Gegensatz zu den Rechnern, die als Endgeräte die Schnittstelle zum Benutzer bilden und solchen, die Dienste z.B. als WWW-Server zur Verfügung stellen.

Prinzipiell muß man die Daten und Programme von verschiedenen Benutzern oder Anwendungsprotokollen trennen, damit sie sich nicht gegenseitig beeinflussen können.

Im SANE-Projekt der University of Pennsylvania (siehe Abschnitt 3.2) wird die Netzknotensicherheit der Netzsicherheit gegenüber gestellt. Dabei wird festgestellt, daß die Knotensicherheit Grundlage für ein sicheres Netzwerk ist, aber allein nicht ausreicht, um die Netzsicherheit zu gewährleisten. Es ist zu prüfen, ob das System als Ganzes sicher ist. Eine Voraussetzung für ein stabiles, sicheres Netzwerk ist auch, daß die Algorithmen, z.B. für die Wegewahl, in „kurzer“ Zeit terminieren, damit gesichert ist, daß die Rechenleistung, die ein Prozeß belegt, wieder für andere Nutzer verfügbar wird.

Eine Möglichkeit ein Netzwerk vor einer Überflutung von ziellos umherkreisenden Paketen zu schützen, ist ein Time-To-Live-Feld (TTL-Feld) nach dem IP-Vorbild. Damit wird sichergestellt, daß ein Paket nach einer gewissen Zeit (oder in diesem Fall eigentlich nach einer gewissen Anzahl Übertragungsabschnitte) verworfen wird und nicht endlos im Netz kreist. Man muß auch verhindern, daß z.B. ein Multicastprotokoll, das neue Pakete im Netz erzeugt, nicht beliebig immer neue Pakete generieren kann. Eine Möglichkeit wäre hier sicherzustellen, daß ein Paket nur Pakete mit kleineren TTL-Feldern erzeugen kann.

## 2.4 Folgerungen aus den Ende-zu-Ende-Argumenten

Die *Ende-zu-Ende-Argumente (engl. End-To-End Arguments)* sind ein Entwurfsprinzip für die Organisation und Plazierung von Funktionalitäten in einem System. Sie geben allerdings nur eine Struktur des Entwurfsraumes vor und lösen das eigentliche Entwurfsproblem nicht.

Das zugrundeliegende Prinzip lautet folgendermaßen: *eine Funktion oder ein Dienst sollte nur dann in einem Teilsystem implementiert werden, wenn sie bzw. er dort vollständig implementiert werden kann oder wenn man dadurch, daß man sie bzw. ihn zum Teil in diesem System implementiert, die Leistung von anderen Teilen beträchtlich verbessert* [BCZP+98]. Für geschichtete Netzwerksysteme heißt das im speziellen, daß eine Funktion, die von einer Anwendung verwendet wird, auch in der Nähe der Anwendung implementiert werden sollte, d.h. in einem geschichteten System sollte sie sich möglichst weit „oben“ befinden [BCZP+98].

Ein Netzwerk sollte also nicht Funktionen bieten, die besser in den Endsystemen implementiert werden können. Manche Dienste benötigen sogar das Wissen und die Hilfe der Endsystemanwendungen oder der Benutzer und können daher nicht im Netzwerkinneren implementiert werden. Wenn nicht alle Anwendungen einen Dienst benutzen, sollte er so implementiert sein, daß auch nur die Anwendungen dafür bezahlen müssen, die ihn benutzen, d.h. daß z.B. Anwendungen, die keinen zuverlässigen Netzwerkdienst benötigen, nicht dafür bezahlen müssen.

Für manche Dienste ist es aber auch durchaus sinnvoll, sie im Netzinneren zu implementieren, da man sie so am besten unterstützen oder stark verbessern kann. Dort hat man Informationen zur Verfügung, z.B. wo und wann Stau auftritt oder der Ort von Paketverlusten in einem Multicastbaum, auf die man nur im „Netzwerkinnern“ direkt zugreifen kann. Bei der Unterstützung eines Ende-zu-Ende-Dienstes im Netzwerk müssen die Kosten der Implementierung und der Leistungsgewinn für das Anwendungsprogramm gegeneinander aufgerechnet werden.

Programmierbare Netzwerke sind nicht mehr monolithisch aufgebaut, sondern können viele verschiedene Dienste parallel unterstützen. Dies verursacht Kosten für die Bereitstellung und Benutzung, die von der Verbesserung der Ende-zu-Ende-Leistung übertroffen werden müssen.

### 2.4.1 Programmierbare Netzwerke und die Vermittlungsschicht

Man kann sich in den verschiedenen Schichten, z.B. Anwendungs- oder Transportschicht, diverse sinnvolle Einsatzmöglichkeiten für programmierbare Netzwerke vorstellen, aber die Vermittlungsschicht stellt sich hier als eine Ausnahme dar. Die Vermittlungsschicht sollte eine universelle Verbindungs- und Kommunikationsmöglichkeit zwischen vielen beliebigen und unterschiedlichen Computersystemen herstellen.

Wenn man in dieser Schicht ebenfalls eine Programmierung durch den Benutzer erlaubt, wird daher bei fehlerhaftem Code in einem Paket die Chance verringert, daß das Paket sein Ziel erreicht – eventuell genügt auch schon ein Fehler in einem vorangegangenen Paket. Programme haben üblicherweise Fehler bzw. können fehlerhaft sein; auch die Ausführungsumgebungen in den Netzknoten sind teilweise schlecht implementiert oder nicht mehr auf dem neuesten Stand. Wenn ein Fehler auftritt, hat nur der Programmierer das Wissen, um das Stückwerk aus verschiedenen Programmen und Daten zu untersuchen und herauszufinden wo der Fehler liegt. Mit den größeren Einwirkungsmöglichkeiten bei programmierbaren Netzwerken ergibt sich hier auch ein deutlich größerer möglicher Schaden, welcher auch noch schwerer zu beheben ist, als in traditionellen Netzen.

Dabei ist die Aufgabe der Vermittlungsschicht eigentlich ein einfacher Prozeß: ein Vermittlungsknoten hat drei Möglichkeiten: das Paket weiterschicken, verzögern oder verwerfen. Programmierbare Netzwerke können nur die Flexibilität erhöhen, zwischen diesen drei Möglichkeiten zu wählen, und damit auch das Risiko steigern, die falsche Entscheidung zu treffen [BCZP+98].

### 2.4.2 Anderer Blickwinkel auf die Ende-zu-Ende-Argumente

Man kann die Ende-zu-Ende-Argumente auch anders sehen: es ist die Frage „*Wer stellt das Programm zur Verfügung?*“ und nicht: „*Wo wird es ausgeführt?*“

Wenn man Funktionen und Dienste in einem geschichteten System nach oben verschiebt, also näher zu den Anwendungen, die sie benutzen, vergrößert man die Flexibilität des Anwendungsentwicklers, diese Funktionen und Dienste an die speziellen Wünsche und Bedürfnisse der Anwendung anzupassen. Mit dieser Sicht bedeutet die Programmierbarkeit einer unterliegenden Schicht, daß Entwurfsfreiheiten im Schichtenmodell aufwärts, d.h. näher zur Anwendung und zeitlich nach hinten verschoben werden, obwohl die resultierenden Aktionen tatsächlich irgendwo tief im Inneren des Netzwerks stattfinden [BCZP+98].

# 3 Aktuelle Projekte

## 3.1 Die ANTS-Architektur

Die ANTS-Architektur (Active Network Transport System), die am MIT entwickelt wurde, baut auf dem Konzept der Kapseln auf. Diese Kapseln ersetzen die herkömmlichen Datenpakete und enthalten außer Nutzdaten auch noch Programme. Die Programmcodeverteilung und der Datentransfer sind gekoppelt und können sich überlappen, was zu einem guten und schnellen Verhalten beim Übertragungsbeginn führt. Kapseln können auch Zustandsinformationen in Knoten ablegen und Klassen aufrufen, die von anderen Kapseln installiert wurden.

ANTS ist in Java implementiert und stellt eine Klasse Kapsel (engl. Capsule) sowie eine Klasse Protokoll (engl. Protocol) zur Verfügung, die dann abgeleitet werden. Dabei werden

Kapseltypen, die auf die gleichen Informationen im Netzwerk zugreifen, in Protokolle gruppiert. Ein Protokoll stellt einen Dienst bereit und ist die Basiseinheit der Netzwerkanpassung und des Netzwerkschutzes.

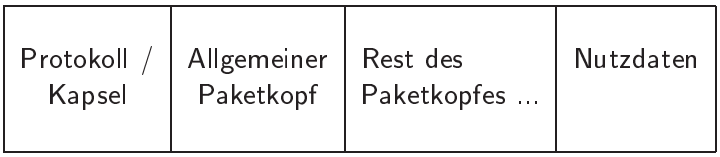


Abbildung 3: Kapselformat

Das ANTS-Kapselformat ist kompatibel zum IP-Paketformat, so daß es möglich ist, IP-Vermittlungsknoten und aktive Netzknoten in einem Netzwerk zu kombinieren.

Ein Protokoll wird über einen Code-Identifikator referenziert. Dieser Identifikator ist ein Fingerabdruck des Protokollprogrammcodes, der mit der kryptografischen Hashfunktion MD5 erzeugt wird. Dadurch besteht weniger Gefahr durch Protokollschnüffler und die Identifizierung von Programmcode kann dezentralisiert und schnell durchgeführt werden. Als Schutzbasiseinheit wird das Protokoll gewählt, das heißt, daß die Daten und der Programmcode eines Protokolls vor Benutzern eines anderen Protokolls geschützt sind [WeLG98, MIT99].

## 3.2 Die SwitchWare-Architektur

Die *SwitchWare*-Architektur der University of Pennsylvania zeichnet sich durch ihr geschichtetes Modell aus. Es besteht aus drei Schichten: den *aktiven Paketen*, den *aktiven Erweiterungen* und einer *sicheren Infrastruktur von aktiven Vermittlungsknoten* (siehe Abb. 4) [Alex98, Penn99].

Auch hier werden herkömmliche Datenpakete durch Pakete ersetzt, die zusätzlich Programmcode enthalten. Beim SwitchWare-Projekt werden diese Pakete *aktive Pakete (engl. Active Packets)* genannt und als Programmiersprache dient *PLAN (Programming Language for Active Networks)*. PLAN ist eine einfache Skriptsprache, die lediglich ein paar Basisprimitive und deren Verkettung unterstützt. PLAN-Programme sind streng typisiert und bieten eine minimale Funktionalität, da PLAN entworfen wurde, um aufwendige Überprüfungen zu vermeiden und unnötig zu machen.

Aktive Pakete können aber ihre Möglichkeiten und ihren Wirkungsbereich erweitern, indem sie Programme aufrufen, die in den Netzknoten gespeichert sind, sogenannte *aktive Erweiterungen (engl. Active Extensions)*. Solche aktiven Erweiterungen können dynamisch geladen werden oder zur Grundfunktionalität eines Vermittlungsknotens gehören. Aktive Erweiterungen haben weitergehenden Zugriff auf die Ressourcen und können größer und aufwendiger gestaltet sein als aktive Pakete. Deshalb kann man sie auch in einer allgemein gebräuchlichen Hochsprache kodieren.

Eine Implementierung von diesen aktiven Erweiterungen heißt *Switchlet*. Es wurde in einem Experiment gezeigt, daß ein einfacher Switchlet-basierter, gepufferter Repeater über das Netzwerk zu einer Bridge erweitert werden kann. Dabei wird das Repeater-Switchlet mit einem lernfähigen Bridge-Switchlet ergänzt. Diese aktive Bridge wurde in Caml programmiert. Caml ist wie Java maschinenunabhängig, aber effizienter als jedes derzeitige Java-System.

Die unterste Schicht der SwitchWare-Architektur bildet die *sichere aktive Vermittlungsinfrastruktur (engl. Secure Active Network Environment – SANE)*. SANE setzt seine Sicherheitsarchitektur auf einer minimalen Menge von Systemelementen auf, die als sicher vorausgesetzt werden. Darauf wird eine Integritätskette mit kryptografischen Hashes aufgebaut. Es wird

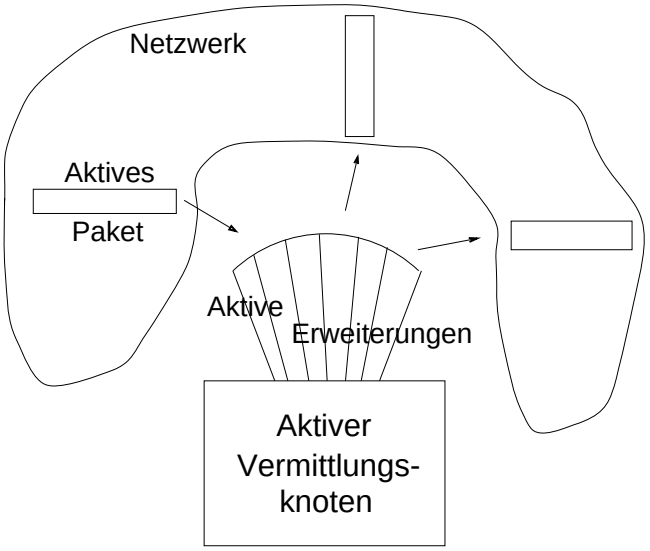


Abbildung 4: SwitchWare

erst die Integrität einer Schicht überprüft, bevor dann die Kontrolle an sie übergeben wird [AAKS98].

### 3.3 Hochleistungsvermittlungsknoten für programmierbare Netze

Da in einem programmierbaren Netzwerk sehr viel höhere Anforderungen an die Rechenleistung eines Vermittlungsknotens gestellt werden, ist eine Projektgruppe der ETH in Zürich und der Washington University in St. Louis dabei, einen skalierbaren Hochleistungsvermittlungsknoten für programmierbare Netzwerke zu entwickeln. Dies ist für den praktischen Erfolg von programmierbaren Netzwerken eine große Herausforderung.

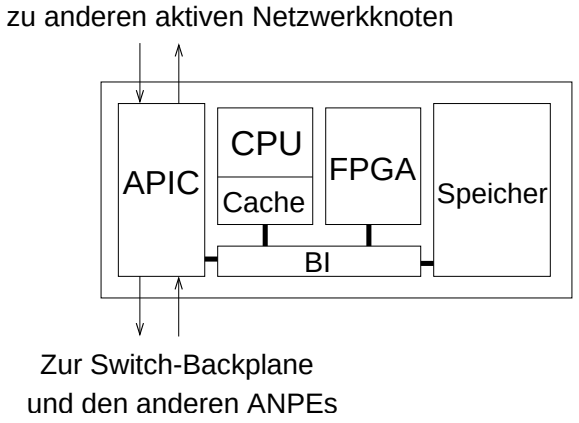


Abbildung 5: Active Network Processing Element

Die Rechenlast dynamisch verteilt auf eine Vielzahl von *ANPEs (Active Network Processing Elements)*, einer Kombinationen aus einer Standard-CPU, einem FPGA (Field Programmable Gate Array), Arbeitsspeicher und Kontrollelementen (APIC, BI). Die CPU übernimmt den größten Teil der aktiven Funktionalitäten und der FPGA implementiert besonders leistungskritische Funktionen in Hardware (siehe Abb. 5).

Es wird zwischen dem Betriebssystem eines aktiven Knoten (engl. *NodeOS*) und den Ausführungsumgebungen (engl. *Execution Environments* – EEs) unterschieden. Dabei stellt das

NodeOS grundlegende, implementierungsunabhängige Betriebssystemkomponenten zur Verfügung, auf welche die EEs aufbauen und protokollspezifische Funktionen implementieren. Die Softwarearchitektur besteht aus Grundelementen wie dem *Selector Dispatcher*, *Packet Classifier* und dem *Packet Scheduler*, sowie aktiven Plugins, die dann die anwendungsspezifischen Netzwerkfunktionen implementieren.

Ein Beispiel für eine Execution Environment (EE) ist *DAN (Distributed Code Caching for Active Networks)*, eine Form des verteilten Programmcodespeicherns. Ein aktiver Netzknoten fordert benötigten Programmcode bei Bedarf von einem Codeserver an und legt ihn dann im lokalen Speicher ab. Dabei werden die Programme mit einer digitalen Signatur gesichert. Es werden auch Optimierungen vorgenommen, um die Zeit zu minimieren, die gebraucht wird, bis der Code an den einzelnen Netzknoten verfügbar ist [DePl99, Wash99].

## 3.4 Tempest

Im Cambridge Computer Laboratory wird an einem Projekt namens *Tempest* gearbeitet. Tempest ist ein Rahmenwerk für sichere programmierbare Netzwerke, bei dem Garantien für die Ressourcenaufteilung gegeben werden können. Dabei geht es vor allem um die Frage, wie verschiedene solcher neuer Systeme nebeneinander und zusammen mit den herkömmlichen koexistieren können.

Das grundlegende Prinzip des Tempest-Projekt ist es, das Netzwerk in verschiedene virtuelle Netzwerke aufzuteilen. Der Kontroll- und der Nutzdatenfluß sind streng getrennt. Jedem privaten *virtuellen Netz (engl. Virtual Private Network, VPN)* wird ein definierter Anteil der Netzwerkressourcen zugeteilt. Wie diese Ressourcen dann benutzt werden, liegt in der Kontrolle des VPN. Dabei werden die Switchressourcen in mehrere Teile aufgeteilt, damit diese von verschiedenen Kontrollsystemen benutzt werden können. Diese Switchteile werden *Switchlets* genannt und erscheinen für Kontrollprogramme dann wie ein kompletter Switch.

Tempest erlaubt also die Koexistenz von vielen VPNs, die von verschiedenen Kontrollarchitekturen gesteuert werden, in einem einzigen physikalischen Netzwerk. Daraus ergibt sich zunächst als Granularität der Programmierbarkeit das VPN. Da aber die einzelnen Kontrollarchitekturen selbst programmierbar sein können, erlaubt Tempest auch eine feingranulare, anwendungsspezifische Programmierung.

Eine wichtige Systemkomponente ist der *Prospero Switch Divider*. Dieser teilt den einzelnen Switchlets Systemressourcen zu und sorgt dafür, daß die festgelegten Werte eingehalten werden. Es wurden Schnittstellen zwischen diesem Divider und dem physikalischen Switch (*Ariel*) definiert, sowie zwischen dem „Restsystem“ und der gesamten Kontrollarchitektur (*Caliban*), um sicherzustellen, daß Tempest unabhängig von den eingesetzten Switches und der zugrundeliegenden Netzmanagementstruktur ist.

Durch die Aufteilung in VPNs stellt Tempest eine sichere Umgebung bereit, in welcher Programme von Drittanbietern oder sogar dynamisch geladener Code ausgeführt werden kann, ohne andere Benutzer zu schädigen [RMCL98].

## 3.5 Weitere Projekte

- Die Forschungsprojekte *Liquid Software* und *Scout* der *University of Arizona* machen eine feinere Granularität des lokalen Ressourcenmanagements möglich. Durch die Entwicklung eines spezialisierten Knotenbetriebssystems (*Scout*) wird eine große Leistungsfähigkeit erreicht. Darauf setzt eine Implementierung der virtuellen Maschine von Java auf (*Joust*), die sowohl eine Laufzeitumgebung als auch einen Just-in-Time-Compiler

enthält. Alle festgelegten Komponenten sind in C oder Java geschrieben und werden schon vor der Laufzeit in Maschinencode übersetzt. Die Joust/Scout-Implementierung von ANTS ist zwei- bis dreimal schneller als eine Implementierung, die das JDK von Sun und ein Standardbetriebssystem wie z.B. Linux benutzt.

- In einem Projekt der *Columbia University* wurde NetScript, eine Programmiersprache und Laufzeitumgebung, entwickelt. Mit der Programmiersprache kann man Paketströme mittels eines Skripts bearbeiten. Das Projekt ist speziell für die Implementierung von Vermittlungsabläufen, der Analyse von Paketen und Managementfunktionen gedacht. Netscript-Agenten können auch zu entfernten Systemen gesendet werden.

  Eine andere Gruppe an der Columbia University untersucht Architekturen für Open Signaling, mit dem Schwerpunkt auf Verbindungsmanagement und Dienstgüteunterstützung in ATM. Die Xbind-Implementierung basiert auf einer Knotenprogrammierschnittstelle, die eine Hochsprache (z.B. C) enthält und Kontrollprimitiven, die ein ATM-Switch-Hersteller zur Verfügung stellt. Die Granularität der Kontrolle ist die virtuelle Verbindung.

- *Xbind* entwickelt eine programmierbare Transportarchitektur mit Dienstgütegarantien. Die Architektur ist objektorientiert und erlaubt es, auf Anforderung eine Auswahl von Protokollstapeln dynamisch zusammenzubinden. Sie basiert auf einem Verbraucher/Erzeuger-Modell (engl. Consumer/Producer Model). Die Verbraucher/Erzeuger-Komponenten sind die Medienverarbeiter (engl. Media Processors) und die Medientransporter (engl. Media Transporters). Jede Komponente wird separat durch ihre Transportabstraktion, ihre Kontroll- und Managementabstraktion und einem Satz von Controllers repräsentiert, welche die Netzwerkdienste, wie z.B. das dynamische Binden von Protokollstapeln, implementieren [HuLa98].

- Forschungen am *Georgia Tech* untersuchen die Möglichkeiten zur Realisierung von Diensten innerhalb eines Netzwerkes, z.B. Staukontrolle oder Filter für MPEG-Videoströme. Dabei werden die Funktionen Out-of-Band, d.h. über einen separaten, von den Nutzdaten getrennten Zugang, in die Netzwerkknoten geladen. Es wurde auch gezeigt, daß selbstorganisierende Netzwerk-Caches mit Hilfe von programmierbaren Netzwerken realisiert werden können.

- Das SmartPackets-Projekt bei *BBN* benutzt die Technologie der aktiven Netzwerke, um mit dem wachsenden Problem des Netzwerkmanagements umzugehen. Es wurden 2 Programmiersprachen entwickelt: *Sprocket*, eine Hochsprache mit eingebauter Unterstützung für Netzwerkmanagement und *Spanner*, eine CISC-Assemblersprache, in die Sprocket übersetzt wird. Ein Entwurfsziel ist es, sinnvolle und nützliche Netzwerkmanagementprogramme sehr kompakt in weniger als einem Kilobyte zu codieren [BBN 99].

# 4 Schlußbetrachtung

Der Entwurf eines programmierbaren Netzwerks wirft viele Fragen auf. Es gibt verschiedene Ansätze, die ihre Vor- und Nachteile haben. So stellt sich die Frage, wie weit die Möglichkeiten des Benutzers, direkt in das Netz einzugreifen, reichen sollen und wie dabei trotzdem Effizienz und Sicherheit gewährleistet bleiben. Hier besteht noch Raum für Forschung – es wird sich aber erst in der Praxis zeigen, ob dieser neue Ansatz die alten Netzwerktechnologien ergänzen oder gar ersetzen kann.

Auch bei programmierbaren Netzwerken, die insbesondere eine Vielzahl von Diensten und Protokollsystemen auf dem selben physikalischen System ermöglichen sollen, ergibt sich ver-

stärkt das Problem der Heterogenität. Wie können verschiedene Netztypen miteinander interagieren und wie können die nötigen Schnittstellen standardisiert werden, ohne die Möglichkeiten und die Vielfalt zu sehr einzuschränken? Es wird auf jeden Fall eine längere Übergangszeit geben, so daß es wichtig ist, daß neue programmierbare Netze mit den „traditionellen“ koexistieren können.

## Literatur


[AAKS98] S. Alexander, W. Arbaugh, A. Keromytis und J. Smith. Safety and Security of Programmable Network Infrastructures. *IEEE Communications Magazine*, Oktober 1998.

[Alex98] S. Alexander. The SwitchWare Active Network Architecture. *IEEE Network*, Mai/Juni 1998.

[BBN 99] BBN Technologies. Smart Packets. http://www.net-tech.bbn.com/smtpkts/smtpkts-index.html, 1999.

[BCZP+98] S. Bhattacharjee, K. Calvert, E. Zegura, C. Partridge, T. Strayer, B. Schwartz, A. Jackson, D. Reed, J. Saltzer und D. Clark. Commentaries on 'Active Networking and End-to-End Arguments'. *IEEE Network*, Mai/Juni 1998.

[CBZS98] K. Calvert, S. Bhattacharjee, E. Zegura und J. Sterbenz. Directions in Active Networks. *IEEE Network*, Mai/Juni 1998.

[DePl99] J. Decasper und B. Plattner. A Scalable High-Performance Active Network Node. *IEEE Network*, Januar/Februar 1999.

[HuLa98] J.-F. Huard und A. Lazar. A Programmable Transport Architecture with QoS Guarantees. *IEEE Communications Magazine*, Oktober 1998.

[MIT99] MIT. ActiveWare. http://www.tns.lcs.mit.edu/activeware/, 1999.

[Orti98] Sixto Ortiz Jr. Active Networks: The Programmable Pipeline. *IEEE Computer Magazine*, August 1998.

[Penn99] University of Pennsylvania. SwitchWare. http://www.cis.upenn.edu/~switchware/, 1999.

[RMCL98] S. Rooney, J. van der Merwe, S. Crosby und I. Leslie. The Tempest: A Framework for Safe, Resource-Assured, Programmable Networks. *IEEE Communications Magazine*, Oktober 1998.

[Wash99] Washington University. Active Network Node. http://www.arl.wustl.edu/arl/projects/ann/ann.html, 1999.

[WeLG98] D. Wetherall, U. Legedaza und J. Guttag. Introducing New Internet Services: Why and How. *IEEE Network*, Mai/Juni 1998.

# Möglichkeiten des Cachings bei Audio- und Video-Datenströmen

Henning Dammer

**Kurzfassung**

Um Dienste wie *Video on Demand (VoD)* über bestehende Netzwerkstrukturen und Cache-Systeme nutzen zu können, müssen diese auf verschiedene Weise erweitert werden. Aktuelle Cache-Systeme berücksichtigen kontinuierliche Medien nur ungenügend und führen somit zu sehr hoher Netzlast, indem sie z.B. nur Unicast-Verbindungen erlauben. Ansätze zur dynamischen Selbstkonfiguration von Caches mit Hilfe von *Helpern* sowie der Implementierung von *Staggered VoD (S-VoD)* sind nur zwei Vorschläge, die versuchen, den Ansprüchen von heutigen multimedialen Anwendungen gerecht zu werden. Diese Ausarbeitung liefert neben einer kurzen Zusammenfassung über die Funktionsweise von statischen Caches Lösungsvorschläge für Implementierungen sowie Simulationsauswertungen einer bestehenden Testumgebung.

## 1 Einleitung

Mit der steigenden Popularität des *World Wide Web (WWW)* geht auch ein Wandel dieses Informationsmediums einher. Anfänglich nur dafür bestimmt, Textinhalte und vielleicht einfache Grafiken darzustellen, sind die Anforderungen an dieses Medium unaufhörlich gestiegen. Dienste, die heutzutage dem Benutzer angeboten werden sollen, waren anfangs undenkbar, und folglich hatte man auch keine Möglichkeiten der Implementierung vorgesehen. Die heutige Problematik steckt in der Entwicklung und Spezifizierung von Mechanismen, die einer Vielzahl von Anwendern moderne „Streaming"-Dienste anbieten können und auch den rasant wachsenden Bedürfnissen gerecht werden.

Als „kontinuierliche Medien" bezeichnet man Audio- , Video- und häufig auch Multimedia-Datenströme, wobei sich diese Ausarbeitung vornehmlich mit dem Realisierungsproblem am Beispiel von VoD befassen wird. Um diesen Dienst anbieten zu können, bedarf es einer Umgestaltung der globalen Caching-Systeme von statischen Caching zu dynamischen Caching-Methoden. Altbekannte Systeme, wie Harvest oder Squid, müssen mit neuen Mechanismen der Cache-Verwaltung erweitert werden, um langfristig die Netzlast auch bei „kontinuierlichen Medien" zu reduzieren und Wartezeiten bei Endanwendern erträglich zu gestalten.

An diesem Punkt setzt diese Ausarbeitung an und liefert umfassende Ideen zur sofortigen Implementierung als auch zur langfristigen Umgestaltung der „Netzwerklandschaft". Es werden neben drei verschiedenen Ansätzen zur Cache-Erweiterung von bestehenden Systemen auch Algorithmen und Modelle zur Cache-Verwaltung vorgestellt.

## 2 Statisches Caching

Nachdem 1993 die erste funktionierende Caching-Software (CERN Web Server) der Öffentlichkeit zugänglich gemacht wurde, ging man mit der Entwicklung des „Harvest" einen Schritt

weiter. Dieses System wurde mit dem Ziel entwickelt, möglichst effektiv Index-Informationen zu erstellen und zu verarbeiten. Zentrale Komponenten sind:

- *Gatherer*: Hat die Aufgabe, auf jenen Servern Indexinformationen zu sammeln, auf denen die Harvest-Software nicht läuft.
- *Broker*: Sammelt die Informationen von mehreren Gatherern, um daraus Indizes zu erstellen und sie über das Internet verfügbar zu machen.
- *Index/Search Subsystem*: Legt eine Schnittstelle zwischen Broker und Indizierung fest, um Suchanfragen, die von anderen Benutzern oder Brokern stammen, bearbeiten zu können. Dadurch wird es einem Broker ermöglicht, nur Teilmengen von Indextabellen eines anderen Brokers abzufragen. Hierzu wurden zwei verschiedene Subsysteme entwickelt: *Glimpse* und *Nebula*.
- *Replicator*: Dieser kopiert gesamte Verzeichnisbäume oder komplette Server. Er besteht aus dem *Flood-Daemon (Flood-d)*, der seine Informationen entlang eines von ihm selbst festgelegten und verwalteten Weges an andere Gruppen (Mirror-d) liefert, und dem *Mirror Daemon (Mirror-d)*, welcher in periodischen Abständen Statusmeldungen an seine Nachbarn in der Gruppe sendet, um neue Informationen zu entdecken, die ihm Flood-d nicht liefern konnte.
- *Object Cache*: Dieser hierachisch angeordnete Cache bietet die Möglichkeit, Baumstrukturen von zusammenarbeitenden Caches (Eltern- oder Nachbarbeziehungen) über große Entfernungen hinweg aufzubauen (vergleiche Abbildung 1).

  Gründe zur Implementierung waren neben dem bereits erwähnten Leistungsgewinn (geringere Antwortzeiten und Netzwerkauslastung) auch der Schutz vor ununterbrochenen Client-Anfragen nach bestimmten Objekten. Organisiert ist der Object Cache nach dem *Write-through-Prinzip* und wird über *Hash-Tabellen* angesprochen.

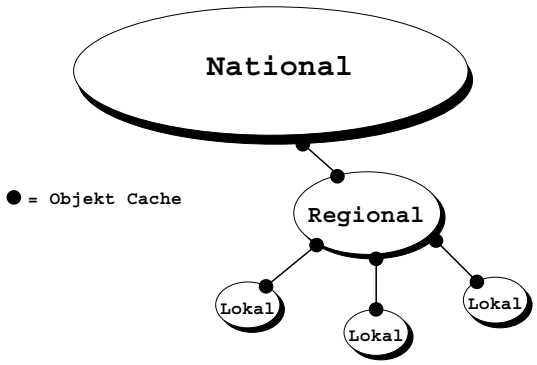


Abbildung 1: Hierachische Netzwerkstruktur

Die Kommunikation zwischen den Komponenten des Harvest-Systems erfolgt dabei über das *Summary Object Interchange Format (SOIF)*. Bei einer Objektanfrage schaut der Cache zuerst in seinem eigenen Speicher nach, ob er die Anfrage bedienen kann. Wenn dies nicht der Fall sein sollte, schickt er parallel Anfragen an seine Nachbarn und seinen Vater. Schließlich wird bei der Beantwortung der Anfrage dem Cache mit der geringsten Latenzzeit der Vorzug gegeben, wobei das Objekt, sobald die ersten Daten empfangen wurden, direkt an den anfragenden Client übermittelt wird.

Dadurch, daß die Cache-Hierarchien statisch konfiguriert werden, können Anfragen mit langen Wartezeiten verbunden sein. Die Suche nach Objekten kann dabei nur über ein Aufsteigen in

der Baumstruktur geschehen (siehe Abbildung 1). Dieser Nachteil wurde selbst bei der Weiterentwicklung des Systems – *Squid* – nicht beseitigt. Hier handelt es sich um ein kooperatives Caching, das jedoch auf Basis einer Konfigurationsdatei seine Umgebung erkennt.

# 3 Caching-Techniken für kontinuierliche Medien über das Internet

Da bestehende Caching-Verfahren – wie sie in dem vorangehendem Kapitel angesprochen wurden – keine Unterstützung zur Berücksichtigung von Objektgrößen und Ströme anbieten, würde eine kontinuierliche Medien Anfrage, wie z.B. VoD, zu einer langfristig gesehenen Überlastung des Netzes führen. Dies resultiert aus der Tatsache, das Harvest oder Squid Objekte immer in ihrer Gesamtheit zwischenspeichern. Bei Bildern oder Texten hat dieses Verfahren keinen offensichtlichen Nachteil, weil es sich hier um kleine Objekte handelt; doch sobald Filme übertragen werden sollen, ist es nicht möglich, dieses Verfahren auch hier zu realisieren, da der Cache-Speicher (CM) in seiner Größe nicht auf solche Datenmengen zugeschnitten ist. Neue Verfahren und Ideen müssen entwickelt werden.

Es gilt, Algorithmen zu finden, die das Verbraucherverhalten bei der Objektauswahl und während der Objektabfrage (bei Filmen wären dies das Zurück- oder Vorspulen, Pause oder Stop) möglichst identisch nachbilden, um somit schließlich die Netzlast durch gezieltes Bereitstellen der am häufigsten abgefragten Objekte zu reduzieren. Darüberhinaus muß neben Fragen der optimalen Cache-Größe und Anpassung auch über neue Kommunikationsmöglichkeiten der Caches untereinander nachgedacht werden. Aktuelle Forschungsergebnisse zeigen erste Ansätze zur Lösung des Problems aufbauend auf bestehenden Systemen, oder auf den Einführung von *Helpern* und vereinten Streaming-Architekturen.

## 3.1 Streaming-Erweiterungen für bestehende Caching-Systeme

Auf Basis von VoD-Netzwerken werden mögliche Architekturen, die zur Implementierung benutzt werden können, diskutiert. Die hierbei getroffenen Aussagen stützen sich auf die Ausarbeitung [DaMo98]. Die verschiedenen Möglichkeiten der Handhabung von Objekten in Abschnitt 3.1.3 sind der Literatur [HPGN+99] entnommen. Bei aktuell implementierten Systemen unterscheidet man zwischen einer zentralen (*Centralised Architecture*) und einer verteilten (*Distributed Architecture*) Netzarchitektur. Eine bereits bestehenden Netzwerk-Umgebung zu benutzen, bedeutet einerseits eine immense Kosteneinsparung, aber andererseits auch eine Einschränkung bei der Auswahl und der Qualität der angebotenen Dienste. In diesem Abschnitt werden, basierend auf der Annahme, daß ein ATM-Netzwerk[1] zur Verfügung steht (dieses verbindet den Video-Server (VS) mit den Benutzern), die verschiedenen Phasen bzw. die angebotenen Dienste entwickelt und analysiert (vergleiche Abbildung 3). Darüber hinaus beschäftigt er sich mit der Implementierung von Stromtechniken in bestehende Systeme.

### 3.1.1 Client-Server-Verbindungen

Am Beispiel von VoD, wo ein kontinuierlicher Datenstrom von 2 Mbit/s aufgrund einer MPEG1 Kodierung zur Verfügung steht, unterscheidet man zwischen zwei extremen Arten der Datenanlieferung vom Server an die Clients. Diese sind *Near VoD (N-VoD)* und *Interactive VoD (I-VoD)*. Vergleiche hierzu Abbildung 2.

[1] ATM ist die Übertragungstechnologie des *Breitband ISDN Netzwerk (B-ISDN)*.

*N-VoD* — Bei N-VoD werden Filme zu festen Zeitpunkten vom Server ausgestrahlt, wodurch Teilnehmer falls sie einen Film von Anfang an sehen möchten, verpflichtet sind, sich auf diese vordefinierten Zeitpunkte festzulegen.

Durch dieses Verfahren können sich mehrere Benutzer den gleichen Videostrom teilen. Neben dem Nachteil der Vorgabe fixer Zeitpunkte steht auch die mangelnde Flexibiliät der Lösung, da Benutzer nicht interaktiv (Pause, Vor-, Zurückspulen) eingreifen können. Dieses Verfahren reduziert zwar die Netzlast in hohem Maße, würde aber den Bedürfnissen der Anwender nur ungenügend gerecht werden.

*I-VoD* — I-VoD ist hingegen vollständig auf interaktive Möglichkeiten, von Seite des Benutzers aus gesehen, ausgerichtet. Jeder Teilnehmer erhält eine eigene Verbindung zum Server, wodurch er unabhängig von anderen Benutzern agieren kann. Diese Lösung bietet das höchste Maß der Flexibilität, ist aber aus Sicht des Providers nicht akzeptabel, da seine Server bei diesem Verfahren zu sehr belastet werden.

*S-VoD* — Um dennoch den Bedürfnissen von Benutzer und Anbieter zu entsprechen, kombinierte man beide Verfahren zu *Straggered VoD (S-VoD)*. Diese Implementierung basiert auf der Idee, daß viele Benutzer mit geringer Zeitverzögerung, bei der Anfrage an den Server, den gleichen Videostrom nutzen können. Dazu wird wie bei N-VoD zwischen Benutzer und Video-Server ein Switch geschaltet. Dieser ist jedoch mit Zwischenspeicher (Cache) ausgestattet, der in der in der Lage ist, bestimmte Segmente eines Films über festgelegte Zeitintervalle aufzunehmen. Durch diese Kompromißlösung kann einerseits die Belastung des Netzes signifikant reduziert, und andererseits die Interaktivität berücksichtigt werden.

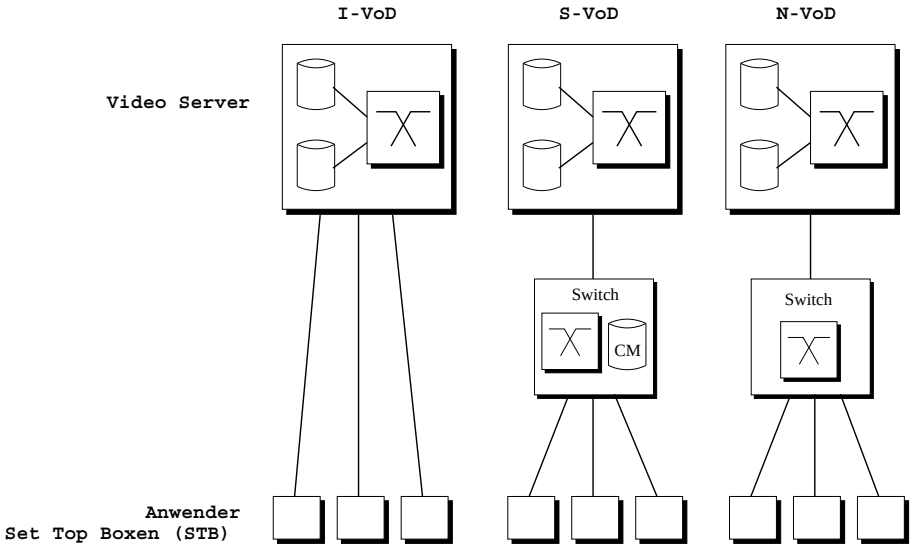


Abbildung 2: VoD-Service Architektur

### 3.1.2 Aufbau der Dienste

Basierend auf der Implementierung von S-VoD teilt man eine Serveranfrage in eine Initialisierungs- und Fortlaufende-Phase ein.

*Initialisierungsphase* — Sie beschreibt die ersten Minuten nach der Objektanfrage an den Server. Man unterscheidet dabei folgende Fälle:

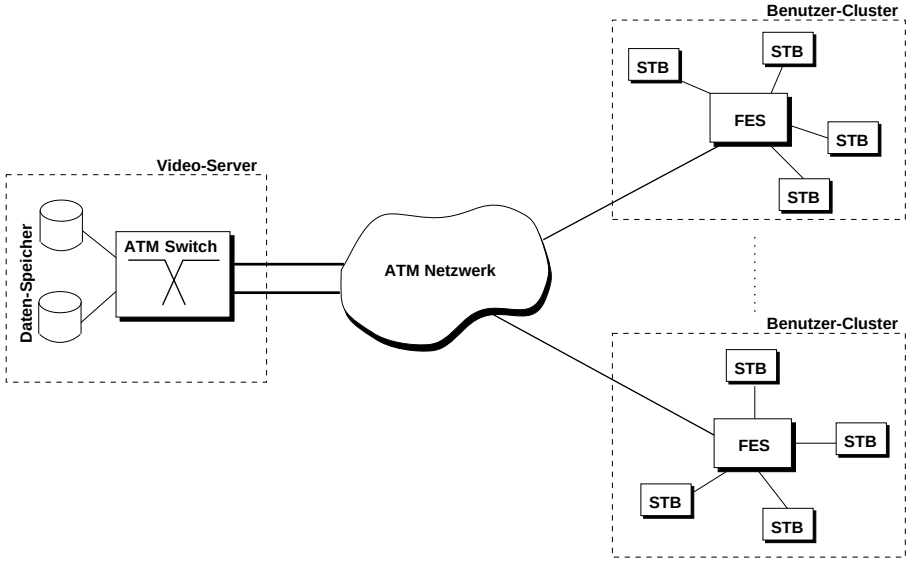


Abbildung 3: Referenzarchitektur des VoD Netzwerks

- Die Daten sind bereits in dem lokalen Cache Memory (CM) des *Front-End Switch (FES)* vorhanden. Somit können diese direkt, mit z.B. 2 Mbit/s, abgefragt werden. Dies nennt man auch *Cache Memory Primary Rate Transfer Schedule (CMPR).*
- Die Daten sind nicht im lokalen CM vorhanden. Man kann diese nun entweder schnell vom Server beziehen, falls der lokale CM über freien Speicher verfügt – dies nennt man *Video Server High Rate Transfer Schedule (VSHR)* – oder, falls kein Speicher frei ist, vom zentralen Video-Server durch den Aufbau einer Punkt-zu-Punkt-Verbindung (vgl. I-VoD) *Dedicated Server Primary Rate Transfer Schedule (DSHR).*

*Folgephase* — Nachdem die Initialisierung abgeschlossen ist, wird der CM mit konstanter Geschwindigkeit gefüllt und ausgelesen. Es ist einerseits möglich, sich einem bestehenden Videostrom eines anderen CM anzuschließen *(MCPR = Multiple Casting Primary Rate Transfer Schedule)*, oder es muß, falls dies nicht möglich ist, eine neue Verbindung zum Server hergestellt werden *(SSPR = Shared Server Primary Rate Transfer Schedule).*

*Kombinationen* — In der Praxis findet man häufig Kombinationen dieser vier Übertragungsverfahren. Abhängig von der vorgefundenen Umgebung wird CMPR und VSHR mit MCPR oder SSPR kombiniert. Folgende Beispiele sollen dies verständlich machen:

- Die gesamten benötigten Daten zu einem bestimmten Zeitpunkt sind im lokalen CM. Daraus folgt: nur CMPR ist am Abrufvorgang beteiligt.
- Teile der Daten sind im lokalen CM vorhanden (CMPR). Dieser hat noch genügend freien Speicher, um den Rest der Daten vom Server schnell anzufordern – vorausgesetzt Bandbreite steht zur Verfügung (VSHR). Der kontinuierliche Datenfluß wird über einen anderen CM aufrechterhalten, der die benötigten Daten zur Verfügung stellen kann (MCPR). Somit werden CMPR, VSHR und MCPR benutzt.
- Die gesamten Daten werden über eine Verbindung zwischen dem lokalen CM und dem zentralen Server übertragen. Die ersten Segmente werden dabei bei einer hohen und die folgenden bei einer niedrigen Bit-Rate übertragen. CMPR+VSHR+SSPR werden hierbei benutzt.

- Die gesamte Anfrage wird durch eine Verbindung, zwischen zentralem Server und Benutzer, bedient. Somit benötigt man in diesem Fall nur DSHR.

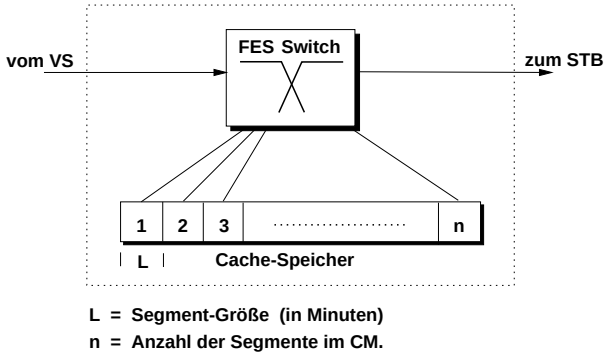


Abbildung 4: Front-End Switch (FES)

### 3.1.3 Cache- und Objekt-Verwaltung

Die Frage, wie kontinuierliche Medienobjekte zu handhaben sind, so daß ein CM sie aufnehmen kann, wurde bis jetzt noch nicht geklärt. In diesem Kapitel werden drei Techniken vorgestellt, mit deren Hilfe man kontinuierliche Medien verwalten kann.

*Segmentation von Objekten* — Wie bereits erwähnt, ist es aufgrund der Datenmenge nicht sinnvoll, ganze Filme in einen CM abzulegen. Eine Möglichkeit, dieses Problem zu umgehen, besteht in der Aufteilung des Objekts in kleinere Teilsequenzen (vergleiche Abbildung 4). Angenommen, die kleinste zuweisbare Segmentgröße im statischen Cache ist $S$, so beträgt die Größe von Teilobjekten immer ein Vielfaches von $S$. Diese werden im weiteren Verlauf als *Chunks* bezeichnet. Da die Größe immer $k * S$ sein muß, wobei k eine *Integer*-Zahl größer als 0 ist, treten die Chunk-Grenzen bei $k * S$, $2 * k * S$, etc. auf. Aufgrund der Aufteilung des Medienobjekts können Segmente auch unabhängig voneinander gespeichert oder überschrieben werden. Ein wichtiger Unterschied zwischen statischen und dynamischen Objekten liegt in der Tatsache, daß dynamische Segmente beim Ablegen in den CM stets einen Start- und Endzeitpunkt zugewiesen bekommen. Da es nun möglich ist, daß beim Abspielen des Films dieser auf verschiedene CM zurückgreift, die Teilsegemente halten, kann es aufgrund von verschiedenen Zugriffzeiten zu Lücken (Wartezeiten) beim Zusammensetzen kommen. Um dies zu reduzieren, erscheint es sinnvoll, daß ein CM aufeinanderfolgende Segmente eines Film hält, welche dann als *Chunk* bezeichnet werden und deren Größe ein Vielfaches von $S$ beträgt. Der Vorteil, der sich aus dem Zusammenlegen der Segmente ergibt, ist eine Begrenzung der Lückenhäufigkeit. Um aber das Auftreten generell zu minimieren, werden Chunks nach dem Prinzip *Least Recently Used (LRU)* und Segmente innerhalb durch *Prefix Caching*[2] organisiert. Da die Länge und Anzahl der Lücken von der Größe der *Chunks* abhängen, muß diese möglichst optimal angepaßt werden. Je größer sie bestimmt wird, desto eher blockiert der CM, und je kleiner sie ist, umso größer wird die Wahrscheinlichkeit, daß Lücken auftreten.

*Dynamisches Caching* — Um dem Charakter von kontinuierlichen Medien gerecht zu werden, wird der dynamische Cache eingeführt. Dieser soll den zeitlichen Unterschied bei der Anfrage von Objekten überbrücken – siehe auch S-VoD–, um somit mehreren Benutzern den gleichen Medienstrom zugänglich zu machen. Dabei basiert das dynamische Caching auf den

[2]Das letzte Segment eines Chunks wird immer zuerst überschrieben.

Techniken *Data Patching* und *Dynamische Ring-Puffer* (vergleiche Abbildung 5). Dies soll am folgenden Beispiel erklärt werden.

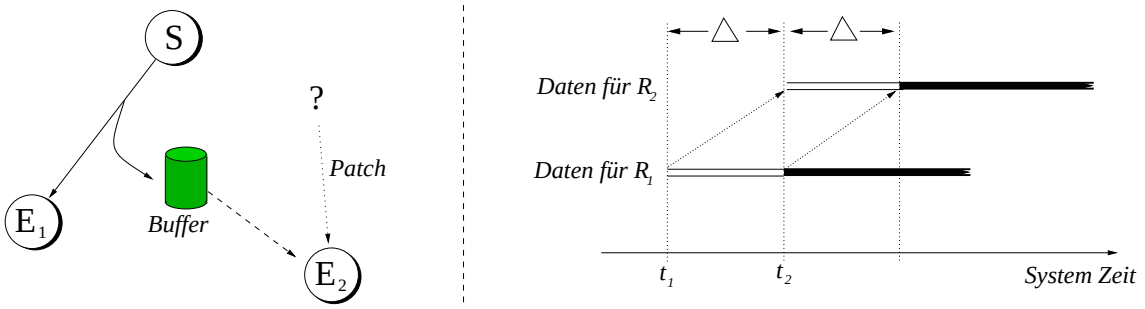


Abbildung 5: Dynamischer Cache

Empfänger $E_1$ stellt eine Objektanfrage an Server $S$ zum Zeitpunkt $t_1$. Kurze Zeit später stellt $E_2$ eine Anfrage an das gleiche Objekt ($t_2$). Da $E_1$ bereits die Daten erhalten hat, die nun $E_2$ nach weiteren $\Delta = t_2 - t_1$ Sekunden benötigt, lohnt es sich, einen Puffer einzusetzen, der in der Lage ist, diese Zeitdifferenz, die in der Größenordnung einiger Sekunden liegt, aufzunehmen. Dieser Speicher, der als *Ring-Puffer* organisiert nach dem FIFO-Verfahren arbeitet, überbrückt nun die zeitliche Distanz der beiden Anfragen. (Wo sich dieser befindet, ist im Moment weniger relevant; dies kann bei einem der Empfänger oder auf einem Host im Netz sein.) Nun ist es für $E_2$ möglich, über die ganze restliche Zeit hinweg aus dem Ring-Puffer bedient zu werden, und er muß nicht auf mehrere Caches zugreifen. $E_2$ fehlen aber noch die ersten Sekunden, die $E_1$ bereits erhalten hat, die sich aber nicht im Ring-Puffer befanden. Diese Datensegmente erhält $E_2$ nun entweder über einen statischen Puffer eines CM oder über den Ursprungs-Server.

Daten, die man nicht aus dem Ring-Puffer erhält, werden als *Patch* und die Technik als *Data Patching* bezeichnet.

*Dynamische Cache-Konfiguration* — In Kapitel 2 wurden statische Caching-Verfahren und Programme vorgestellt, die ausschließlich auf festen Baumhierachien basieren. Anfragen wurden stets an die gleichen, fest in einer Liste eingetragenen Nachbarn oder Eltern gesendet. Die Basis dafür bildete die Topologie der CM im Netz und nicht die Latenzzeit bei Anfragen. Da diese aufgrund von Netzwerkcharakteristiken und der Auslastung der CM variiert, dürfen diese Listen nicht statisch sein, sondern müssen ständig, nach bestimmten Algorithmen aktualisiert werden. Daraus resultierende, dynamisch aufgebaute Verknüpfungen der CMs bezeichnet man als *Meshes*. Es besteht nun die Möglichkeit, diese Listen zentral oder verteilt zu verwalten, wobei folgende Unterschiede vorhanden sind:

- *Zentrale Verwaltung*: Ein Vorteil ist die globale Sicht eines einzelnen CM über die gesamten *Meshes*, wodurch dieser die optimalen Beziehungen bestimmen kann. Jedoch ist die Aktualität der Liste bei großen Netzen schwer zu gewährleisten, da die Menge der Informationen, die allein zur Aktualisierung und Verteilung benötigt wird, bei steigender CM Anzahl exponentiell wächst.

- *Verteilte Verwaltung*: Hier sammelt jeder Cache-Proxy für sich Informationen über bestehende *Meshes* und wertet diese aus. Zwar kann diese Auswahl schlechter ausfallen, jedoch ist sie dafür nicht so komplex und bei großen Netzen auch wesentlich stabiler einzusetzen.

## 3.2 Vereinte Streaming Architektur - Helper

Alle Techniken des Abschnitts 3.1.3 bilden zusammen eine vereinte Architektur, die im folgenden *Dynamic Cooperative Caching Architecture (DCCA)* genannt wird. Das zentrale Element dieser Architektur bildet der *Helper*. Dieser ist in der Lage, Daten zwischenzuspeichern und zu verteilen. Er befindet sich, wie eine Art „Agent" im Netz, mischt sich aber nicht in Routing-Protokolle ein. Somit bedarf seine Implementierung keiner Umstellung bestehender Netzwerkkomponenten. *Helper* bedienen Benutzeranfragen nach kontinuierlichen Medien (z.B. VoD), indem sie bestehende Resourcen (statische und dynamische CM) so oft wie möglich verwenden. Dabei versucht ein Helper, andere Helper, die ihm die benötigten Informationen liefern können, zu identifizieren. Wie diese Mechanismen im einzelnen funktionieren, wird in Abschnitt 3.2.1 erklärt.

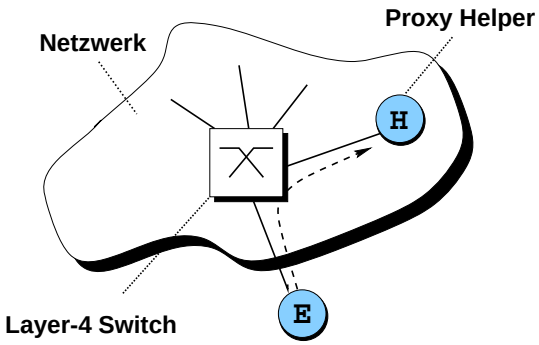


Abbildung 6: Funktionsweise der Helper

Wenn nun ein Benutzer eine Objektanfrage an einen Server im Netz stellt, wird diese automatisch zu seinem *Proxy Helper* umgeleitet (Abbildung 6), der nun versucht, die Daten, mit Hilfe seiner Informationen von seiner Umgebung[3], zu holen. Falls dies nicht möglich ist, wird die Anfrage an den Ursprungsserver (entspricht $S$ in Abbildung 5) weitergeleitet. Folgender Abschnitt stellt mögliche Fälle vor:

In der Ausgangssituation gibt es zwei Empfänger ($E_1$, $E_2$), zwei Helper ($H_1$, $H_2$) – deren CMs leer sind – und einen Sender ($S$). Empfänger $E_1$ fragt nun zum Zeitpunkt $t_1$ nach einem Objekt. Diese Anfrage wird zu seinem Proxy Helper $H_1$ umgeleitet, der die Anfrage zum Zeitpunkt ($t_1'$) erhält (1 in Abbildung 7). Da sich die angefragten Informationen nicht in dem CM des Helpers befinden (vgl. Ausgangssituation) und auch nicht von anderen CMs geholt werden können, wird die Anfrage zu $S$ weitergeleitet ($t_1''$) (2 in Abbildung 7). Dieser legt für $E_1$ einen neuen Datenstrom[4] an. Zum Zeitpunkt ($t_1'''$) erhält $H_1$ die ersten Daten von $S$ und beginnt, diese in seinem *Ring-Puffer*, der $\Theta$ Sekunden aufnehmen kann, abzulegen (3 in Abbildung 7). Diese Daten werden unverzüglich an $E_1$ weitergeleitet (4 in Abbildung 7). In dem Moment, in dem $H_1$ Daten von $S$ erhält, gibt er dies auch anderen Helpern bekannt (5 in Abbildung 7). $E_2$ stellt darauf, zum Zeitpunkt $t_2$, eine Anfrage an den Server $S$ für das gleiche Objekt, welche seinen Helper $H_2$ zum Zeitpunkt $t_2'$ erreicht (6 in Abbildung 7). Für $H_2$, der über den Status von $H_1$ informiert ist, gibt es verschiedene Möglichkeiten:

- Er kann jedesmal zuerst den Ursprungsserver($S$) kontaktieren, um die Daten zu bekommen. Vorteil ist, daß $H_2$ für diese Variante keinen zusätzlichen freien *Puffer* benötigt. Ein offensichtlicher Nachteil liegt jedoch in der hohen Netzwerkbelastung (für jede Benutzeranfrage würde bei $S$ ein eigener Stream geöffnet), die es gerade zu reduzieren gilt.

[3] vgl. Aufbau von *Meshes*

[4] Dieser kann die Form einer Unicast- oder Multicast-Kommunikation haben, wobei wir hier vom letzteren ausgehen.

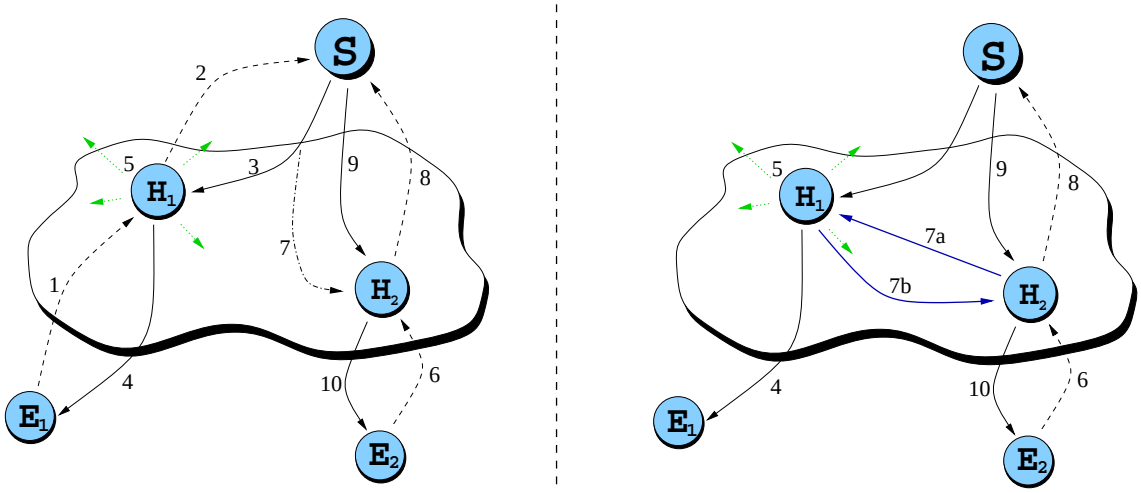


Abbildung 7: Beispiel für eine Vereinte Cache Architektur

- $H_2$ kann sich genausogut der Multicast-Gruppe ($H_1$, $E_1$) anschließen, da die Daten bereits von $E_1$ über $H_1$ abgefragt werden (7 in Abbildung 7). Hier muß der *Puffer* von $H_2$ groß genug sein, um $\Delta = t'_2 - t'_1 + \epsilon$ Sekunden[5] des Objekts (O) aufzunehmen. Gleichzeitig holt sich $H_2$ die ersten Sekunden von O (*Patching Data*) entweder von $H_1$ oder direkt von $S$ (8,9 in Abbildung 7). Sobald nun der *Patch* bei $E_2$ komplett vorhanden ist, werden ihm aus dem *Ring-Puffer* heraus die Daten gesendet (10 in Abbildung 7).

- Die letzte Möglichkeit macht den Vorteil der *Helper* direkt deutlich. Da $H_2$ über den Inhhalt von $H_1$ informiert ist, kann er sich, anstatt sich der Multicast-Gruppe anzuschließen, auch an $H_1$ „anhängen“. Um die Daten zu bekommen, sendet $H_2$ eine Anfrage an $H_1$, der diese zum Zeitpunkt $t''_2$ erhält (7a in Abbildung 7). Daraufhin vergrößert $H_1$ die Größe seines dynamischen *Ring-Puffers* um $\Delta = t''_2 - t'''_1$ Sekunden und beginnt unverzüglich mit der Auslieferung der angeforderten Daten an $H_2$ (7b in Abbildung 7). Der Rest ist mit der vorangehenden Möglichkeit identisch, wobei zusätzlich die ersten Sekunden des Objekt an $E_2$ gesendet werden müssen.

Natürlich ist es für $H_2$ möglich, lediglich statischen Cache zu benutzen. Dies kann situationsbedingt besser sein, da es selbst beim Zugriff auf dynamischen CM Vor- und Nachteile gibt.

- *Vorteile des dynamischen CM:* Auf ihn kann, da er sich im Speicher und nicht auf der Festplatte befindet, sehr schnell zugegriffen werden. Darüberhinaus ändert er sich fortlaufend mit dem Medienstrom und kann deshalb letztendlich für einen Empfänger das gesamte Objekt bereitstellen.

- *Nachteile des dynamischen CM:* Falls ein Anwender den Medienstrom abbricht (z.B. möchte er einen Film nicht weiter anschauen) würde sich trotzdem eine gewisse, nicht abgerufene Länge des Medienobjekts in dem dynamischen CM befinden, da dieser vorausschauend gefüllt werden muß. Dadurch würde notgedrungen für Daten Bandbreite verschwendet, die nicht mehr benötigt werden. Ein weiterer Nachteil des dynamischen CM wird offensichtlich, falls dynamisches Caching ohne statischem Caching verwendet

[5] $\epsilon$ beschreibt die Verzögerung, die aufgrund von verschiedenen Faktoren, wie Netzwerkbelastung, auftreten kann.

| Intervall Nr. | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TTL | 15 | 31 | 15 | 63 | 15 | 31 | 15 | 127 | 15 | 31 | 15 | 63 | 15 | 31 | 15 | 254 | 15 | 31 |

Tabelle 1: TTL-Sequenz

wird. In diesem Fall würde jedes *Data-Patching* zu einer Anfrage an den Ursprungs-Server führen und somit das Netzwerk belasten.

Wenn der dynamische Ring-Puffer voll ist, werden die Daten in den langsameren statischen Cache, der sich in der Regel auf der Festplatte (HD) befindet, geschrieben. Es zeigt sich, daß eine Kombination von statischen und dynamischen CMs selbst bei kontinuierlichen Medien eine Einsparung an Bandbreite bewirken kann.

### 3.2.1 Mechanismen zur Kooperation und Auswahl von Helpern

Wie bereits angesprochen, liegt der zentrale Vorteil der *Helper* in der dynamischen selbstständigen Konfiguration der *Meshes*. Dabei senden diese in bestimmten Abständen Meldungen über ihren eigenen Status an andere Helper, um diese zu informieren. Diese Statusinformationen werden genutzt, um sich die besten „Partner“ zur Kooperation herauszusuchen, wobei die Selektion auf Basis von Leistungs-Indizes[6] geschieht. Je nach Auswahl wird entweder ein neuer Datenstrom erzeugt oder ein bereits aktiver Multicast-Strom mitbenutzt. Im folgenden werden Techniken zur Auswahl und Kooperation vorgestellt.

Dem grundsätzlichen Bestreben nach minimaler Netzwerkbelastung steht eine möglichst umfassende und aktuelle Sicht eines jeden Helpers gegenüber. Da der Austausch von Statusinformationen zwischen Helpern selbst für Netzwerkverkehr sorgt, aber man auf der anderen Seite jedem Helper eine möglichst globale Sicht ermöglichen will, muß man ein Verfahren finden, das einerseits den Netzverkehr gering hält, andererseits aber die Auswahl der optimalen Partner nicht zu sehr durch ein zu geringes ‘Sichtfeld‘, einschränkt.

Da man sich vorstellen kann, daß Helper, die näher beieinander liegen, öfter miteinander kooperieren, wurde darauf aufbauend der Algorithmus des *Expanded Ring Advertisement (ERA)* entwickelt. Dieser basiert auf Abständen, die in *Hops* gemessen werden. Pakete mit verschiedenen TTL-Zeiten[7] werden in bestimmten Zeitabständen gesendet, wobei mit steigender TTL-Zeit die Häufigkeit, in der diese verschickt werden, abnimmt.

Die Tabelle 1 ist so zu verstehen, daß in einem TTL-Radius von 15 (incl.) jedesmal eine Statusmeldung von einem Helper zu anderen Helpern gesendet wird. Jedes zweite mal werden Meldungen bis zu einem Radius von 31 verschickt. Je nach Intervall werden nun die Statusmeldungen bis zu einer TTL 31, 63, 127 oder 254 gesendet.

Man erkennt deutlich, daß Meldungen bis TTL 15 am häufigsten gesendet werden. Weltweit wird hingegen nur jedes 16. Mal der Status verbreitet.

Dieses Verfahren bildet somit einen guten Kompromiß zwischen Aktualität und Belastung des Netzes.

Wie bereits aus dem vorangehenden Beispiel aus Abschnitt 3.2 ersichtlich, muß sich ein Helper entscheiden, woher er sich die benötigten Daten besorgt. Er kann dabei Teile oder gesamte Objekt von seinem eigenen lokalen Cache, direkt vom Server oder von dem dynamischen bzw. statischem Cache eines anderen Helpers beziehen. In der Praxis wird er sich in der Regel

[6] z.B. zeitlicher Abstand, Netzwerk (räumlicher) Abstand, Auslastung des *Helpers*

[7] Time To Live, enspricht der Reichweite.

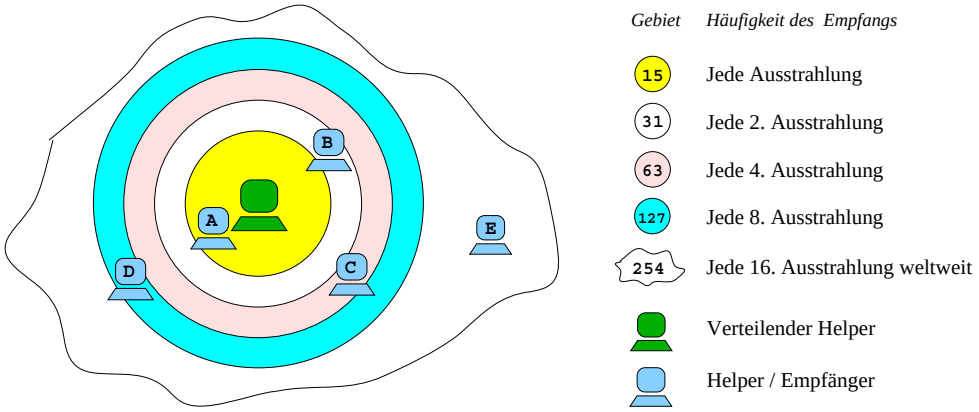


Abbildung 8: TTL Ausbreitungsgebiete

mehrerer verschiedener Quellen bedienen, wobei das Ziel in der optimalen Kombination bei der Datenanfrage liegt. Es gibt zwei Möglichkeiten, um dies zu erreichen:

- Der Algorithmus vergleicht alle in Betracht kommenden Sequenzen und wählt schließlich die beste aus. Jedoch setzt dieses Verfahren voraus, daß der Helper vorausschauend handelt. Dies verlangt eine aufwendige Implementierung, da sich dynamische und statische Caches mit der Zeit ändern und somit eine zur Kalkulationszeit optimale Lösung beim späteren Abruf eine schlechte Wahl darstellen kann. Benutzergewohnheiten, die aber nicht vorhersehbar sind, müßten bei der Wegewahl mit berücksichtigt werden. Da aber für dieses Problem noch keine Lösung existiert, muß der optimale Weg immer neu berechnet werden. Daten, die in einem dynamischen Cache eines anderen Helpers in Betracht kommen, müssen bei diesem Verfahren unverzüglich angefordert werden, da sie zu einem späteren Zeitpunkt eventuell nicht mehr vorhanden sind.
- Eine zweite Möglichkeit ist das schrittweise Vorgehen bei der Suche nach dem optimalen Cache. Wenn eine optimale Lösung gefunden wird, fährt der Helper sofort mit der Objektanfrage fort und sucht die nächste optimale Lösung erst nach Erhalt des zuvor angefragten Objekts. Der ausführliche Algorithmus wird in [HPGN+99, Seite 15ff.] erklärt. Bei der Auswahl des optimalen Servers fließen auch verschiedene Kostenfaktoren mit ein, die es zu minimieren gilt. Diese entstehen bei der Benutzung von statischem und dynamischen CMs.

## 3.3 Kostenfaktoren

Bei der Kostenkalkulation muß man die Bedürfnisse des Dienstanbieters und des Benutzers miteinander verbinden. Während der Anbieter die optimale Benutzung seiner Ressourcen wünscht, steht bei dem Benutzer der schnellstmögliche Zugriff auf die angeforderten Daten im Vordergrund. Da es keine Funktion gibt, die beide Interessen berücksichtigt, muß man versuchen, einen Kompromiß zu finden, der beiden gleichmäßig gerecht wird. Deshalb wird eine Funktion aufgestellt, in der neben der System- (*System Load (L)*) auch die Netzwerkbelastung (*Network Load (N)*) berücksichtigt wird. Die Netzwerkbelastung wird über die Entfernung im Netz approximiert. Hierbei kommt der ERA-Algorithmus zur Anwendung. Die Systembelastung wird über die Anzahl eingehender und ausgehender Ströme bestimmt. Zu beachten ist auch, daß bei hoher Belastung des Servers eine neue Anfrage zu signifikanten Leistungseinbrüchen führen kann, hingegen bei einem niedrigen Auslastungsgrad eine neue Anfrage keinerlei relevanten Auswirkungen hat. Daraus folgt, daß die Kostenfunktion bei niedriger Belastung langsamer als bei hoher Belastung steigt. Eine normalisierte Kostenfunktion (C)

entsteht nun über die Verknüpfung von N mit L. Weitere Angaben zu dieser Formel kann man in [HPGN+99] finden:

$$C = \begin{cases} 2 \cdot N \cdot L & , für \quad \frac{s}{2} \leq \Delta \\ ((2 \cdot \frac{\Delta}{s}) + 1) \cdot N \cdot L & , für \quad \Delta < \frac{s}{2} \end{cases}$$

mit

- $\Delta$: Beschreibt die zeitliche Distanz zwischen der Anfrage und der folgenden Antwort des Helpers. Dabei ist $\Delta \geq 0$.
- $s$: Die Anzahl der Segmente von Beginn bis Ende des Objekts. Dabei ist $s \geq 0$.
- $x$: Die Anzahl der Objekte, die tatsächlich bis zu diesem Zeitpunkt benutzt wurden, wobei $0 < x \leq s$.

### 3.4 Simulation

Um die Ideen und Algorithmen einer Prüfung zu unterziehen, wurde eine Testumgebung aufgebaut, in der verschiedene Effekte bei der Variation der Parameter zu beobachten waren:

- *Effekt der Chunk-Größe*: Sie bestimmt die maximale Größe für einen Chunk (vergleiche Abschnitt 3.1.3) im statischen Cache, bevor ein neues Chunk angelegt wird. Solange ein Chunk in Benutzung ist, kann es nicht aus dem Speicher gelöscht werden. Bei der Variation dieser Größe zwischen 50 MB, 500 MB, 750 MB und 1 GB wurde erkannt, daß je größer diese bemessen wird, eine höhere Leistung erreicht werden kann. Dies ist darauf zurückzuführen, daß beliebte, häufig angefragte Objekte nun mehr Platz auf dem Speichermedium belegen, da diese aufgrund der Zugriffsbeschränkung nicht gelöscht werden können. Daraus resultiert, daß das System, ähnlich der frequenzabhängigen Caching-Politik (im folgenden Abschnitt), auf Benutzergewohnheiten eingeht und nicht häufig benutzte Objekte löscht, wie es bei einer kleineren Chunk-Größe der Fall wäre. Der Nachteil einer zu kleinen Chunk-Größe ist somit, daß man auf mehrere statische Caches zugreifen müßte, um Segmente zu erhalten, die bereits aus dem lokalen CM, da sie wieder freigegeben waren, gelöscht bzw. überschrieben wurden.
- *Effekt der dynamischen Cache-Konfiguration*: Wie zu erwarten war, trägt die dynamische Cache-Konfiguration entscheidend zur Reduzierung des Netzverkehrs bei. Durch die Implementierung des Helper-Auswahl-Algorithmus konnte eine sehr kostensparende Auswahl bei der Datenbeschaffung beobachtet werden.
- *Effekt des dynamischen Caching*: Zu zeigen ist, ob die dynamischen Caches tatsächlich für eine Überbrückung der zeitlich versetzten Anfragen von Vorteil sind. Da dynamischer und statischer Speicher, bei dieser Messung, aus den gleichen Ressourcen gespeist werden und dynamischer Cache solange zugewiesen bleibt, bis explizit die Zuweisung gelöscht wird, kam es zu Leistungseinbrüchen. Zu erklären ist dies dadurch, daß ab einer bestimmten Objektgröße der dynamische dem statischen Puffer benötigten Speicher wegnahm, wodurch dieser nicht mehr effektiv arbeiten konnte. Bestimmte Segmente mußten zwangsläufig vom statischen Cache gelöscht werden um dem dynamischen Cache, dieser hat eine höhere Priorität, Speicher zu übergeben. Dieses Problem zu beheben bedarf weiterer Forschung und Analyse!

- *Effekt des Multicasting*: Der Vorteil von Multicasting konnte nicht herausgefunden werden. Auch wenn nur Unicast-Verbindungen explizit zur Verfügung standen, waren die Leistungswerte nahezu identisch mit denen des Multicastings. Ein Grund hierfür ist, daß die Helper-CMs selbst ein „Multicast"-Netz aufbauten, wodurch Anfragen immer aus den CMs heraus bedient werden konnten. Dieser Seiteneffekt, der eigentlich für die Leistungsfähigkeit der Helper spricht, konnte nicht unterbunden werden, so daß eine Messung auf Basis von explizitem Multicasting nicht möglich war.

- *Effekt anderer Parameter*: Das Reduzieren des Zeitintervalls, in dem die einzelnen Helper Statusmeldungen aussenden, führte zwar zu einer aktuelleren Sichtweise jedes Helpers, jedoch nahm die Belastung des Netzes zu. Es wird angenommen, daß ein Zeitintervall, welches sich an dem Eintreffen der Anfragen orientiert, die besten Ergebnisse liefert. Auch hier besteht noch Forschungsbedarf.

## 3.5 Algorithmen zur Verbesserung der Cache-Auslastung

Speziell bei VoD kommt die Frage auf, wie man die Benutzergewohnheiten am besten in die Implementierung der Cache-Systeme einbindet. Je genauer man weiß, welche Filme am häufigsten abgefragt werden, um so effektiver könnte man die CMs belegen. Dieses Problem wurde basierend auf der Modellarchitektur, die in Abbildung 9 dargestellt ist, in dem Paper [HPGN+99], behandelt. Im Folgenden wird das Fazit dieser Ausarbeitung in Auszügen vorgestellt.

Wo man die Video-Server (VS) und Video-Datenbanken (VD) am besten aufstellen sollte, geht auf die Frage der Netzwerkarchitektur zurück (siehe Abschnitt 3.1.1). Welche Filme (Daten) aber in dem jeweiligen Video-Puffer (VB) gespeichert werden sollten, kann man langfristig nur durch genaue Marktbeobachtungen herausfinden. Da solche Daten sehr schwer zu bekommen sind, wurden Ideen und Lösungsansätze basierend auf den Top 50 Album Charts Daten der Australian Record Industry Association (ARIA) entwickelt. Diese Daten wurden zwei Jahre lang gesammelt und ausgewertet. Zwar ist bekannt, daß die Gewohnheiten, Trends und Bewegungen der Musik-Charts andere sind als die der Video-Charts, jedoch spielt dies in dieser Anlayse eine nebensächliche Rolle. Ziel war es, einen Algorithmus zu entwickeln, der Ergebnisse liefert, die man mit den tatsächlichen Gewohnheiten vergleichen kann.

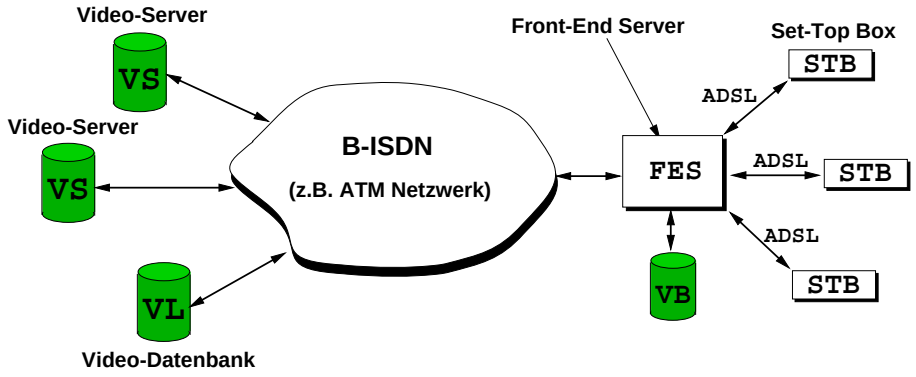


Abbildung 9: Modellarchitektur für VoD

Dieses VoD-Netzwerk (Abbildung 9) basiert auf der Benutzung verschiedener Schichten zur Speicherung der Daten. Die VS, die verschiedenen Filmgesellschaften gehören, speichern alle Filme, die dem Endanwender zur Verfügung stehen. Diese müssen für den *Front-End Server (FES)*, der sich geographisch in der Nähe der Benutzer befindet, zugänglich sein, um bei Bedarf den angeforderten Film abzurufen. Da bestimmte Filme nun häufiger abgefragt werden

als andere, wird dies durch das Duplizieren dieser Filme in einen *Video-Puffer (VB)* berücksichtigt. Dadurch kann die Belastung des Netzwerks reduziert werden, da nicht mehr alle Anfragen an die zentralen *Video-Server (VS)* bzw. *Video-Datenbanken (VD)* weitergeleitet werden müssen. Für das Füllen des VB stehen mehrere Algorithmen zur Verfügung:

- *Ideal:* Die beliebtesten $N$ Filme, basierend auf den Charts, werden immer im Cache gehalten.
- *Statisch:* Der Cache wird einmalig mit den beliebtesten $N$ Filmen gefüllt.
- *Häufigkeitsabhängig:* Abhängig von der Häufigkeit der Anfragen nach einem Film wird dieser mit in den Cache aufgenommen. Da die absolute Anzahl der gespeicherten Filme konstant bleibt, fällt daraufhin der am seltensten gesehene Film aus dem Cache heraus.
- *Frequenzabhängig:* Der VB wird abhängig von der Zeit zwischen zwei nacheinander folgenden Anfragen an einen Film gefüllt.
- *Durchschnittsfrequenz:* Identisch mit dem frequenzabhängigen Füllen des VBs, jedoch werden nicht nur die letzten zwei Benutzeranfragen nach einem Film berücksichtigt, sondern mehrere zurückliegende Anfragen, deren Zeitintervalle zwischen den einzelnen Benutzeranfragen gemittelt werden.

Jeder dieser Algorithmen wurde in einer Testumgebung mit 30.000 Benutzern und 1000 Filmtiteln implementiert, um die Leistungsfähigkeit zu messen. Dabei varierte man neben der Cache-Größe auch die Geschwindigkeit der Positionswechsel der Filme in den Charts. Es stellte sich heraus, daß der frequenzabhängige Algorithmus die besten Ergebnisse bezüglich Leistung und Minimierung der Netzwerkbelastung liefert. Bei dieser Caching-Politik spielt es auch keine Rolle, mit welcher Geschwindigkeit sich die Charts bewegen. Die Ergebnisse (gemessen in Cache Hits) sind nahezu identisch, so daß dieses Verfahren relativ unabhängig von Benutzergewohnheiten eingesetzt werden kann [BaAB96].

## 4 Zusammenfassung und Ausblick

Die wissenschaftlichen Veröffentlichungen, die dieser Ausarbeitung zugrunde liegen, bilden erste Ansätze zur Lösung des Problems der Verwendung von kontinuierlichen Datenströmen über das Internet. Es wurde gezeigt, daß das gezielte Einsetzen von Helpern oder auch das Verwenden der DCCA-Architektur zu meßbaren Leistungssteigerungen führen kann. Jedoch mußte in den eingesetzten Simulationsumgebungen festgestellt werden, daß u.a. die Verwendung von Multicast oder Unicast keine Leistungsänderungen hervorriefen. Zukünftige Messungen sollten durchgeführt werden, um das Zusammenspiel der einzelnen CMs noch besser verstehen zu können. Aber auch die Algorithmen, die zum Füllen der CMs mit entsprechenden Filmen verwendet wurden, basieren bis jetzt noch auf Daten der Musik- und nicht auf denen der Videoindustrie. Zusammenfassend kann man sagen, daß der Grundstein für eine Implementierung von kontinuierlichen Medien gelegt ist, es aber noch weiterer Forschung bedarf, um die Technik für die breite Masse der Benutzer verfügbar zu machen.

## Literatur


[BaAB96] Scott A. Barnett, Gary J. Anido und H.W. Peter Beadle. Caching Policies in a Distributed Video-on-Demand System. Technischer Bericht, The Institute for Telecommunications Research, University of Wollongong - Australia, 1996.

[DaMo98] Giacinto Dammicco und Ugo Mocci. Program Caching and Multicasting Techniques in VoD Networks. Technischer Bericht, Fondazione Ugo Bordoni, 1998.

[HPGN+99] Markus Hofmann, Sanjoy Paul, Katherine Guo, T.S. Eugene Ng und Hui Zhang. Caching Techniques for Streaming Multimedia over the Internet. *Technical Memorandum*, 1999.

# Multicast Routing – Wegewahlverfahren für die Gruppenkommunikation

Matthias Grimm

**Kurzfassung**

Im Rahmen der stetigen Vergrößerung der Bandbreite moderner Netze sowie der Integration neuer Dienste entsteht eine Vielzahl neuer Anwendungen, die nicht mehr auf der Kommunikation zwischen zwei Rechnern, sondern zwischen einer ganzen Gruppe von Rechnern aufbauen. Im Gegensatz zum Unicast, wo nur ein anderer Rechner adressiert wird, spricht man bei der gleichzeitigen Adressierung mehrerer Rechner von Multicast. In dieser Arbeit wird die grundsätzliche Funktionsweise der an der Gruppenkommunikation beteiligten Aspekte, nämlich sowohl die Bestimmung von Subnetzen, die Multicast-Daten erhalten sollen, als auch die Wegewahl für die Paketvermittlung zwischen diesen Subnetzen, beschrieben. Im Rahmen der Gruppenverwaltung wird auf das *Internet Group Management Protocol IGMP* und die Gruppenadressierung in der aktuellen IP Version 4 sowie auf die Änderungen in IPv6 eingegangen. Nach einer Klassifizierung der Routing-Protokolle nach ihrem Einsatzgebiet, für die sie am besten geeignet sind, werden die wichtigsten Protokolle im Detail vorgestellt, darunter das weit verbeitete MOSPF und das noch wenig eingesetzte PIM-SM. Abschließend wird mit dem MBone eine stark im Wachstum befindliche Testumgebung für Multicast im Internet vorgestellt.

## 1 Einleitung

In einer ständig steigenden Zahl von Anwendungsszenarien werden Daten von einzelnen Rechnern an Gruppen von anderen Rechnern gesendet, z.B. Internet-Radio, Whiteboards oder Telekonferenzen. Würden solche Rechner jedes einzelne Paket an jeden Empfänger über eine separate Punkt-zu-Punkt Verbindung versenden, man bezeichhnet dies als *Unicast*, so wäre dadurch bald unnötigerweise sehr viel Bandbreite belegt (Abb. 1). Bandbreite kann in erheblichem Maße eingespart werden, wenn ein Paket an eine Gruppe von Systemen nur einmal verschickt und vom Netz an Zweigstellen repliziert wird. Dieses Vorgehen bezeichnet man als *Multicast* (siehe Abb. 2). Um diesen Weg zu gehen, müssen zwei Fragen geklärt werden: Wie und wo können solche Gruppen von Empfängern verwaltet werden und wie können sie adressiert werden? Diese Fragen werden in den folgenden Abschnitten behandelt.

### 1.1 Gruppenverwaltung

Um Daten an mehrere Rechner zu übermitteln, ohne jeweils ein Paket an jeden Rechner zu senden, ist es notwendig, die Empfänger in eine Gruppe mit einer eindeutigen Gruppenadresse zusammenzufassen. Dafür wird eine Gruppenverwaltung benötigt, die es Rechnern ermöglicht, Gruppen beizutreten, Gruppen zu verlassen oder eventuelle Attribute der Mitgliedschaft zu ändern.

Damit die Zwischensysteme Pakete korrekt an Gruppen weiterleiten können, müssen sie die Gruppenzugehörigkeiten kennen. Dazu können sie entweder zyklisch die Hosts fragen, zu welchen Gruppen sie gehören, oder die Hosts berichten von sich aus jede Änderung. Wie dies im

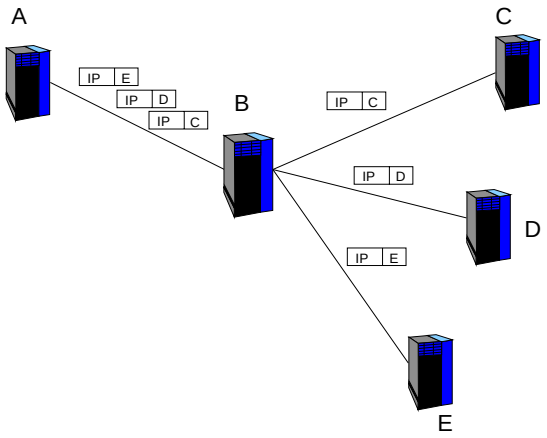


Abbildung 1: Unicast an mehrere Systeme

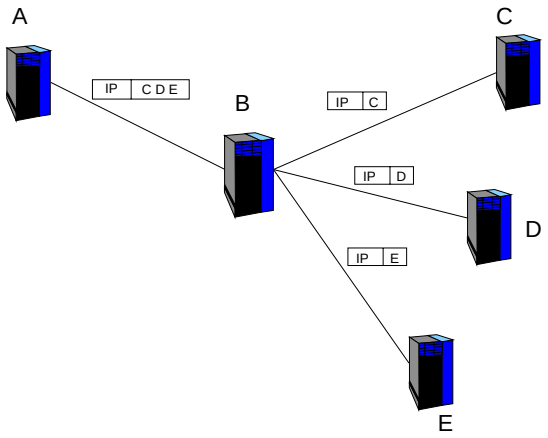


Abbildung 2: Multicast an mehrere Systeme

Internet realisiert ist, wird in Abschnitt 2.2 genauer beschrieben. Wurden diese Informationen von den Routern gesammelt, können diese untereinander die Gruppendaten austauschen und somit einen Baum zur effizienten Datenverteilung zusammenstellen. Die Knoten in diesem Baum sind die Vermittlungssysteme, und die Endsysteme werden durch die Blätter dargestellt. Die Wurzel entspricht dem Sender, so können die Pakete von der Wurzel durch die Äste in die Blätter zu den Endsystemen vermittelt werden.

Ein Beispiel für ein Gruppen-Mitgliedschafts-Protokoll sind die *Domain-Wide Multicast Group Membership Reports, DWR's*. Sie spezifizieren eine protokollunabhängige Möglichkeit, Gruppenzugehörigkeitsinformationen innerhalb einer Domäne zu sammeln. Es stehen den Routern Möglichkeiten zur Nachfrage (*Query*) von Zugehörigkeiten zur Verfügung (*Domain-Wide Query, Domain-Wide Report, Domain-Wide Leave* und *Non-Authoritive Domain-Wide Leave*). Anfragen beziehen sich ohne weitere Angaben auf alle Gruppen, spezifische Gruppen können aber im Datenteil angegeben werden.

## 1.2 Weiterleitung von Multicast-Paketen

Die Wegewahl (Routing) für Gruppenkommunikation unterscheidet sich vom herkömmlichen (Unicast-)Routing hauptsächlich darin, daß die Verbindungsstruktur zwischen den Kommunikationspartnern durch die hohe Dynamik innerhalb der Gruppen durch hinzukommende und verlassende Empfänger starken zeitlichen Wandlungen unterliegt. Außerdem kann ein Router an der Multicast-Adresse a priori nicht erkennen, an welche Domäne ein Gruppenpaket gesendet werden soll und an welche nicht (siehe auch 2.1), wodurch der Router die Pakete anfangs nicht auf eine bestimmte Ausgangsleitung legen kann, sondern das Paket flutenmuß, also an jede Leitung senden, mit Ausnahme derjenigen, auf der er das Paket empfangen hat. Diese Vorgehensweise ist nun aber kein echter Multicast sondern Broadcast, und das Netz wird sehr stark belastet. Eine Verbesserung namens *Pruning* erlaubt Routern, an denen kein Empfänger einer Multicast-Gruppe angeschlossen ist, mitzuteilen, daß sie keine Pakete an die zuvor adressierte Gruppe benötigen. So kann für jede Multicast-Gruppe durch Abschneiden von Ästen ein optimierter Routing-Baum erstellt werden.

Da dieses Thema Schwerpunkt der Arbeit ist, werden die Verfahren für die Wegewahl im nächsten Abschnitt ausführlicher behandelt.

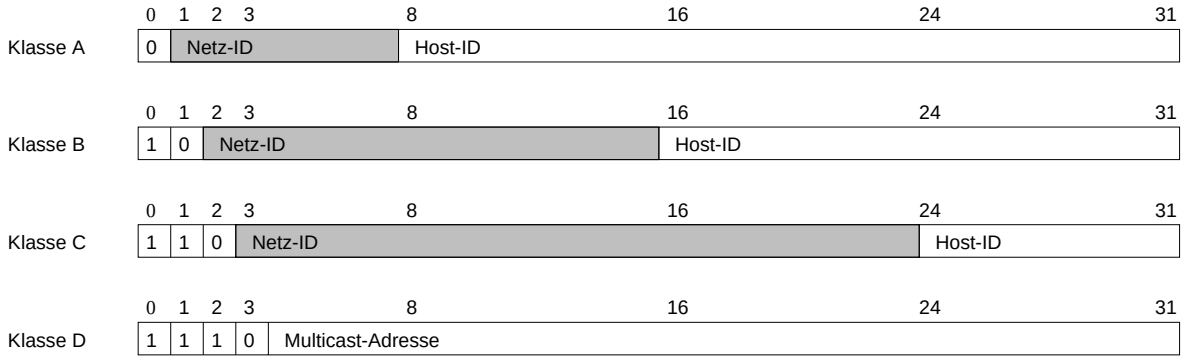


Abbildung 3: Die unterschiedlichen Adreßklassen in IPv4

# 2 Gruppenkommunikation und IP

## 2.1 IP-Adressierung

Die Adressierung von IP Paketen erfolgt zur Zeit nach dem noch vorwiegend eingesetzten IP-Protokoll Version 4. In einer stark wachsenden Anzahl von Versuchsnetzen in Überlagerungstechnick ähnlich dem MBone (6Bone) wird die neue Version 6 von IP mit erweiterter Funktionalität und einem größeren Adreßraum erprobt. Die Eigenschaften der beiden IP Versionen sowie deren Unterschiede werden in den beiden folgenden Abschnitten erläutert.

### 2.1.1 IP Version 4

In der Version 4 von IP sind Adressen 32 Bit lang und in zwei Teile gegliedert, die Netz-ID und die Host-ID. Die Länge dieser beiden Teiladressen wird durch die Klassen A, B, C, und D festgelegt, wie es in Abb. 3 dargestellt ist.

In den Klassen A bis C sinkt die mögliche Hostanzahl pro Netz, die Anzahl möglicher Netzadressen nimmt allerdings stark zu. Es ist offensichtlich, daß Klasse A Netze mit ihren unterschiedlichen $2^{24} \cong 16,7 Mio$ Host-Adressen für sehr große Firmennetze eingesetzt werden, während Klasse-C-Adressen mit ihren 255 unterscheidbaren Adressen in kleinen privaten Netzen Anwendung finden.

Klasse-D-Adressen werden verwendet, um Multicast-Gruppen zu adressieren. Ihr Unterschied zu Adressen der anderen Klassen ist zum einen, daß sie nicht unterteilt in Netz- und Host-Adresse, sondern flach sind. Außerdem werden sie nicht fest vergeben wie andere Adressen, sondern immer nur für eine Sitzung, und das meist manuell. Ihre automatische Vergabe auf Anforderung ist nämlich zur Zeit noch nicht geklärt. Ihr Wertebereich reicht von 224.0.0.0 bis 239.255.255.255, wobei der Bereich von 224.0.0.0 bis 224.0.0.255 für bestimmte Anwendungen reserviert ist, wie z.B. „alle Router im Subnetz“, „alle OSPF Router“ etc.

Die Verwaltung der Gruppen innerhalb eines Subnetzes geschieht in IPv4 über das *IGMP (Internet Group Management Protocol)*. Dieses Protokoll existiert in den Versionen IGMPv1 und IGMPv2, an einer dritten Version IGMPv3 wird zur Zeit noch gearbeitet. Näheres dazu, insbesondere über den Protokollablauf, folgt in Abschnitt 2.2.

### 2.1.2 IP Version 6

In der aktuellen Version IPv6 des IP-Protokolls gibt es einige Neuerungen gegenüber dem Multicast in der 4 Version 4 von IP, die durch folgende Probleme motiviert wurden:

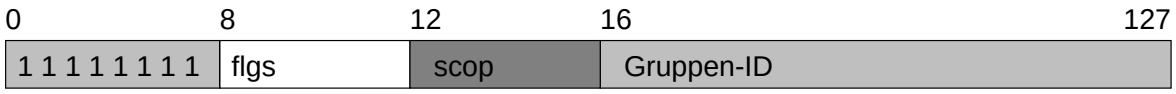


Abbildung 4: Multicast-Adressen in IPv6

- Unzureichendes Adreßformat: aufgrund des starken Wachstums des Internet wird der bisher großzügig verteilte Adreßraum des Internets aufgrund des hohen Verschnittes knapp. Ein Netz mit beispielsweise 300 Rechnern benötigt eine Klasse-B-Adresse, die aber den Betrieb von 65500 Rechnern ermöglicht, d.h. ca. 65000 Adressen bleiben ungenutzt.
- Unnötig komplexes Paketformat: Vereinfachung des IP-Paketkopfes und Verschieben einiger Optionen in flexible Paketkopferweiterungen.
- Ungenügende Sicherheit: Sicherheit (Authentifikation und Datensicherheit) werden in IPv4 kaum unterstützt.

IPv6 löst einige dieser Probleme. Eine der auffallendsten Änderungen ist die Erweiterung der Adressen von 32 auf 128 Bits. Neben unterschiedlichen Adreß-Hierarchieebenen wird eine *Anycast-Adresse* eingeführt, die die Kommunikation zu einem beliebigen Mitglied einer Gruppe ermöglicht. Die Adreßklassen wie in IPv4 gibt es in dieser Form nicht mehr. Multicast-Adressen (Abb. 4) werden durch jeweils eine Eins in den ersten acht Bit der Adresse markiert. Zwei weitere Felder, nämlich *flgs* und *scop* geben die Art der Multicast Adresse an. So gibt das niederwertigste Bit in *flgs* beispielsweise an, ob die Adresse vorübergehend (transient) oder permanent ist. Das *scop* Feld definiert den Gültigkeitsbereich der Adresse. Zu den möglichen Werten gehören unter anderem „Link-local Scope“, „Site-local Scope“, „Organization-local Scope“ und „Global Scope“. So kann die gleiche ID unter Umständen gleichzeitig mehrfach an jeweils unterschiedlichen Orten verwendet werden.

## 2.2 IGMP

Die Aufgabe des *Internet Group Management Protocol (IGMP)* besteht hauptsächlich darin, Multicast-Routern zu ermöglichen, Informationen über Gruppenmitgliedschaften von Endsystemen in den an sie angeschlossenen lokalen Netzwerken zu sammeln und somit zu entscheiden, welche Multicast-Pakete für sie von Bedeutung sind. Diese Informationen werden dann an das zugrundeliegende Multicast-Protokoll weitergegeben, um ein echtes Routing von Multicast-Paketen zu ermöglichen. Dabei reicht es dem Router zu wissen, daß sich mindestens ein Endsystem aus einer bestimmten Multicast-Gruppe in dem an ihn angeschlossenen Netz befindet. Die genaue Kenntnis über die Mitgliedschaft und Identität eines jeden einzelnen angeschlossenen Endsystems ist unrelevant.

Um diese Gruppeninformationen zu sammeln, gibt es zwei Arten von Nachrichten: Anfragen *(Queries)* und Berichte *(Reports)*. Mit Hilfe der Anfragen kann ein Router Informationen anfordern, und als Antworten darauf erhält er Berichte von Endsystemen, die den Router über die für ihn relevanten Gruppen informieren.

Der Ablauf zur Feststellung der Gruppenmitgliedschaft vernetzter Rechner ist wie folgt festgelegt: der Router sendet periodisch Gruppenzugehörigkeitsanfragen an alle Rechner des LANs mittels Broadcast. Damit sich diese Anfrage nur auf dieses Netz beschränkt, wird der TTL-Wert (TTL: *Time to Live*, ein Zähler im Kopf der IP Pakete, der von jedem Vermittlungsknoten dekrementiert wird, so daß durch ihn die Anzahl der durchlaufenen Knoten (Hops) beim Senden eines Paketes festgelegt werden kann) des IP-Pakets auf den Wert 1 gesetzt.

Nach Erhalt einer solchen Anfrage startet jeder angeschlossene Host für jede Gruppe, in der er Mitglied ist, einen Zeitgeber mit einem zufälligen Startwert und hört ab, ob entsprechende Report-Nachrichten gesendet werden. Ist dieser Zeitgeber abgelaufen, sendet er eine Antwort bezüglich der Gruppenmitgliedschaft an alle Gruppenmitglieder im LAN (TTL-Feld hat den Wert 1), also adressiert an die Multicast-Adresse seiner Gruppe. Multicast-Router erhalten alle Nachrichten. Alle anderen Mitglieder einer Gruppe empfangen das Paket auch und stoppen ihren Zeitgeber. Es wird somit verhindert, daß redundante Antworten über das Netzwerk transportiert werden. Der Router, der nun alle Informationen über die an ihn angeschlossenen Gruppen erhalten hat, aktualisiert seine Routingtabelle. Hat er von einer Gruppe keine Antwort bekommen, so entfernt er diese aus seiner Tabelle. Tritt ein neuer Rechner in eine Gruppe ein, so sendet er sofort von sich aus eine Meldung an alle Router im LAN.

Multicast-Empfänger, die einer neuen Gruppe beitreten möchten, senden sofort einen Report mit TTL=1. Wenn sich außer ihm kein anderer Empfänger dieser Gruppe im Teilnetz befindet, könnten einige an die Gruppe adressierte Pakete verloren gehen, falls sein Report verloren geht oder beschädigt wird. Um dieses Problem zu vermeiden, sendet er ihn häufiger kurz hintereinander. Wenn ein Empfänger seine Gruppe verlassen möchte, kann er eine LEAVE-Nachricht senden. Dadurch wird der Austritt des letzten Mitgliedes einer Gruppe drastisch beschleunigt, da der Router auch im schlechtesten Fall nicht mehr den zeitlichen Abstand zwischen zwei Queries abwarten muß.

Die neueste Protokollversion IGMPv3 unterstützt zusätzlich *Source Filtering*, d.h. die Möglichkeit, Pakete nur von einer bestimmten Quelle zu beziehen. Dies erfordert zusätzliche Anfrage-Pakete, so daß es in IGMPv3 drei unterschiedliche gibt: allgemeine Anfrage, gruppenspezifische Anfrage und gruppen- und quellenspezifische Anfrage. Die genaue Beschreibung der Protokollversion 3 ist [Cain99], ein Internet-Draft, also ein Arbeitsdokument der IETF. Ein RFC zu dieser Version existiert bisher noch nicht.

# 3 Routing-Protokolle

Multicast-Routing-Protokolle werden von Routern in Netzwerken dazu genutzt, den effizientesten Weg herauszufinden, um Daten von einer Quelle zu mehreren Empfängern zu transportieren. Diese Protokolle sind in drei Hauptgruppen unterteilbar, nämlich solche, die

- auf dem *Distance Vector* Protokoll aufbauen (*Distance Vector Multicast Routing Protocol – DVMRP, Protocol Independent Multicast, Dense Mode – PIM-DM*),
- auf dem *Link State* Protokoll aufbauen (*Multicast Open Shortest Path First – MOSPF*) und solche, die
- verteilte Bäume (*Shared Trees*) verwenden (*Protocol Independent Multicast, Sparse Mode – PIM-SM, Core Based Trees – CBT*).

Eine weitere Unterscheidung der Protokolle ist durch ihr Einsatzgebiet gegeben, und zwar anhand der Verteilungsdichte der Gruppen. Da DVMRP, PIM-DM und MOSPF Pakete fluten, um Multicast-Routen zu finden, scheiden sie für den Einsatz in weit verteilten, dünn besetzten Netzwerken aus. Es empfiehlt sich hier der Einsatz von PIM-SM oder CBT, da diese für Netzwerke konzipiert wurden, die sich durch eine weite und lose Verbreitung der Gruppenmitglieder auszeichnen. In den folgenden Abschnitten werden zunächst die den Multicast-Protokollen zugrundeliegenden Unicast-Protokolle vorgestellt, um anschließend auf deren Grundlage die einzelnen Multicast-Protokolle in ihrer Funktionsweise, sowie deren Vor- bzw Nachteile erläutern zu können.

## 3.1 Unicast-Routing

### 3.1.1 Distance Vector Routing

Beim *Distance Vector Routing* verwaltet jeder Router eine Tabelle, die für jeden im Teilnetz vorhandenen Router einen Eintrag mit zwei Feldern enthält: die bevorzugte Ausgangsleitung zu diesem Ziel sowie die Entfernung zu diesem Ziel. Welche Metrik hier verwendet wird, ist nicht festgelegt, vorstellbar ist die Anzahl der Übertragungsabschnitte (Hops), die Warteschlangenlänge oder die mittlere Verzögerungszeit.

Es wird davon ausgegangen, daß jeder Router die Entfernung zu seinen unmittelbaren Nachbarn kennt. Verwendet man als Metrik die Verzögerungszeit, so kann diese mit Hilfe von HELLO-Paketen gemessen werden. Um die Routing-Tabelle aufzubauen bzw. zu aktualisieren sendet jeder Router an jeden Nachbarn die geschätzten Verzögerungen von sich bis zu jedem Ziel. Jetzt kann jeder Router seine eigene Routing-Tabelle Eintrag für Eintrag mit denen seiner Nachbarn vergleichen. Sei beispielsweise gerade die Tabelle von Router $X$ eingegangen, wobei $X_i$ die von $X$ geschätzte Dauer ist, bis er Router $i$ erreicht. Weiß der Router, daß die Verzögerung zu $X$ $m$ ms beträgt, so weiß er auch, daß er $i$ über $X$ in $X_i + m$ ms erreichen kann. Ist dieser Wert besser als der in seiner aktuellen Tabelle, so wird dieser Eintrag aktualisiert und der bessere Wert übernommen.

Diese Methode führt allerdings zu einem Problem, das als *Count-to-Infinity*-Problem bekannt ist. Wird eine Leitung im Netz unterbrochen, so merkt das zunächst der direkte Nachbar des abgetrennten Routers. Dieser sieht allerdings, daß sein anderer Nachbar den nicht mehr erreichbaren Router in einer akzeptablen Zeit erreichen kann, er weiß jedoch nicht, daß diese Route genau über ihn selbst führt. Er ändert seinen eigenen Tabelleneintrag ab, indem er, wie oben erklärt, auf die Zeit, die sein Nachbar liefert, die Zeit aufaddiert, in der er seinen Nachbar erreichen kann. Daraufhin merkt der Nachbar allerdings, daß er den abgetrennten Router nicht mehr so schnell erreichen kann und ändert seinen Eintrag in der Routing-Tabelle entsprechend ab. Es zählen also alle Router immer weiter, bis sie „unendlich“ (*Infinity*) erreichen. Ein möglicher Ausweg aus dieser Situation wäre die Einschränkung, daß ein Router diejenigen Tabelleneinträge nicht an seinen Nachbar $X$ weiterreicht, die über diesen Nachbar $X$ weitergeleitet werden.

Abb. 5 soll dies verdeutlichen: die Router B, C, D und E haben zu A die Entfernung 1,2,3 bzw. 4. Wird nun die Leitung zwischen A und B unterbrochen, erfährt B nichts mehr von A. C teilt B nun mit, daß es A in 2 Schritten erreichen kann. Also vermutet B, das C in einem Schritt erreicht, es könne A über C in 3 Schritten erreichen, da es nicht weiß, daß es selbst auf dem Weg von C nach A liegt. C merkt nun allerdings, daß B 3 Schritte zu A benötigt und erhöht seine eigene Entfernung zu A auf 4, was B wiederum veranlaßt seinen Eintrag auf 5 zu erhöhen usw. Die weiteren Austauschvorgänge ergeben die in der Tabelle dargestellte Situation. Eine ausführliche Beschreibung des Protokolls und der damit verbundenen Probleme mitsamt den möglichen Lösungen ist in [Tane97] nachzulesen.

### 3.1.2 Link State Routing

Das *Link State Routing* kann im wesentlichen in fünf Teile gegliedert werden: Jeder Router muß

- seine Nachbarn erkennen
- Verzögerung oder Kosten zum Nachbarn messen
- ein Paket zusammenstellen, in dem alles Gelernte steht

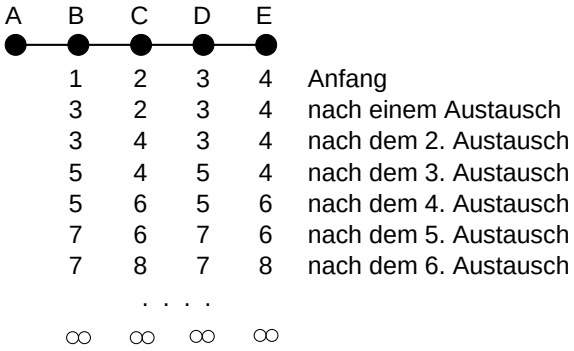


Abbildung 5: Das Count-to-Infinity-Problem

- dieses Paket an alle anderen Router versenden
- den kürzesten Pfad zu allen Routern berechnen.

Router erkennen ihre Nachbarn, indem sie ein spezielles HELLO-Paket mit TTL=1 an jeden Ausgang senden. Dadurch, daß die TTL den Wert eins besitzt, leitet der Empfänger-Router das Paket nicht weiter und kann seinem Nachbarn eine Antwort senden. Es ist also wirklich gewährleistet, daß nur direkte Nachbarn antworten.

Um die Verzögerung zu seinem Nachbarn zu messen, kann jeder Router nun ein spezielles HELLO-Paket senden, dessen Zeitbedarf er für Hin- und Rückweg mißt. Diese Zeit dividiert durch zwei gibt eine gute Abschätzung für die Verzögerung. Um das Ergebnis noch genauer zu bestimmen, kann dieser Test wiederholt und über alle Versuche gemittelt werden.

Sind nun die Informationen über die Nachbarn gesammelt, so kann der Router die Link-State-Pakete erstellen. Sie bestehen aus der Identifikation des Senders, einer Folgenummer, dem Alter, das verwendet werden kann, um alte Informationen zu verwerfen, und der Liste aller Nachbarn mit den zum jeweiligen Nachbarn gehörenden Verzögerungszeiten.

Ein Problem bringt die folgende Situation mit sich: Router, die sich jeweils dicht am Sender-Router befinden, bekommen diese Pakete schneller als weiter entfernt ansässige Router und ändern somit also auch ihre eigenen Routing-Tabellen früher ab, wodurch es zu Inkonsistenzen, Schleifen oder sogar nicht erreichbaren Rechnern im Netz kommen kann. Um diese Probleme handhaben zu können, wird die Folgenummer im Paket benötigt. Trifft ein Link-State-Paket bei einem Router ein, so vergleicht er die Folgenummer mit den entsprechenden Nummern der anderen bisher verarbeiteten Pakete. Ist die neue Nummer größer als die anderen, so wird das Paket an jede Ausgangsleitung weiterverteilt, mit Ausnahme derjenigen, auf der ihn das Paket erreicht hat. Ist die Nummer niedriger als die höchste bereits verarbeitete, so wird das Paket als nicht mehr aktuell verworfen. Handelt es sich um ein Duplikat, so wird es ebenfalls verworfen.

Hat jeder Router nun die vollständige Liste der anderen Router mit den jeweiligen besten Wegen zu deren Nachbarn, kann ein Graph über das gesamte Netzwerk erstellt, und mit dem Algorithmus von Dijkstra der kürzeste Pfad zu allen möglichen Zielen bestimmt werden.

## 3.2 Multicast Routing Protokolle

Nach dieser Vorstellung der Protokolle zur Wegewahl zwischen zwei Kommunikationspartnern können nun die Verfahren für die Gruppenkommunikation vorgestellt werden. Einen Überblick über die Protokolle vermittelt [Mill99], Details sind den entsprechenden RFCs bzw. Internet-Drafts entnommen.

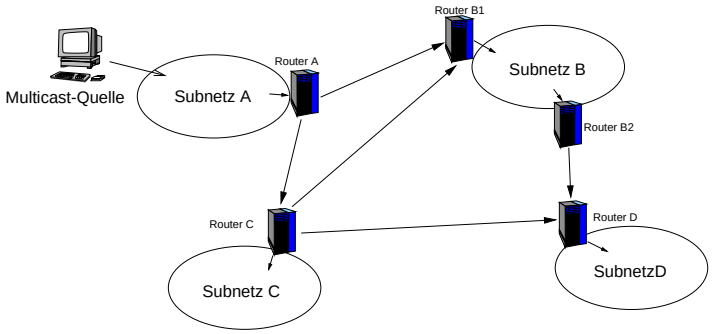


Abbildung 6: Fluten mit DVMRP

### 3.2.1 DVMRP

Das *Distance Vector Multicast Routing Protocol* ist ein auf dem *Distance Vector Routing Protocol* und dem *Reverse Path Forwarding* aufbauendes Routing-Protokoll. Eingesetzt wird es bei großer Dichte der Gruppenteilnehmer. Es ist das älteste Verfahren für Gruppenkommunikation und wurde in RFC 1075 definiert und nach und nach aktualisiert und verbessert. Der aktuelle Stand der Untersuchungen kann [Pusa99] bzw. dessen Aktualisierungen entnommen werden, ein Internet-Draft, der Ergänzungen zu Version 1 des Protokolls beinhaltet.

*Reverse Path Forwarding* ist eine Verfeinerung des allgemeinen Flutungs-Algorithmus, bei dem jeder Router jedes Datagramm, das er empfängt, auf jede Ausgangsleitung legt, mit Ausnahme derjenigen, auf der er es empfangen hat. Beim *Reverse Path Forwarding* überprüft der Router, ob das Datagramm, das er bekommen hat, auf der Eingangsleitung angekommen ist, auf der er selbst Pakete an den Sender dieses Datagramms schicken würde. Ist dies der Fall, flutet er das Paket weiter, anderenfalls verwirft er es, um den Paketverkehr auf optimale Routen zu beschränken. Um zu entscheiden, auf welcher Leitung er dem Sender ein Paket senden würde, baut er mit Hilfe eines eigenen DVRP eine Routing-Tabelle auf. Erhält er ein Paket von einem Nachbarn, an den er kein Paket weiterleiten würde, so verwirft er das Paket, wie bereits erwähnt. Abb. 6 illustriert diese Funktionsweise. Die Datenquelle ist ein am Netz A angeschlossener Host. Router A flutet die Pakete ins Netz, Router C empfängt sie und flutet sie ebenfalls. Router B1 erkennt nun, daß C nicht auf dem kürzesten Pfad zu A liegt und verwirft das Paket.

Nun ist gewährleistet, daß jedes Subnetz die Datenpakete erhält. Daß sie auf dem jeweils kürzesten Weg zugestellt werden, ist dadurch gesichert, daß die Router ihre Routing-Tabellen wie in 3.1.1 beschrieben untereinander austauschen. Ein Problem gibt es aber dennoch: wahrscheinlich wird das Netz immer noch durch eine Menge nicht falsch, aber doch unnötig zugestellter Pakete belastet: wie bereits erwähnt, wissen die Router anfangs nicht, wo sich die Mitglieder einer Gruppenkommunikation befinden. Ist ein Router an ein Subnetz angeschlossen, das kein Mitglied aus der angesprochenen Gruppe enthält, so sendet er seinem angeschlossenen Router, der ihm das Paket zugestellt hat, eine *Prune*-Meldung, die den Weg zu sich abschneidet. Dadurch wird der Routing-Baum auf diejenigen Subnetze beschränkt, welche die Multicast-Pakete auch tatsächlich benötigen. In gleicher Weise wird die *Graft*-Meldung verwendet, um neue geziehlt und schnell neue Äste einem Baum hinzuzufügen, wenn sich in einem Subnetz ein neuer Benutzer einer Gruppe anschließt.

Dadurch, daß jeder Router für jedes Sender-/Empfängerpaar und jede Prune-Nachricht einen Eintrag in einer Tabelle anlegen und speichern muß, wird viel Speicher in den Routern benötigt und die Entscheidungsfindung nimmt aufgrund großer Tabellen an Komplexität zu. Um diesem Problem aus dem Weg zu gehen und die Tabelleneinträge nicht unendlich lange halten zu müssen, verzichtet man auf etwas Bandbreite und versieht die Prune-Nachrichten mit einer

Zeitmarke. Läuft diese ab, verliert die Nachricht ihre Gültigkeit, und der abgeschnittene Ast wird wieder an den Baum angehängt, es werden also periodisch tote Äste neu geflutet.

Aufgrund des Protokollablaufs ist nun auch klar, warum DVMRP ein Protokoll für dicht besetzteMulticast-Netze ist: sind von insgesamt sehr vielen vorhandenen Subnetzen nur sehr wenige an einer Gruppenkommunikation beteiligt, so wird durch das Fluten eine immense, fast aber völlig unnötige Netzlast erzeugt, insbesondere nach dem zyklischen Verwerfen der Prune-Nachrichten.

### 3.2.2 Multicast Open Shortest Path First

Das *Multicast Open Shortest Path First* ist eine Multicast-Erweiterung des *Open Shortest Path First*-Algorithmus für Punkt-zu-Punkt-Routing, welches ein Link State Protokoll ist. Es dient zur internen Wegewahl in Domänen, bezeichnet als *IGP (Interior Gateway Protocol).*

OSPF tauscht Verbindungsinformationen (Link-State Information) über sog. *LSA*s *(Link-State Announcements)* zwischen Routern aus. Dadurch kennt jeder Router alle Verknüpfungen des Netzes, und durch Hinzufügen von Multicast-Informationen zu diesen LSAs wird das Netzwerk multicastingfähig. Diese speziellen LSAs heißen *Group-Membership LSAs* und dienen dazu, die optimale Route zwischen dem Router und der Gruppe zu finden. Die Router, welche die Gruppenverwaltung übernehmen, senden durch Fluten diese Mitgliedschaftshinweise an alle anderen MOSPF-Router im Netzwerk.

Wird das erste Multicast-Paket empfangen, wird sein Absender mit Hilfe der Link-State-Datenbank und der LSAs ausfindig gemacht. Aus diesen Informationen kann ein minimal spannender Baum erstellt werden. LSAs werden ausgewertet, um Pruning durchzuführen und im Baum wirklich nur solche Subnetze zu berücksichtigen, die tatsächlich Empfänger der Multicast-Pakete enthalten. Es entsteht ein beschnittener, minimaler Baum, dessen Wurzel im Subnetz des Senders liegt. Da alle Informationen in der OSPF Datenbank schon enthalten sind, muß der Router, abgesehen von den LSAs, keine Pakete fluten, das Netzwerk ist schwach durch Redundanz belastet.

MOSPF bietet außerdem Routing zwischen mehreren OSPF-Domänen an, indem es sogenannte *ABR (Area Border Routers)* beschreibt, die als Gateways mehrere Subnetze miteinander verbinden. Diese tauschen untereinander Gruppenmitgliedschaftsinformationen und Multicast-Datenpakete aus.

Da MOSPF nur eine Erweiterung von OSPF ist, bietet sich seine Anwendung in Netzen an, deren Routing sowieso schon durch OSPF gelöst ist. Problematisch wird es, wenn MOSPF in einem Netzwerk mit vielen Gruppen, die sich häufig ändern, eingesetzt wird. Es empfiehlt sich dann, das Netz in „Areas“ aufzuteilen, die durch ABRs verbunden werden.

### 3.2.3 Protocol Independent Multicast – Dense Mode

Das *Protocol Independent Multicast – Dense Mode (PIM-DM)* ähnelt in seiner Funktion sehr dem DVMRP. Anfangs erfolgt die Vermittlung über einfaches Fluten, und Äste, in denen sich keine Multicast-Empfänger befinden, werden abgeschnitten. Der Hauptunterschied ist der, daß es kein eigenes Protokoll verwendet, um den kürzesten Weg zurück zu einer Quelle zu finden, so wie beispielsweise DVMRP, sondern auf einem existierenden Unicast-Protokoll aufbaut, daher ist es protokoll-unabhängig.

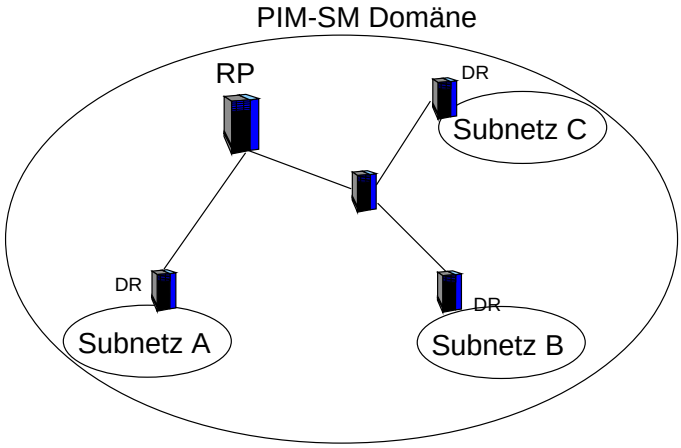


Abbildung 7: Architektur einer PIM-SM-Domäne

### 3.2.4 Protocol Independent Multicast – Sparse Mode

Das *Protocol Independent Multicast – Sparse Mode* wurde entwickelt, um unabhängig von einem zugrundeliegenden Unicast-Protokoll Multicast-Pakete in schwach besetzten Netzen zu vermitteln. Schwach besetzt heißt, daß entweder die Anzahl der Subnetze mit Gruppenmitgliedern niedrig ist relativ zur Gesamtzahl der Subnetze, so daß die Gruppenmitglieder ein Gebiet überspannen, das aufgrund der Ausdehnung kaum zu kontrollieren ist oder daß das Netz, in dem sich die Gruppe befindet, so knapp an Ressourcen ist, daß man sich die durch das Fluten bedingte Redundanz, die traditionelle Protokolle mit sich bringen, nicht leisten kann. Eine weitere Forderung an das Protokoll ist, die Zusammenarbeit mit einem herkömmlichen Protokoll wie DVMRP oder MOSPF zu ermöglchen. Dies kann dann benötigt werden, wenn sich innerhalb von Domänen viele Gruppenmitglieder befinden, so daß dort beispielsweise DVMRP verwendet wird, es aber nur wenige weit verstreute Domänen gibt, die an der Gruppenkommunikation teilnehmen möchten, die also über PIM-SM verbunden werden können. Diese Aufgabe der Kombination unterschiedlicher Protokolle nach innen und außen kommt den *Border Routern (BR)* zu, die sich am Rand der Domänen befinden. Definiert wird das Protokoll in [Deer98] und [Estr98].

Der grundlegende Unterschied zu den oben genannten Protokollen ist der, daß anfangs nicht davon ausgegangen wird, daß sich an allen angeschlossenen Leitungen Empfänger befinden und einfach geflutet wird, sondern jeder Multicast-Empfänger muß sich erst anmelden, um die Gruppendaten zu erhalten. Dafür muß er seinen Wunsch zur Gruppenmitgliedschaft dem *Designated Router* (DR) seines Subnetzes, dem Router mit der höchsten IP-Adresse, melden. Dieser nimmt in seine Routing-Tabelle einen Eintrag der Form (*,G) auf, was bedeutet, daß er ein Mitglied der Gruppe G in seinem Netz hat, das von jedem Sender die Daten empfangen möchte, die an die Gruppe G adressiert ist. Der DR sendet an den *Rendezvous Point* (RP) der Gruppe eine Join/Prune-Meldung, so daß dieser über die neue Mitgliedschaft informiert wird (s. Abb. 7). Router, die zwischen DR und RP liegen, tragen diese Informationen ebenfalls in ihre Routingtabelle ein, so daß sie, falls Daten von einem Sender an die Gruppe gesendet werden, diese auch tatsächlich an den DR des neu angemeldeten Empfängers vermitteln.

Die Wurzel des Routing-Baumes liegt also im RP, und dieser Baum ist nicht unbedingt minimal spannend. Dies hat aber den Vorteil, daß nicht für jeden Sender ein eigener Baum benötigt wird, sondern daß die Rolle der angeschlossenen Systeme zwischen Sender und Empfänger wechseln kann, ohne daß ein neuer Baum gebildet werden muß. Möchte nun ein Sender Multicast-Daten senden, so schickt er die Daten an seinen DR, der diese in eine PIM-Register Meldung verpackt und per Unicast an den RP sendet. Der RP weist den DR an, zukünftig Multicast-Pakete zu schicken, d.h. nur das erste Paket wird getunnelt. Der RP-Router kann

die Pakete anhand der gespeicherten Gruppeninformationen an die *Downstream-Router*, also diejenigen Router, die auf einem Weg zu einem Gruppenmitglied liegen, weiterleiten. Ist die Datenrate des Senders hoch, so kann ein senderspezifischer, optimierter Baum nötig sein. In diesem Fall antwortet der RP mit einer Join/Prune-Nachricht, welche die zwischen RP und DR liegenden Router dazu veranlaßt, einen Pfad von der Quelle zum RP aufzubauen. Möchten die Last-Hop-PIM-Router (normalerweise sind dies die DR) diesem senderspezifischen Baum beitreten, senden sie eine Join/Prune-Nachricht an den Sender, und in die Routing-Tabellen wird ein neuer Eintrag der Form (S,G) eingetragen. In Abb. 7 wird dies verdeutlicht. Liegt der Sender in Subnetz B und ein Empfänger in Subnetz B, so läuft der Transport anfangs über den RP. Wenn nun dieser neue, senderspezifische Baum eingerichtet wurde, kann der Router zwischen dem DR in Netz B und dem RP dem RP-Router eine Join/Prune-Nachricht senden und den Ast zum RP damit abschneiden. Es existiert somit ein minimal spannender Baum zwischen B und C.

### 3.2.5 Core Based Trees

Dieser Ansatz verfolgt im wesentlichen die gleichen Ziele wie das bereits beschriebene PIM-SM, also Routing in einem schwach besetzten oder weit verteilten Netzwerk (Sparse Mode) zu unterstützen. Wie beim PIM-SM gibt es hier einen Router, der als „Treffpunkt“ zwischen Sendern und Empfängern fungiert. In *Core Based Trees (CBTs)* heißt er *Core Router* oder einfach *Core* und hat die gleiche Aufgabe wie der RP beim PIM-SM. Die Gruppenmitgliedschaft wird über das *Internet Group Management Protocol (IGMP)* (siehe Kap. 2.2) organisiert. Die vollständige Spezifiaktion von CBT ist durch [Ball98] gegeben.

Möchte ein Host einer Multicast-Gruppe beitreten, sendet er eine IGMP-Nachricht an den nächsten Router. Dieser lokale CBT-Router sendet daraufhin eine CBT JOIN_REQUEST-Meldung an den nächsten Router, der auf dem Weg zum Core-Router liegt. Dieser muß die Anfrage mit einer JOIN_ACK-Nachricht bestätigen. Auf diese Weise wird eine bidirektionale virtuelle Verbindung zwischen Core und Empfänger hergestellt. Auf Wunsch kann diese Verbindung allerdings unidirektional aufgebaut werden.

Dieses Beitreten zu einem Multicastbaum erzeugt in dem Ausgangs-Router genau wie in allen anderen Routern zwischen ihm und dem Core Router einen Übergangszustand (*Transient State*) der Form <Gruppe, (Quelle), Adresse des Abwärtsstroms, Adresse des Aufwärtstroms>. Die Quelle ist optional und nur für quellspezifische Steuer-Nachrichten nötig.

CBT-Router müssen ein Weiterleitungs-Cache (*Forwarding Cache*) implementieren, der sowohl quellenspezifische (also (S, G)), als auch unspezifische (d.h. (*, G) und (*, Core)) Routing-Einträge umfaßt. Diesen bezeichnet man als privaten Weiterleitungs-Cache des Routers (*Router's Private CBT Forwarding Cache – PFC*). Alle Implementierungen sollten auch einen verteilten, das heißt protokollunabhängigen, Cache bereitstellen, der ausschließlich von den Grenzroutern (Border Routers – BR) genutzt und von allen Protokollen verwendet wird, die auf dem BR eingesetzt werden.

In Broadcast-Netzen, in denen mehrere Router als Gateways das Netz mit anderen Netzen verbinden, wo also beispielsweise Ethernet-Teilstränge über Router verbunden sind, muß sichergestellt werden, daß für den Upstream, also für die Übertragung vom Netz zum Core-Router, immer der gleiche Border-Router verwendet wird, da sonst Schleifen im Baum entstehen können. Dieser eindeutige Router heißt, wie bei PIM-SM, *Designated Router (DR)*. Er wird nicht automatisch über die Höhe der IP-Adresse bestimmt, sondern über einen Auswahlprozeß gewählt, der einen einzigen Router zum DR macht. Dies funktioniert über das CBT-HELLO-Protokoll. Ein Router repräsentiert seinen Status als DR bzw. nicht-DR über das DR Flag, das entweder den Wert wahr oder falsch annimmt. Anfangs ist dieses Flag bei allen Routern auf falsch gesetzt, so daß ein DR bestimmt werden muß. Die „Ambition“ DR

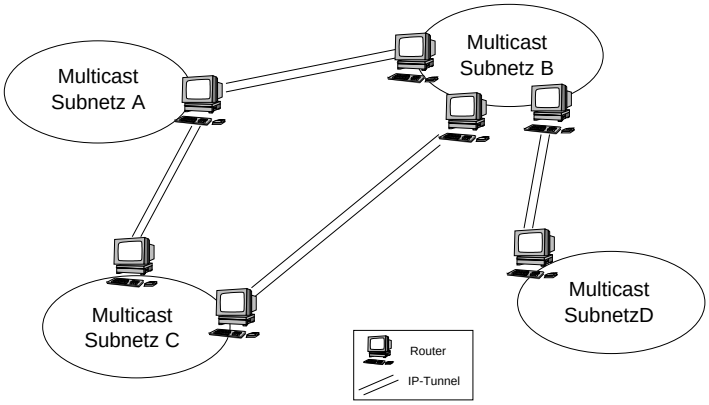


Abbildung 8: Tunneln im MBone

zu werden, kann über eine HELLO-Präferenz in den vom Router gesendeten HELLO-Paketen gesetzt werden. Mögliche Werte liegen zwischen 1 (hohe Präferenz) und 255. Der gewählte DR versendet HELLO-Pakete mit dem Wert 0. Wird ein Router gestartet, sendet er über jedes angeschlossene Broadcast-Medium zwei HELLO-Pakete, während sein DR-Flag noch den Wert „falsch“ hat. Wenn nach dem Senden eines HELLO-Paketes kein „besseres“ HELLO-Paket empfangen wird, also keines, mit einer niedrigeren Präferenz, dann wird der Router zum DR, setzt das DR-Flag auf „wahr“ und sendet periodische HELLO-Pakete mit dem Wert 0. Er bleibt für seine gesamte Lauzeit DR. Empfängt auf der anderen Seite ein Router ein HELLO-Paket mit einer niedrigeren Präferenz als seine eigene, so beantwortet er das Paket nicht. Liegt seine eigene Präferenz höher als die des empfangenen Pakets, so sendet er sein eigenes HELLO-Paket an das Teilnetz, auf dem er die HELLO-Nachricht empfangen hat.

# 4 Das MBone

Bereits in den 80er Jahren wurde Multicast-IP in der Internetgemeinde beschrieben und spezifiziert. Anwendungen kamen allerdings erst 1992 mit der Einführung des MBone auf, der Verbindung einiger lokaler Multicast-Netze über IP-Tunnel (Abb. 8). Tunneln bedeutet, daß Multicast-IP-Pakete, die zwischen zwei Multicast-Netzen transportiert werden sollen, vom Border-Router des Senders in den Nutzdatenteil normaler IP-Pakete verpackt werden und durch das Netz transportiert werden, bis sie vom BR der Zielgruppe wieder „ausgepackt“ werden und unter Verwendung der Multicast-Routing-Protokolle im Subnetz weitergeleitet werden. DVMRP hat Methoden zum Tunneln als Teil des Protokolls eingebaut, da es, wie bereits erwähnt, das älteste Multicast-Protokoll ist und somit auch im MBone eingesetzt wird.

Anfangs arbeiteten sämtliche Router im MBone mit einer Software namens *Mrouted*, einem Multicast-Routing-Daemon. Mrouted implementierte DVMRP, allerdings anfangs ohne *Pruning*, so daß es sich nicht um Multicast sondern um Broadcast innerhalb des MBone handelte. Dieses Problem wurde mit neueren, Pruning-unterstützenden Implementierungen von Mrouted gelöst, allerdings gibt es immer noch eine geringe Anzahl Router ohne Pruning. Die heutige Verbreitung des MBone in Deutschland ist in Abb. 9 dargestellt. Weitere Informationen über das MBone und dessen Verbreitung, insbesondere in Deutschand, gibt es unter *http://www.mbone.de*.

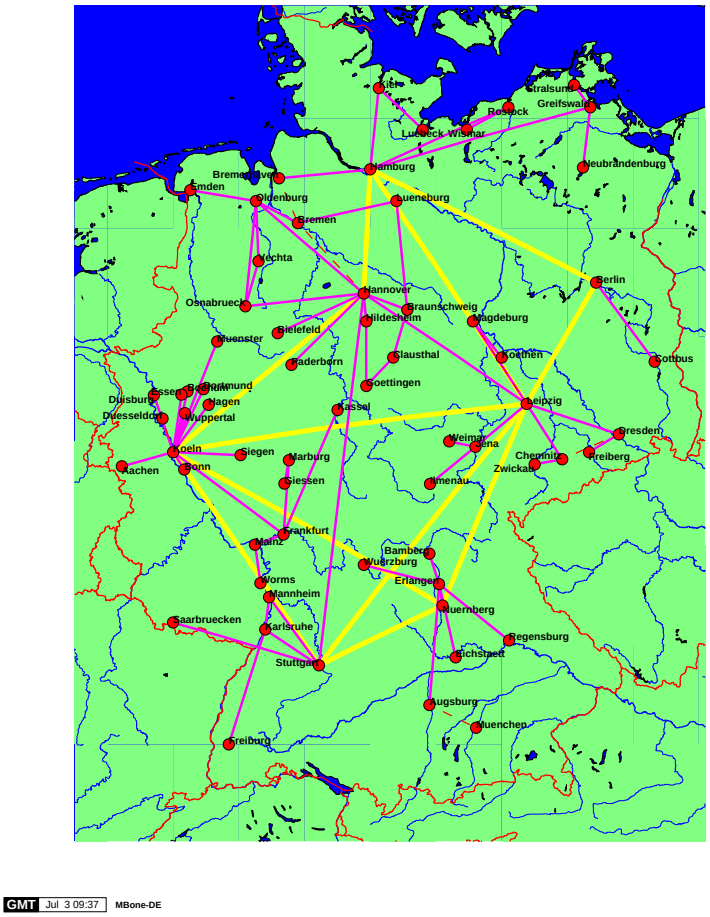


Abbildung 9: MBone-Verteilung in Deutschland

# 5 Zusammenfassung

Die wachsende Bandbreite und zunehmende Verbreitung des Internet ermöglicht den Einsatz einige neuer, multimedialer Anwendungen, wie es bisher kaum denkbar war. Diese Anwendungen bringen allerdings meist das Problem mit sich, daß nicht mehr nur zwei Endsysteme, sondern ganze Gruppen von Systemen miteinander kommunizieren, was die Entwicklung neuer Verfahren für die Wegewahl von Datenpaketen durch das Netz erfordert.

Auf der Ebene der Subnetze wird das Problem über ein Protokoll zur Feststellung von Gruppenmitgliedschaften, dem *Internet Group Management Protocol (IGMP)*, gelöst. Durch IGMP wissen die Grenz-Router der Subnetze, von welchen Gruppen Empfänger an sie angeschlossen sind.

Ein Multicast-Routing-Verfahren ermöglicht nun, diese Grenzknoten miteinander zu verbinden und somit Daten an jedes Mitglied einer Gruppe weiterzuleiten. Ein wichtiges Entscheidungsmerkmal für die Wahl des Verfahrens ist die Verteilung der Gruppenmitglieder: gibt es sehr viele Gruppenmitglieder die beispielsweise eine große Konferenz im Netz verfolgen, muß ein anderes Verfahren gewählt werden, als wenn über Europa verteilt vier Teilnehmer eine Videokonferenz durchführen.

Als Prüfstand für neue Verfahren und Protokolle dient zur Zeit das MBone, das einige „Multicast-Inseln“ durch Tunnel über das Internet verbindet. Es wurde 1992 in Betrieb genommen und erfreut sich ähnlich großer Wachstumszahlen wie das restliche Internet.

## Literatur


[Ball98] A. Ballardie. Core Based Trees (CBT Version 3) Multicast Routing: Protocol Specification, August 1998. IETF Internet-Draft.

[Cain99] B. Cain. Internet Group Management Protocol Version 3, Februar 1999. IETF Internet-Draft.

[Deer98] S. Deering. Protocol Independant Multicast-Sparse Mode (PIM-SM): Motivation and Architecture, August 1998. IETF Internet-Draft.

[Estr98] D. Estrin. Protocol Independant Multicast-Sparse Mode (PIM-SM): Protocol Specification, Juni 1998. RFC2362.

[Mill99] C. Kenneth Miller. *Multicast Networking and Applications*, Kapitel 3, S. 19–50. Addison-Wesley. 1999.

[Pusa99] T. Pusateri. Distance Vector Multicast Routing Protocol, Februar 1999. IETF Internet-Draft.

[Tane97] Andrew S. Tanenbaum. *Computernetzwerke*, Kapitel 5.2.5. Prentice Hall. 1997.

# Ressourcenreservierung für Differentiated Services

Alexander Lange

**Kurzfassung**

Viele Anwendungen im heutigen Internet besitzen zeitkritische Anforderungen und fordern deshalb gewisse Garantien von den Netzwerkdiensten. Eine Möglichkeit, Dienstgüte in das *Internet Protocol* zu integrieren, ist die Differentiated-Services-Architektur. Innerhalb der Differentiated-Services-Architektur sind verschiedene Dienste vorgesehen, darunter der Premium Service, welcher einen Dienst anbietet, der vergleichbar mit einer Virtual Leased Line ist. Die Funktionalität der Differentiated-Services-Architektur wird durch verschiedene Klassen von Routern beschrieben. Um die Ressourcen innerhalb einer Domäne zu reservieren und zu verwalten, werden drei Lösungen vorgestellt.

## 1 Einleitung und Problemstellung

### 1.1 Quality of Service

Viele Anwendungen im Internet verlangen heutzutage eine bestimmte Dienstgüte, die über die bisherigen Möglichkeiten des Best-Effort-Dienstes hinausgehen. Zeitkritische Anwendungen können nur korrekt arbeiten, wenn Paketverlust und Verzögerungen innerhalb bestimmter Grenzen liegen. Solche Echtzeit-Anwendungen sind häufig im Multimedia-Bereich anzutreffen, z.B. Video- oder Audioübetragungen. Besonders interaktive Anwendungen tolerieren keine Zeitverluste, die durch Schwankungen in der Paketlaufzeit oder durch wiederholtes Übertragen von Paketen entstehen. Anbieter solcher Dienste verlangen vom Netzbetreiber eine Qualitätsgarantie, die eingehalten werden muß. Die gültigen Kriterien werden zwischen Anbieter und Netzbetreiber (Internet Service Provider, ISP) in einem *Service Level Agreement (SLA)* festgeschrieben. Zusätzlich müssen die Dienste skalierbar sein, um sich den Änderungen in der Anzahl der Benutzer anzupassen.

Die Möglichkeiten, Pakete mit Prioritäten zu versetzen, ist bereits im Internet Protocol (IP) v4 vorgesehen. Dazu ist in jedem Paketkopf ein *Type of Service*-Feld (TOS) definiert, welches jedoch nie wirklich benutzt wurde, da für den Anwender weder tatsächliche Ressourcen reserviert wurden noch der ISP besondere Kosten abrechnen konnte. Die Differentiated-Services-Architektur ersetzt das TOS-Feld des IP v4 und das Service-Feld des IP v6 und definiert diese zum Differentiated-Service-Feld um.

### 1.2 Zwei Ansätze zur Sicherung von Dienstgüte

Ein erster Ansatz zur Integration von Dienstgüte ins Internet war die Architektur der *Integrated Services (IntServ).* Dazu wurden Anwendungen untersucht und vier Dienstklassen eingeführt:

1. Garantierte Diente für intolerante Echtzeitanwendungen

2. Vorhersagbare Dienste für tolerante Echtzeitanwendungen
3. Dienste mit kontrollierter Last für adaptive Echtzeitanwendungen
4. Der garantielose bestmögliche Dienst für elastische Anwendungen (*Best-Effort*-Prinzip)

Bei diesem Modell müssen in jedem einzelnen Router des Netzwerkpfades Reservierungsinformationen gehalten werden, was einen hohen administrativen Aufwand bedeutet und auf Veränderungen unzuverlässig reagiert. Das Modell der Integrated Services hat sich nicht durchgesetzt; ein Hauptgrund für das Scheitern war die fehlende Skalierbarkeit und die damit verbundene Überlastung der Router mit Verwaltungsaufgaben. Für jede einzelne Verbindung müssen Reservierungsdaten gespeichert, kontrolliert und periodisch aufgefrischt werden, was bei stark belasteten Routern zu geringer Effizienz bei der Bewältigung der eigentlichen Arbeit führt. Außerdem fehlt es an Möglichkeiten zur Authentifizierung, Zugangskontrolle und Abrechnung [Wehr99]. Zur Signalisierung der Reservierungsanfragen wird bei der Integrated-Services-Architektur das *Ressource Reservation Protocol (RSVP)* verwendet.

Die hier vorgestellte Architektur der *Differentiated Services (DiffServ, DS)* ist ein Ansatz, der sich gerade in der Entwicklung befindet und in der Praxis erprobt wird. Ein Ziel während der Entwicklung war es, vorhandene Router und Software nicht obsolet zu machen und die erforderlichen Modifikationen gering zu halten. Die komplexen Funktionen innerhalb einer DS-Domäne wird an den Grenzen des Netzes durchgeführt, was zu einer einfacheren Verwaltung innerhalb beiträgt.

Als besonders interessanter Dienst der Differentiated-Services-Architektur hat sich der auf Expedited-Forwarding basierende *Premium Service* erwiesen [Wehr99]. Er entspricht weitestgehend dem garantierten Dienst des Integrated-Services-Modells und garantiert dem Dienstnutzer eine konstante Übertragungsrate bei minimaler Verzögerung. Damit eignet er sich für zeitkritische Echtzeitanwendungen. Im Unterschied zum verwaltungslastigen Modell der Integrated Services betrachtet der Premium Service nur *aggregierte Datenströme*, die sich aus einzelnen Strömen einer Dienstkategorie zusammensetzen. Eine kostengünstigere Alternative ist der *Assured Service*, welcher nur eine auf Wahrscheinlichkeiten basierende statistische Garantie bietet und sich deshalb nicht für zeitintolerante Anwendungen eignet.

Es ist nicht zu erwarten, daß ein Großteil des Internet-Verkehrs nun plötzlich den Premium-Dienst benutzen wird. Die meisten Anwendungen im Netz wie E-Mail, FTP oder WWW sind unkritisch gegenüber tragbaren Verzögerungen und werden weiterhin den kostengünstigen Best-Effort-Dienst beanspruchen. Dennoch entsteht ein stärker werdendes kommerzielles Interesse an garantierten Diensten, und es ist notwendig für Internet Provider, einen solchen Service anzubieten. Eine anschauliche Analogie ist der Vergleich mit Fluggesellschaften: Flüge lassen sich sowohl in der Tourist-Class als auch in der First-Class buchen. Obwohl der Großteil des Umsatzes mit der einfachen Tourist-Class gemacht wird, gibt es Kunden, die die speziellen Leistungen der First-Class wünschen [NiJZ99].

# 2 Differentiated Services

## 2.1 Ressourcenreservierung und Kontrolle

Bei der Entwicklung der Differentiated-Services-Architektur wurde darauf geachtet, die oben angesprochenen Nachteile der Integrated-Services-Architektur zu vermeiden, ohne jedoch den Anspruch der Dienstgüte aufzugeben. Die wichtigsten Neuerungen gegenüber den Integrated Services sind die folgenden:

- Paketströme werden innerhalb der Netze nicht mehr einzeln betrachtet, sondern zusammengefaßt (aggregiert). Eine Unterscheidung erfolgt innerhalb der Domäne nur noch anhand der Dienstkategorie, welche durch das DS-Feld im IP-Header beschrieben wird. Eine aufwendige Klassifikation der Pakete nach IP-Adressen und Ports von Sender und Empfänger wird nur im ersten Router (First-Hop-Router) durchgeführt; die inneren Router werden entlastet. Das DS-Feld besteht aus sechs Bits für die Kennzeichnung des Diensttyps, die *Codepoints* genannt werden, und zwei weiteren reservierten Bits. Dies ergibt $2^6 = 64$ Möglichkeiten für die Codepoints. Ein Dienst kann mehrere Codepoints beanspruchen, um z.B. Prioritäten zu unterscheiden.

- Die Reservierungsdaten werden nur noch in den Grenzroutern verwaltet und gespeichert. Der Grenzrouter speichert zu jeder Kombination von Eingangs- und Ausgangsadapter und Dienstkategorie eine Reservierungsinformation. Auch dies führt zu einer Entlastung der inneren Router.

- Reservierungen sollen längerfristig bestehen. Die Reservierungen werden durch Dienstverträge festgelegt, wobei keine kurzfristigen Änderungen durch ein Signalisierungsprotokoll vorgesehen ist.

Eines *Differentiated-Service-Domain (DSD)* ist ein Teilnetz des Internet. So ein Netz besteht im wesentlichen aus verschiedenen Arten von Routern, die sich folgendermassen unterteilen lassen:

- *First-Hop-Router:* Dieser Router stellt die Verbindung zwischen Workstation und Netzwerk dar. Hier werden die Dienstkategorien der IP-Pakete bestimmt und das DS-Feld der Pakete gesetzt. Dies geschieht nicht beim Sender. Einerseits ist damit keine Änderung der Software des Senders erforderlich, weil die Endsysteme durch die DS-Architektur nicht verändert werden sollen. Andererseits kann ein Sender keinen Dienst in Anspruch nehmen, für den er nicht bezahlt hat. Im First-Hop-Router sind Verkehrsprofile gespeichert, welche die Charakteristik einer zwischen Netzbetreiber und Dienstnehmer vereinbarten Dienstkategorie beschreiben. Der Codepoint des IP-Pakets bestimmt dessen weitere Behandlung *(Per-Hop-Behaviour)* in den folgenden Routern.

- *Interior Router* (innere Router): Diese leiten die Pakete innerhalb einer Domäne weiter. Innere Router sind sehr einfach aufgebaut und führen keine Klassifizierung der einzelnen Ströme durch. Sie führen keine Infomationen über Verkehsprofile und behandeln nur aggregierte Ströme.

- *Border Router* (Grenzrouter): Sie bilden die Grenzposten zu anderen Domänen. Domänen sind untereinander immer nur durch Grenzrouter verbunden. Auch hier sind wie im First-Hop-Router Verkehrsprofile gespeichert, nach denen eine Klassifizierung vorgenommen wird. Ein Dienst, dem kein Verkehrsprofil zugeordnet werden kann, wird nach dem Best-Effort-Prinzip behandelt. Zur Verwaltung der Verkehrsprofile stehen die unter Abschnitt 3 genannten Vorschläge zur Verfügung.

Um den Verkehr innerhalb einer Domäne zu kontrollieren und zu steuern, sieht die DS-Architektur verschiedene Komponenten innerhalb der Router vor.

- *Packet-Classifier* (Klassifizierer). Ein Klassifizierer wählt für jedes Paket, welches in dem Router ankommt, das entsprechende Verkehrsprofil aus.

  Bei der einfachen *Codepoint-Klassifikation (CP)* wird das Profil gemäß dem Codepoint im DS-Feld ausgewählt. Hier wird die oben erwähnte *Aggregation* benutzt, denn alle

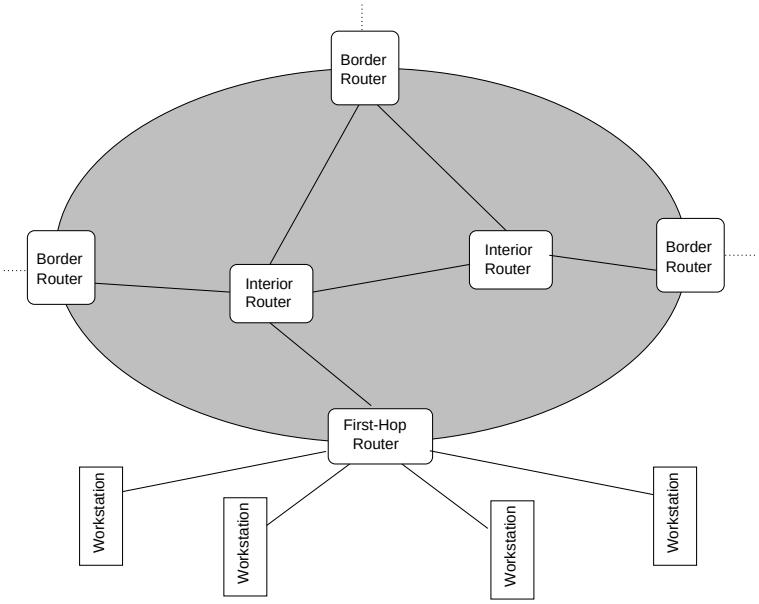


Abbildung 1: Die Struktur einer Differenatiated Services Domain.

Ströme des selben Diensttyps werden wie ein einziger Strom behandelt. Ein Router, der nach dem Codepoint klassifiziert, muß also höchstens 64 Verkehrsprofile speichern.

Eine andere Möglichkeit ist die *Multifield-Klassifikation (MF),* die den Codepoint nicht benutzt. Stattdessen werden die Pakete nach verschiedenen Feldern im IP- und TCP-Paketkopf unterschieden. Relevante Felder sind hier IP-Quell- und IP-Zieladresse. Die MF-Klassifikation ist aber insofern schlechter, daß hier das selbe Problem wie bei der Integrated-Services-Architektur auftritt: Die Anzahl der Verkehrsprofile kann mit der Anzahl der passierenden Ströme stark anwachsen.

Normalerweise wird deshalb die Klassifikation über Codepoints verwendet. Der Codepoint kann sich unterwegs ändern, wenn zwei Provider für einen Dienst unterschiedliche Codepoints benutzen, wenn ein Dienst von einem Provider nicht unterstützt wird und über einen ähnlichen Dienst erbracht werden muß, oder wenn ein Paket als *Out-of-Profile* eingeordnet und mit dem schlechtesten Dienst weitertransportiert wird.

- *Packet-Marker* (Markierer). Der Markierer wertet das Ergebnis des Verkehrsmeters (s.u.) aus und setzt entsprechend einen Codepoint. Ist das Paket in-profile, so wird ein Codepoint gemäß dem gespeicherten Verkehrsprofil gesetzt; bei out-of-profile wird entweder ein Dienst geringerer Priorität (meist Best-Effort) verwendet oder das Paket wird gleich verworfen.

- *Traffic-Meter* (Verkehrsmeter). Die Aufgabe eines Verkehrsmeters besteht darin, die im Verkehrsprofil vereinbarten Dienstgüte-Paramter für den aggregierten Datenstrom zu messen. Beim Assured Service werden die Pakete entweder als *In-Profile* oder als *Out-of-Profile* klassifiziert.

  Es existieren verschiedene Techniken, von denen hier zwei genannt werden. Bei der Methode des *Token Bucket* wird die Rate des eingehenden Paketstroms gemessen. Pakete, die über der im Verkehrsvetrag *(SLA)* vereinbarten Rate liegen werden als out-of-profile gekennzeichnet und mit dem geringsten Dienst weitertransportiert. Die Token des Bucket werden mit der Frequenz der Senderate erzeugt; die Größe des Bucket entspricht der erlaubten Burstrate. Um ein Paket von $n$ Byte senden zu können, müssen mindestens $n$ Token im Bucket enthalten sein; bei unzureichender Anzahl an Token wird das Paket als out-of-profile klassifiziert. Der Vorteil hier gegenüber dem Leaky Bucket

besteht darin, daß bei geringem Verkehr Token (= Berechtigungen) aufgespart werden, die dann für plötzlich auftretende Bursts verwendet werden können.

Beim *Leaky Bucket*-Modell werden auch alle Pakete, die über der erlaubten Rate und dem vereinbarten Burst liegen, als out-of-profile weitergeleitet. Außerdem fungiert der Leaky Bucket als Verkehrsformer, wodurch alle Pakete den Router in der vereinbarten Rate verlassen. Er arbeitet wie ein „löchriger Eimer", d.h. Pakete werden gepuffert und in einer konstanten Ausgangsrate als in-profile klassifiziert weitergegeben. Ist der Puffer voll, werden die neu eintreffenden Pakete als out-of-profile klassifiziert.

- *Traffic-Shaping* (Verkehrsformung). Zur Anpassung der Ausgangsrate des Routers an das vereinbarte Profil wird ein Verkehrsformer eingesetzt. Die einfachste und am meisten verwendete Methode ist das *Verwerfen* von Paketen, die als out-of-profile eingeordnet wurden. Etwas gnädiger ist es, das Paket zu *degradieren* und es mit einer schlechteren Dienstklasse weiterzuleiten. Bei dem oben genannten *Leaky Bucket Shaper* werden Bursts von Paketen abgefangen und in einer geglätteten Rate wieder ausgegeben. Damit wird vermieden, daß die Pakete vielleicht vom nächsten Provider als out-of-profile verworfen werfen.

### 2.2 Probleme bei der Multicast-Kommunikation

Will man IP-Multicast innerhalb einer Differentiated-Services-Domäne verwenden, ergeben sich durch die oben beschriebene Struktur einige Probleme. Zunächst einmal erfahren die Pakete nur stromabwärts den Dienst, dem sie angehören, denn der Codepoint wird ja durch den First-Hop-Router gesetzt. Der Absender bildet hierbei ein Kriterium für die Einordnung des Paketes und die Klassifizierung. Will ein Empfänger einer Multicastgruppe beitreten, muß im First-Hop-Router des Senders ein geeignetes Verkehrsprofil eingerichtet werden und Ressourcen entlang des Netzwerkpfades reserviert werden. Zusätzlich muß eine Zugangskontrolle stattfinden, damit nicht jeder beliebige Empfänger einer Gruppe beitreten kann. Die Abrechnung geht meistens zu Lasten des Empfängers und muß sichergestellt werden. Da es bisher keine Qualitätsgarantien für Dienstgüte im Netz gab, existiert auch keine Infrastruktur, um die Administration sicherzustellen [BlWe99].

Wie bereits beschrieben, sind die inneren Router einer Domäne nach der Differentiated-Services-Architektur sehr einfach aufgebaut. Sie speichern keine Daten über Verkehrsprofile und leiten lediglich die Pakete weiter. Tritt nun ein Empfänger innerhalb der Domäne einer Gruppe bei und wird dabei einfach der Multicastbaum erweitert, ohne daß die tatsächliche Reservierung überprüft wird, kann es auf dem neuen Teilbaum zu einer Überbelegung der Ressourcen kommen. Die Dienstqualität anderer Mehrwertdienste kann sich dadurch verschlechtern. Diese Situation wird als *Neglected Reservation Subtree-Problem (NRSP)* bezeichnet [BlWe99]. Dabei werden andere vereinbarte Dienstverträge (SLA) gefährdet, weil eine Reservierung für einen Dienst mangels Information in den inneren Routern nicht überprüft werden konnte.

## 3 Verwaltung einer Differentiated-Services-Domäne

Um einen Ende-zu-Ende-Dienst innerhalb eines Netzwerkes mit der gewünschten Qualität liefern zu können, müssen einerseits Zugangskontrollen durchgeführt und andererseits die Informationen der Verkehrsprofile in den Grenzroutern bereitgehalten werden. Die Konfiguration der Router kann natürlich zunächst einmal vom Netzwerkadministrator von Hand durchgeführt werden. Dies führt zu einer statischen Reservierung, die jedoch nur in geringem Umfang praktikabel ist. Vielmehr ist eine Automatisierung der Ressourcenreservierung

gewünscht. Die Empfängergruppe eines Multicast-Dienstes ändert sich normalerweise dynamisch, so daß eine ständige Kontrolle und eine laufende Korrektur der Verkehrsprofile nötig ist.

## 3.1 Bandwidth Broker

Der in [NiJZ99] vorgestellte *Bandwidth Broker (BB)* verfolgt eine ähnliche Aufgabe wie der unten erklärte *Domain Manager (DSDM)*, funktioniert jedoch auf einer anderen Ebene. Während der DSDM die Verwaltung ganzer Dienste regelt, verwaltet der Bandwidth Broker lediglich die Ressourcen und hält eine Aufzeichnung der Anfragen bereit.

Das Ziel der Differentiated-Services-Architektur ist es, die Ressourcen eines Netzwerkes kontrolliert und den Bedürfnissen entsprechend auszunutzen sowie Dienstgüte im Netz anzubieten. Eine einfach zu implementierende Methode wäre es, die Benutzer bestimmen zu lassen, welche Priorität ihre Pakete haben. Dies führt aber unweigerlich zu unvernünftigen Angaben und ständiger Benutzung des höchsten Dienstes. Besser ist es dagegegen, einen *Agenten* die Ressourcenzuteilung gemäß einer festgelegten Strategie, die auf den Verkehrsverträgen basiert, machen zu lassen. Diese Agenten werden *Bandwidth Broker* genannt. Es hat sich gezeigt, daß Abkommen zwischen mehrere Netzbetreibern schwierig umzusetzen sind. Deshalb ist es einfacher, ausschließlich bilaterale Verträge aufzusetzen und eine Ende-zu-Ende-Verbindung daraus zusammenzustückeln. Der Bandwidth Broker einer Domäne muß lediglich seinen unmittelbaren Nachbardomänen vertrauen und mit deren Brokern kommunizieren.

Die beiden Aufgaben eines Bandwidth Brokers sind wie folgt:

1. Anfragen *(Requests)*, die für die eigene Domäne bestimmt sind, herauszufiltern und die Router entsprechend zu konfigurieren.
2. Anfragen, die für andere Domänen bestimmt sind, durch das Netz zu benachbarten Brokern leiten.

Werden nun von einer Anwendung Netzwerkressourcen benötigt, so wird eine entsprechende Anfrage an den Bandwidth Broker geschickt. Die Anfrage enthält den Diensttyp, die zu erwartende Übertragungsrate, das Maximum für einen Burst und die Zeitperiode. Diese Anfrage kann vom Netzwerkadministrator manuell eingetragen werden oder bei entsprechender Entwicklung der Software dynamisch erfolgen. Der Bandwidth Broker überprüft daraufhin die Berechtigung des Anfragenden und ob genügend unbenutzte Bandbreite zur Verfügung steht. Stimmen diese Voraussetzungen, wird die verfügbare Bandbreite um die angeforderte reduziert und die Spezifikationen in einer Art interner Ressourcen-Buchhaltung gespeichert. Fließt der Datenstrom außerhalb der eigenen Domäne, so informiert der Bandwidth Broker den benachbarten Broker, daß er etwas von den vertraglich vereinbarten Ressourcen benutzen wird. Die Kommunikation zwischen den Bandwidth Broker muß sicher und robust sein. Diese Information wird periodisch erneuert und bleibt nicht statisch bestehen. Der benachbarte Bandwidth Broker ist für die Konfiguration seiner Grenzrouter verantwortlich, so daß die Pakete das Netz betreten und gegebenenfalls passiern können. Er muß auch für Reservierungen bei weiteren Bandwidth Brokern Sorge tragen. Auf diese Weise sind alle Vereinbarungen zwischen Domänen nur bilateral. Der Bandwidth Broker ist eine logische Einheit und kann entweder zentralisiert oder verteilt aufgebaut sein. Eine konkrete Implementierung steht noch aus [ROTZ+98].

In dem abgebildeten Beispiel will Rechner A aus Domäne 1 Daten zu Rechner B in Domäne 3 übertragen. Dazu sendet er eine Anfrage an den Bandwidth Broker in seiner Domäne, der wiederum die Anfrage an die Transit-Domäne 2 weiterleitet. Da die angefragte Bendbreite

innerhalb der ausgehandelten Kapazität liegt, sendet der Bandwidth Broker aus Domäne 2 eine weitere Anfrage den den Bandwidth Broker in Domäne 3. Die Bestätigung der Reservierung wird an Bandwidth Broker von Domäne 1 und von dort an Rechner A weitergeschickt. A kann also anfangen zu senden.

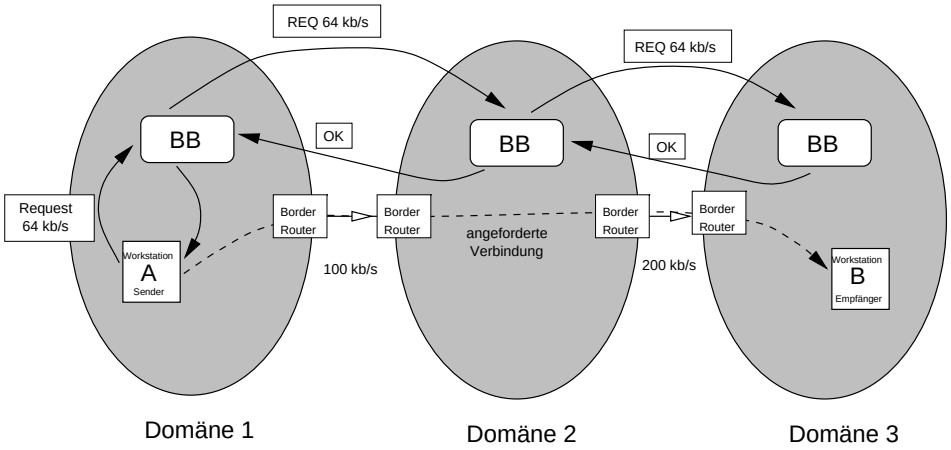


Abbildung 2: Ablauf einer Reservierungsanfrage über verschiedene Bandwidth Broker.

### 3.2 Differentiated Services Domain Manager

Ein Modell für ein integriertes Managment ist der in [BlWe99] vorgestellte *Differentiated-Services Domain-Manager (DSDM).* Dieser besteht aus den folgenden logischen Einheiten:

- *Zugangskontrolle.* Für die einzelnen Dienste muß sichergestellt werden, daß die Summe des Eingangsverkehrs die Bandbreite des möglichen Ausgangsverkehrs nicht überschreitet. Beim Premium Service ergibt sich daraus eine nur geringe Füllung der Warteschlange im Ausgangsadapter. Außerdem muß ständig die reservierte Bandbreite überwacht werden, damit andere Dienste nicht zu stark benachteiligt werden. Der DSDM nimmt auch die Anforderung einer Reservierung entgegen, überprüft die Verfügbarkeit der Ressourcen und leitet die Anfrage dann an den DSDM des nächsten Teilnetzes weiter. Auch politische Aspekte werden vom DSDM an dieser Stelle kontrolliert, denn Anfragen aus bestimmten Domänen können aus Ressourcenmangel oder sicherheitsrelevanten Gründen abgelehnt werden, um Mißbrauch vorzubeugen.

- Der DSDM als zentrale Instanz sorgt für eine optimale *Ressourcenverwaltung.* Da die Software über Informationen zur Netzwerktopologie und der physikalischen Bandbreite der Leitungen verfügt, hat sie stets einen Überblick über den Zustand des Netzwerks in der Domäne und die Auslastung der einzelnen Netzwerkpfade. Die Zuteilung erfolgt über den DSDM, so daß andere Komponenten, insbesondere die inneren Router, von Verwaltungsaufgaben entlastet werden. Grenzrouter kümmern sich allein um die Einhaltung der Verkehrsprofile durch Verkehrsmessung und -formung.

- *Signalisierung.* Empfänger müssen Ressourcen für ihren Dienst anfordern, wobei sich die benötigte Bandbreite jederzeit ändern kann. Idealerweise geschieht die Anforderung direkt vor der Nutzung mittels eines bestimmten Reservierungsprotokolls. Hierfür kann das in [BYFB+99a] beschriebene RSVP benutzt werden. Auch die DSDM benachbarte Domnänen müssen miteinander kommunizieren, um Dienste anzufordern.

- In dem vorgestellten Szenario spielt natürlich auch die *finanzielle Abrechnung* eine große Rolle. Schließlich geht es hier um angeforderte Dienstleistungen, für die bezahlt werden

soll. Da die Verbindungsdaten aller Dienste dem DSDM bekannt sind, kann hier gleich die Dienstabrechung erfolgen. Gleichzeitig muß bei der Anforderung einer Reservierung eine Form von *Authentifizierung* stattfinden, damit Rechnungen durch Nachweise belegt werden können.

Ohne das Vorhandensein eines zentralen DSDM müßten viele der genannten Aufgaben durch die Grenzrouter erledigt werden. Die Instanz des DSDM entlastet also die Router und bringt gleichzeitig eine neue Funkionalität. Im Unterschied zum Bandwidth Broker werden nicht nur die Ressourcen, sondern die Dienste verwaltet. Durch die einheitliche Verwaltung der Ressourcen über den DSDM ist es möglich, verschiedene Reservierungsprotokolle zu verwenden.

Sollte ein DSDM einmal ausfallen, bleiben existierende Reservierungen bestehen; die Verkehrsprofile in den First-Hop- und Grenzroutern sind von dem Ausfall nicht betroffen. Laufende Datenströme erhalten die versprochene Dienstgüte. Der DSDM kann physikalisch durchaus in mehreren Einheiten implementiert werden, um die Ausfallsicherheit zu erhöhen.

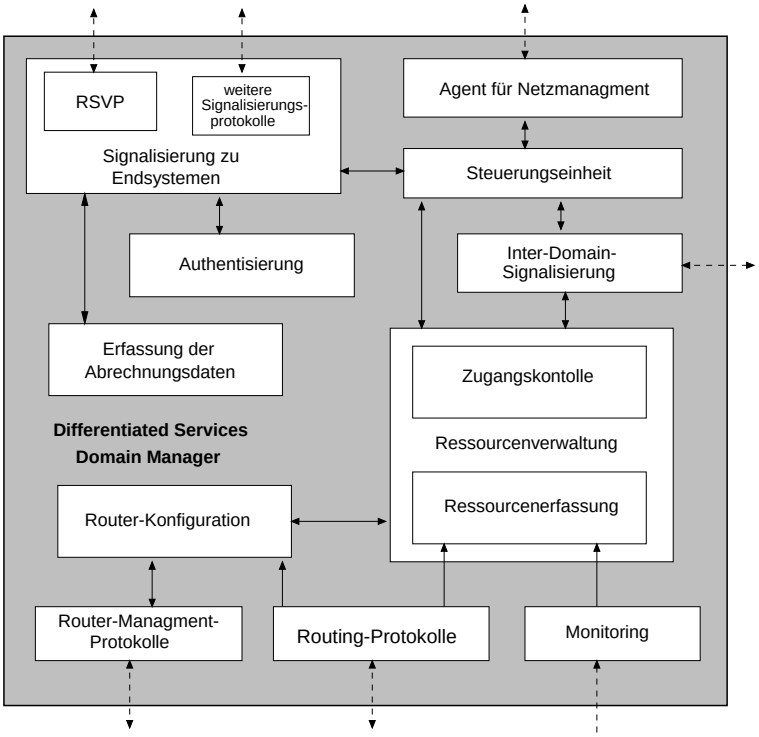


Abbildung 3: Logischer Aufbau des DS Domain Managers.

## 3.3 Interoperation von Integrated und Differentiated Services

Während die Differentiated-Services-Architektur mit ihrer Klassifizierung der Dienste eher qualitätsorientiert ist, ist die starre flußorientierte Integrated-Services-Architektur auf festen Durchsatz (Quantität) ausgelegt. Letztere Eigenschaft wird von manchen Anwendungen eher benötigt als die komplexe Diensteverwaltung. In gewisser Hinsicht ergänzen sich diese beiden Modelle also, so daß sie sich beide innerhalb der selben Domäne einsetzen lassen [BYFB+99b]: Das Modell der Differentiated Services wird im Kern der Transitnetzwerke ingesetzt, das Integrated Services-Modell und RSVP dagegen in den Grenzbereichen des Netzwerks. Aus Sicht der Integrated Services-Router ist das Differentiated Services-Transitnetzwerk nur eine virtuelle Verbindung zwischen zwei IntServ-Routern. Wie stark der DiffServ-Anteil nach außen hin ausgeweitet wird, liegt im Ermessen des Administrators.

Bevor Pakete die DiffServ-Region des Netzwerks erreichen, muß das DS-Feld des Pakets bereits mit dem richtigen Codepoint besetzt sein. Eine Möglichkeit dies zu erreichen besteht darin, im letzten Router der IntServ-Region den Codepoint umzusetzen. Dies führt jedoch zu einer komplexen Konfiguration.

Auch bei diesem Modell ist Kontrolle an bestimmten Punkten des Netzwerks notwendig. Zum Beispiel kann ein First-Hop-Router überprüfen, ob der von einem Rechner ausgehende Verkehr nicht ein bestimmtes Limit übersteigt. Host-Rechner schicken ihre Anfrage mittels RSVP an einen Integrated-Services-Router; danach werden sie transparent durch das Differentiated-Services-Netzwerk bis zum anderen Ende übertragen.

Ein besonders Element dieser Architektur sind die *Edge Routers*. Sie bilden die Grenze zwischen dem IntServ/RSVP- und dem DiffServ-Teil und bestehen quasi aus zwei Hälften. Die Außenseite des innen liegenden DiffServ-Netzes besteht wiederum aus *Border Routers* (wie oben beschrieben). Um einen Integrated- auf deinen Differentiated Services-Dienst abzubilden, wird ein bestimmtes Per-Hop-Behaviour für jeden IntServ-Dienst fest vorgegeben. Die Klassifizierung kann natürlich auch noch feiner spezifiziert werden.

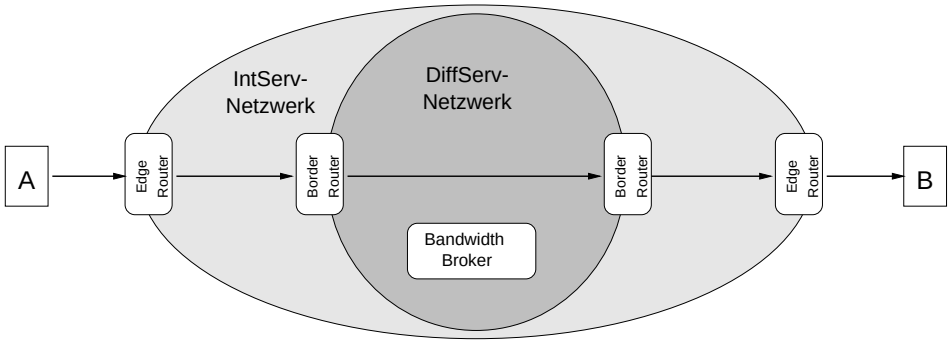


Abbildung 4: Interoperation von IntServ/RSVP und DiffServ innerhalb eines Netzwerks.

# 4 Zusammenfassung

In dieser Ausarbeitung wurde zunächst die Notwendigkeit von Dienstgüte erläutert. Mit der vorgestellten Differentiated-Services-Architektur steht ein Modell zur Verfügung, welches erlaubt, Daten für kritische Anwendungen über das Internet Protocol mit verschiedenen Dienstgüten zu übertragen. Die Router innerhalb einer Differentiated-Services-Domäne lassen sich in drei Kategorien aufteilen. Spezielle Komponenten innerhalb der Router sorgen für eine Sicherung der Dienstgüte. Die Verwaltung einer Differentiated-Services-Domäne ist komplex und stellt zahlreiche Anforderungen. Zur Bewältigung dieser Aufgaben wurden drei verschiedene Ansätze vorgestellt.

## Literatur


[BlWe99] Roland Bless und Klaus Wehrle. Managment-Architektur zur Unterstützung von Gruppenkommunikation in Differentiated-Services-Netzen. In *1. GI-Workshop, Multicast – Protokolle und Anwendungen*, Braunschweig, Mai 1999. S. 15–27.

[BYFB+99a] Y. Bernet, R. Yavatkar, P. Ford, F. Baker, L. Zhang, K. Nichols und M. Speer. A Framework for Use of RSVP with Diff-Serv Networks. IETF Internet-Draft draft-ietf-diffserv-framework-02.txt, Februar 1999.

[BYFB+99b] Y. Bernet, R. Yavatkar, P. Ford, F. Baker, L. Zhang, K. Nichols, M. Speer und R. Braden. Interoperation of RSVP/Int-Serv and Diff-Serv Networks. IETF Internet Draft draft-ietf-diffserv-rsvp-02.txt, Februar 1999.

[NiJZ99] K. Nichols, V. Jacobson und L. Zhang. A Two-bit Differentiated Services Architecture for the Internet. RFC 2638, Juli 1999.

[ROTZ+98] F. Reichmeyer, L. Ong, A. Terzis, L. Zhang und R. Yavatkar. A Two-Tier Resource Management Model for Differentiated Services Networks. Internet-Draft, November 1998.

[Wehr99] Klaus Wehrle. Implementierung und Evaluierung neuartiger Dienste für das Internet der nächsten Generation. Diplomarbeit, Universität Karlsruhe (TH), Institut für Telematik, Januar 1999.

# Optische Netzwerke

Partho Paul

**Kurzfassung**

In der vorliegenden Ausarbeitung werden die Vorteile von Licht als Übertragungsmedium dargestellt, sowie anhand einer schematischen Darstellung eines optischen Zugangsnetzes bisherige optische Netze, wie aktive und passive optische Netze bzw. Hybridnetze, vorgestellt und Konzepte rein optischer Netzwerke betrachtet. Die WDM-Technik, als neue Übertragungstechnik für Glasfasernetze, wird hier in Verbindung mit rein optischen Netzwerken besprochen. Sie erfordert neue Netzkomponenten, wie Add/Drop-Multiplexer, optische Sternkoppler, optische Cross-Connectoren und Konverter, für die Signalverarbeitung. Diese werden im Zusammenhang mit einem vereinfachten Schichtenmodell eines Kommunikationsnetzes näher erläutert. Desweiteren wird die Übertragungssicherheit rein optischer Netze diskutiert und Verfahren wie Protection und Restoration vorgestellt.

## 1 Einleitung

Ein Problem heutiger Kommunikationsnetze ist die immer zu geringe Übertragungsrate. Existierende optoelektronische Netzwerke erreichen höhere Bandbreiten als normales Kupferkabel, sind aber im Vergleich zu photonischen Netzen um ein Vielfaches langsamer. Internet, World Wide Web, Online-Dienste, Video on Demand, Telearbeit, Multimedia, IP-Telefonie erfordern heutzutage immer mehr Bandbreite. Beim wachsenden Zugang der Benutzer zum Netz droht die Übertragungskapazität zu einem gravierenden Engpaß in den weltweiten Kommunikationnetzen zu werden. Heutige Punkt-zu-Punkt-Übertragungsstrecken sind nach wie vor noch weitgehend herkömmliche optoelektronische Netze, d.h. die Übertragung erfolgt optisch, die Vermittlung zwischen Netzen und Knoten elektrisch. Daher versucht man, neben der optischen Übertragung auch die Vermittlung rein optisch zu gestalten.

Seit 1995 wird die WDM-Technologie (Wave Division Multiplexing, siehe Abschnitt 4) eingesetzt, die Übertragungsraten bis zu 100 GBit/s über eine Glasfaser erlaubt [Spät99]. Übliche Übertragungsraten heutiger Glasfasernetze betragen 2,5 GBit/s pro Faser. In Forschungslabors wurden bereits Raten bis in den Bereich von einigen TBit/s erzielt. Ermöglicht haben diese Entwicklung unter anderem auch die Fortschritte bei der Verstärkertechnik. Insbesondere die Faserverstärker (EDFAs, Erbium Doped Fiber Amplifiers), die die gleichzeitige Verstärkung aller auf einer Faser transportierten Kanäle ermöglichen, trugen wesentlich zum Aufschwung der WDM-Technik bei.

## 2 Lichtwellenleiter und Glasfaser

Lichtwellenleiter (LWL) aus Glas haben zwei große Vorteile gegenüber Koaxialkabel aus Kupfer: Sie haben erstens eine wesentlich höhere Übertragungskapazität und eine kleinere Dämpfung, die außerdem nicht von der zu übertragenden Modulationsfrequenz abhängt. Bei gleicher Übertragungskapazität ist das LWL-Kabel zweitens wesentlich kleiner und leichter als ein Kupferkabel. Desweiteren ist die Bitfehlerrate kleiner als $10^{-11}$ [SeHL98].

Für die Datenübertragung über Lichtwellenleiter stehen nur bestimmte Wellenlängen zur Verfügung. Im Bereich von 1300 nm steht eine Bandbreite von ungefähr 200 nm zur Verfügung, in der die Dämpfung weniger als 0,5 dB/km beträgt. Ein weiteres Dämpfungsminimum liegt bei 1550 nm. Zur Verdeutlichung siehe Abbildung 1 [Mukh97]. In den äußeren Bereichen steigt die Dämpfung exponentiell an.

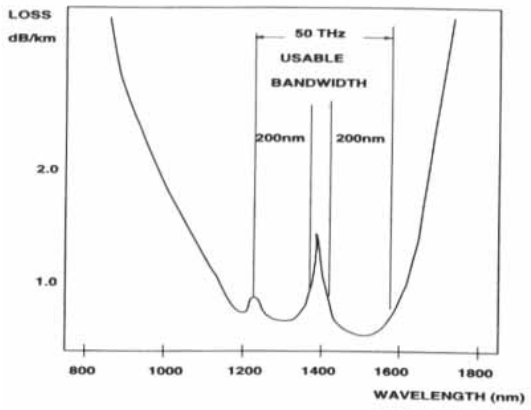


Abbildung 1: Bereiche von Dämpfungsminima.

# 3 Aufbau optischer Netze

In Abbildung 2 ist schematisch die Struktur eines optischen Zugangsnetzes dargestellt. Die Hauptkomponenten sind eine Kopfstation als netzseitiger Glasfaserabschluß (OLT, optical line termination) und eine Anzahl von Glasfaserabschlüssen (ONU, optical network unit) auf der Teilnehmerseite, die über ein optisches Verzweigungsnetz (ODN, optical distribution network) miteinander verbunden sind. Es ist üblich, die Schnittstellen des ODN mit S (send) und R (receive) zu bezeichnen. Die vom OLT kommenden optischen Signale werden in der ONU in elektrische gewandelt, aufbereitet und an die elektrischen Teilnehmerschnittstellen übergeben. Andererseits werden die von den Teilnehmerschnittstellen kommenden elektrischen Signale in der ONU gebündelt und in für die optische Übertragung geeignete Signale gewandelt. Desweiteren werden neben den aktiven Netzabschlüssen und den Glasfaserstrecken für das optische Verzweigungsnetz noch optische Koppel- und Selektionselemente gebraucht, wie Router oder Cross-Connectoren. Diese werden im Abschnitt 5 noch eingehender erläutert.

In heutigen optischen Verzweigungsnetzen werden noch optische Signale in elektrische gewandelt und danach wieder zurück in optische. Dies ist enorm kostenaufwendig, da man einen hohen Aufwand für die elektrischen Komponenten betreiben muß. Ziel zukünftiger Netze wird es sein, die Vermittlungskomponenten rein optisch zu gestalten.

## 3.1 Bisherige optische Netze

Bei den heutigen optischen Netzen handelt es sich entweder um Liniennetze für einfache Punkt-zu-Punkt Kommunikation oder um Stern- sowie Busnetze für Punkt-zu-Multipunkt- (P-MP) bzw. Multipunkt-zu-Punkt- (MP-P) Kommunikation. Bei einem Liniennetz ist jeder Teilnehmer über eine eigene Glasfaser mit dem Netzknoten verbunden. Es existieren also keine Verzweigungselemente. Bei Bus- oder Sternnetzen sind die Teilnehmer über optische Koppelelemente mit der gemeinsam genutzten Glasfaser verbunden.

Im folgenden soll kurz auf die Sternnetze eingegangen werden.

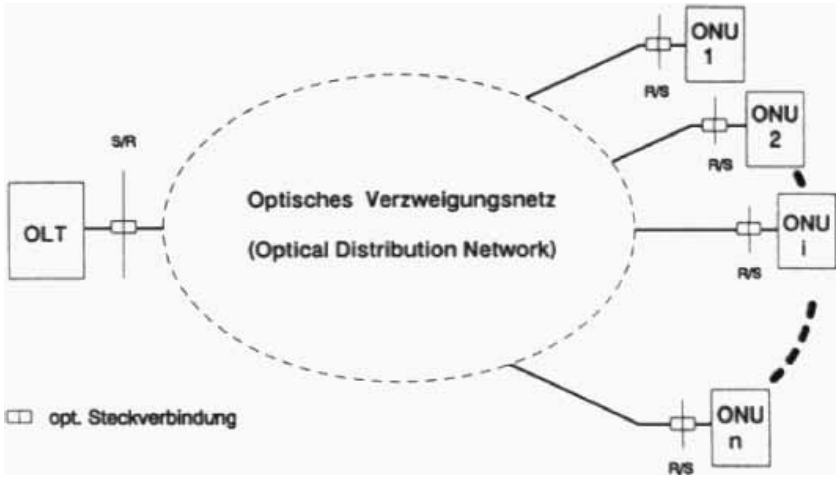


Abbildung 2: Schematische Darstellung optischer Zugangsnetze

- *Aktives optisches Netz*: Wie oben schon erwähnt werden bei herkömmlichen optoelektronischen Netzen Punkt-zu-Punkt-Übertragungstechniken verwendet, d.h. die Übertragung erfolgt optisch über Glasfaser, die Vermittlung zwischen Netzen und Knoten elektrisch. Aktive optische Netze (AON) bestehen aus vielen Punkt-zu-Punkt-Verbindungen. Ankommende optische Signale werden am Eingang aktiver optischer Verteiler (optoelektrische Cross-Connectoren) zunächst in elektrische umgewandelt. Am Ausgang werden sie in optische Signale zurückgewandelt und an die ONUs weitergeleitet. Ein wichtiger Punkt ist, daß bei AONs die Vertraulichkeit der übermittelten Nachrichten besser geschützt wird, weil der aktive Sternpunkt jeder ONU in Abwärtsrichtung nur den sie betreffenden Informationskanal zuweist.

  Diese elektrische Vermittlungsfunktion wird auch oft als *'electric bottleneck'* bezeichnet, da elektrische Signale mit einer viel geringeren Bitrate übertragen werden können als optische. Für die Vermittlung wird auch eine Stromversorgung notwendig, und wegen der elektrischen bzw. optoelektrischen Wandlung vor und hinter dem elektrischen Cross-Connect ist die doppelte Anzahl von teuren aktiven Sende- und Empfangskomponenten erforderlich. Da aber die Übertragung der Signale gezielt an die ONUs erfolgt, sind bei aktiven optischen Netzen die Anforderungen an Sender und Empfänger grundsätzlich geringer als bei passiven optischen Netzen.

- *Passives optisches Netz*: Im Gegensatz zum AON werden beim passiven optischen Netz (PON) keine aktiven signalverarbeitenden Komponenten eingesetzt. Daher ist für das Netz selbst keine elektrische Stromversorgung notwendig. Für die Vermittlung stehen passive Koppel- und Verzweigungselemente zur Verfügung. Gewöhnlich werden Fasernetze vom Typ Doppel- oder Mehrfachstern (Abbildung 3) bevorzugt. Bei der PON-Architektur werden grundsätzlich alle vom OLT ausgesendeten Signale von jedem Teilnehmer empfangen, d.h. die Signale werden an den Verzweigungen aufgeteilt und dann an jede der einzelnen Fasern weitergeleitet. Daher müssen besondere Maßnahmen getroffen werden, damit die Information nur vom jeweiligen Empfänger gelesen werden kann.

  Die maximal mögliche Anzahl der ONUs, die ein OLT über das ODN versorgen kann, hängt von der verfügbaren optischen Sendeleistung ab. Die durch die Aufteilung abgeschwächten Signale können zwar mit Hilfe von optischen Verstärkern verbessert werden, diese sind aber nur auf einer begrenzten Verstärkungsbandbreite von einigen 10 nm einsetzbar. Die optische Transparenz wird also eingeschränkt. Dies führt grundsätzlich bei PONs zum einem Flaschenhals.

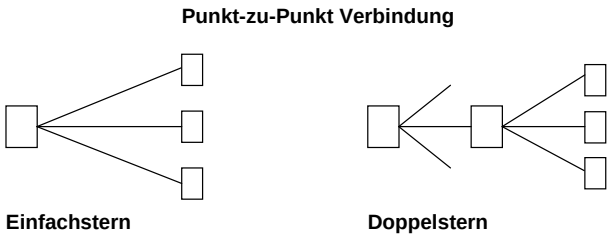


Abbildung 3: Einfach- und Doppelstern

In der Tabelle 1 wird das passive optische Netz mit dem aktiven verglichen.

- *Hybridnetze*: Aus technischen und wirtschaftlichen Gründen kann es sinnvoll sein, für den letzten Netzabschnitt zum Teilnehmer bereits vorhandene Übertragungsnetze zu verwenden oder auch Funknetze neu einzurichten. Im ersten Fall sind dies die vorhandenen Breitbandübertragungsnetze auf Koaxialleitungen. Diese müssen noch für die bidirektionale Kommunikation erweitert werden. Funknetze in Verbindung mit optischen Systemen sind in Gebieten mit geringerem Breitbandbedarf oder dünner Besiedlung geeignet.

| **Merkmale** | **PON**(Stern) | **AON**(Stern) |
|---|---|---|
| Netzstruktur | Punkt-zu-Sternpunkt | Punkt-zu-Punkt |
| Anzahl opt. Sender u. Empfänger | n+1 | 2(n+1) |
| Verteiler im Netz (VtN) | pass. Sternkoppler | elektr. Cross-Connect |
| Stromversorgung im VtN | nein | ja |
| opt. Transparenz vorhanden | ja | nein |
| Abhörsicherheit gewährleistet | durch zusätzliche Mittel (Protokoll, Codierung) | ja |
| Fehlertoleranz | aufwendig | einfach |
| Erweiterbarkeit | ja | nein, vollständiger Systemwechsel oder opt. Bypass notwendig |

Tabelle 1: Vergleich der Eigenschaften von PON und AON

### 3.2 Zukünftige rein optische Netzwerke

Punkt-zu-Punkt-Verbindungen bilden noch keine Netzwerke, sondern allenfalls ein Netz basierend auf optischer Übertragungstechnik. Bei den meisten Kommunikationsnetzen gilt, daß der größte Teil des bei einem Vermittlungsknoten anfallenden Verkehrs aus sogenanntem Transitverkehr besteht. Er ist also gar nicht für den betreffenden Knoten bestimmt, sondern muß nur durchgeleitet werden. Zukünftig soll für die optischen Netzwerke die WDM-Technik eingesetzt werden und die Vermittlungskomponenten werden rein optisch gestaltet. Im folgenden Abschnitt wird auf die WDM-Technik näher eingegangen.

## 4 Wavelength Division Multiplexing

Die Entwicklung der WDM-Technik reicht bis in die 70er Jahre zurück, wurde aber erst seit 1995 in Netzen eingesetzt [Spät99]. WDM ist zuallererst ein spezielles Multiplexverfahren,

das die gleichzeitige oder zeitlich geschachtelte Übertragung von Signalen mehrerer Nachrichten über eine Glasfaser erlaubt. Man kann bei Multiplexverfahren unterscheiden zwischen Raummultiplex (SDM, Space Division Multiplex, die Signale werden in räumlich getrennten Kanälen transportiert), Zeitmultiplex (TDM, Time Division Multiplex, die Signale unterschiedlicher Nachrichten werden zeitlich verschachtelt) und Frequenzmultiplex (FDM, Frequency Division Multiplex, jedem Signal steht ein eigener Frequenzbereich zur Verfügung).

Physikalisch gesehen handelt es sich bei WDM um eine spezielle Variante des Frequenzmultiplex (siehe Abbildung 4). Dabei werden die Signale über mehrere unterschiedliche Wellenlängen über eine Glasfaser übertragen. Dies ist deshalb möglich, da sich die einzelnen Wellenlängen nicht (oder zumindest kaum) gegenseitig beeinflussen. Bei Punkt-zu-Punkt-Verbindungen ist das natürlich kein Problem, da leicht garantiert werden kann, daß nur unterschiedliche Frequenzen gemultiplext werden. Schwierigkeiten können aber bei flexiblen optischen Vermittlungsknoten entstehen. Prinzipiell kann der Fall auftreten, daß unterschiedliche Wellenlängen aus verschiedenen Eingängen bei einem Vermittlungsknoten auf den selben Ausgang geleitet werden. Dieser Wellenlängenkonflikt muß durch spezielle optische Konverter gelöst werden. Die optischen Signale werden auf eine andere Wellenlänge umgesetzt, ohne dabei die Signale selbst auszuwerten.

Die Wavelength-Division-Multiplexing-Technik bietet eine ganze Reihe von Vorteilen. Bei Kapazitätsengpässen im Netz können nachträglich zusätzliche Wellenlängenkanäle hinzugefügt werden, ohne das ganze System austauschen zu müssen. Dies wiederum ermöglicht es, die Anfangsinvestitionen niedrig zu halten. Spätere Verlegung von Kabelsystemen, die die Hauptkosten verursachen, würden entfallen. Ein weiterer Vorteil für den Durchbruch der WDM-Systeme ist die Mehrkanalverstärkung, die ein ganzes Bündel von Wellenlängenkanälen gleichzeitig verstärken kann. Im Gegensatz zu SDM-Systemen, bei dem jeder Kanal separat einen eigenen Verstärker benötigt, können so enorme Kosten bei WDM-Systemen eingespart werden.

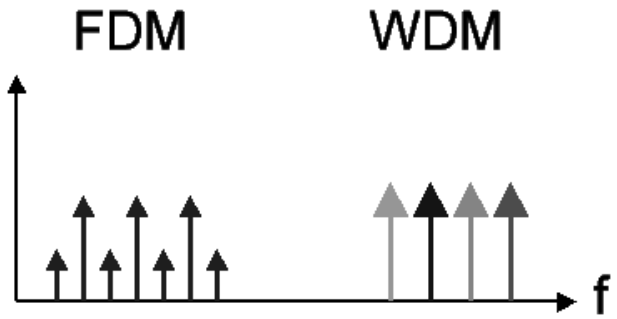


Abbildung 4: Frequenz- und Wavelength Division Multiplex

### 4.1 Einsatz der WDM-Technik in Netzwerken

Die theoretisch mögliche Übertragungskapazität von Glasfaser steht in einem krassen Gegensatz zu den bisher technisch genutzten Datenraten. Dennoch hat die Glasfaser im Verhältnis zu Kupferkabel große Bandbreiten. Betrachten wir im folgenden ein vereinfachtes Schichtenmodell eines Kommunikationsnetzwerkes (siehe Abbildung 5), das aus Verbindungsschicht, Pfadschicht und Übertragungsschicht besteht.

In der Verbindungsschicht werden mit Hilfe von Signalisierung die gewünschten Ende-zu-Ende-Verbindungen von einem Teilnehmer zu einem anderen für bestimmte Zeitabschnitte auf- und abgebaut. Die Pfadschicht stellt dieser Schicht ein Transportnetz zur Verfügung, das

durch ein sogenanntes Managementsystem konfiguriert wird. Der unmittelbare Datentransport, derzeit die einzige Aufgabe der optischen Übertragungstechnik, erfolgt letztendlich auf der physikalischen Ebene in der Übertragungsschicht. Da unvermeidliche Wandlungen von optischen in elektrische und von elektrischen in optische Signale kostenaufwendig sind, stellt sich die Frage, ob man die rein optische Signalverarbeitung in den anderen Schichten umsetzen kann. In den kommenden Unterabschnitten soll die Verwendung der WDM-Technik in der Verbindungsschicht und der Pfadschicht gezeigt werden.

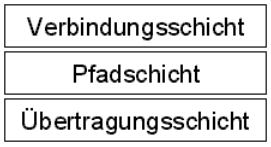


Abbildung 5: Vereinfachtes Schichtenmodell eines Kommunikationsnetzwerkes

### 4.1.1 Wellenlängenmultiplex in der Verbindungsschicht

In dieser Schicht werden die Sendesignale aller angeschlossenen Teilnehmer, denen jeweils unterschiedliche Wellenlängen zugeordnet sind, mit einem optischen Sternkoppler (siehe Abbildung 7) zusammengefaßt. Das Summenspektrum wird dann jedem einzelnen Teilnehmer zurückgesendet, um diesem einen Zugriff auf alle Kanäle zu ermöglichen. Die Vermittlungsfunktion wird dadurch realisiert, daß jede Station mit einem hinsichtlich der Wellenlänge selektiven Empfänger ausgestattet ist und daß entweder die Empfänger oder die Sender oder eventuell auch beide abgestimmt werden können. Will man nun eventuell zwei Netze miteinander koppeln, so muß man darauf achten, daß keine zusätzliche Infrastruktur notwendig wird, d.h. daß das existierende Verbindungsleitungsnetz genutzt werden kann.

### 4.1.2 Wellenlängenmultiplex in der Pfadschicht

In dieser Schicht sollen nun Wege gefunden werden, um die Daten von einem Teilnehmer zu einem anderen zu verschicken. Für die signalverarbeitenden Prozesse sind neuartige Komponenten notwendig, die die Signale an die Teilnehmer weiterleiten können. In der Pfadschicht werden im wesentlichen drei optische Netzwerkelemente benötigt: Optische Multiplexer und Demultiplexer (OMUX/ODEMUX), optische Add/Drop-Multiplexer (OADM) und optische Cross-Connectoren (OCC).

# 5 Optische Komponenten

Rein optische Vermittlungskomponenten haben den Vorteil, daß keine zusätzliche Stromversorgung notwendig ist, das wiederum senkt die Kosten eines Netzes. Vor allem erlauben sie, wesentlich höhere Datenraten weiterzuleiten als optoelektrische Komponenten, bei denen die Wandlung der optischen Signale erstens ein großer Aufwand bei zunehmender Bitrate ist und bei denen zweitens die Verluste höher sind. Optoelektrische Verteiler sind zur Zeit aber die am weitesten entwickelten.

## 5.1 Wellenlängenmultiplexer und -demultiplexer

Aufgabe eines Wellenlängenmultiplexers oder optischen Multiplexers (OMUX) ist es, viele optische Kanäle von unterschiedlichen Quellen auf eine Glasfaser zu bündeln. Mit dem optischen Demultiplexer (ODEMUX) werden die verschiedenen Wellenlängen wieder getrennt.

Diese Komponenten sind Basisbaugruppen für optische Add/Drop-Multiplexer und optische Cross-Connectoren, sie sind aber auch bei Punkt-zu-Punkt-Verbindungen mit WDM-Betrieb unverzichtbar.

## 5.2 Optische Add/Drop-Multiplexer

Ein optischer Add/Drop-Multiplexer (siehe Abbildung 6) trennt aus einer Gruppe von Wellenlängen eine oder auch mehrere Wellenlängen heraus und leitet diese auf einen anderen Ausgang weiter, wobei gleichzeitig Kanäle der entsprechenden Wellenlängen mit neuem Dateninhalt von einem zweiten Eingang in die Gruppe eingefügt werden.

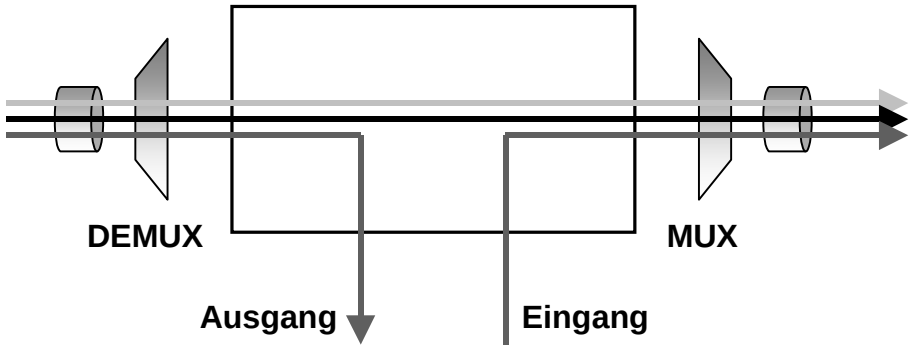


Abbildung 6: Add-Drop-Multiplexer

## 5.3 Sternkoppler

Die Funktion des optischen Sternkopplers (Abbildung 7) besteht darin, ein Eingangssignal von einer Glasfaser aufzuteilen und an alle Ausgangsfasern weiterzuleiten. Eine Abschwächung der aufgeteilten Signale muß eventuell durch optische Verstärker aufbereitet werden. Ein Problem kann entstehen, wenn zum Beispiel zwei gleiche Wellenlängen aus unterschiedlichen Eingangskanälen auf den selben Ausgang geroutet werden, dann müssen Konverter (siehe Abschnitt 5.5) den Wellenlängenkonflikt beheben.

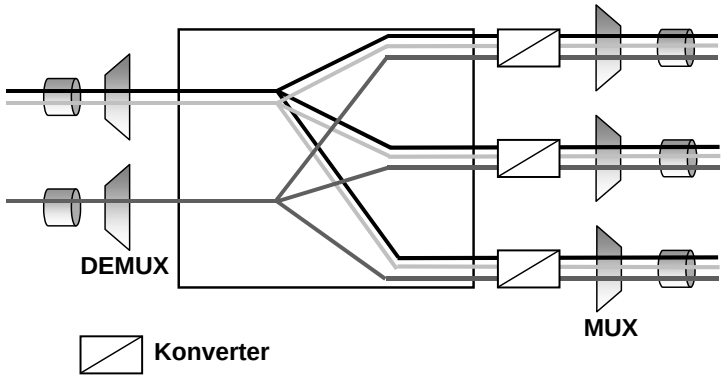


Abbildung 7: Optische Sternkoppler

## 5.4 Optische Cross-Connectoren

Die Funktion optischer Cross-Connectoren (siehe Abbildung 8) besteht darin, Wellenlängenkanäle bezüglich ihrer Lage im Raum (Faser) und ihrer Wellenlänge (Frequenz) umzuschalten,

d.h. Cross-Connectoren können Eingangssignale auf beliebige Ausgänge weiterleiten. Bei Wellenlängenkonflikten werden auch hier Konverter eingesetzt.

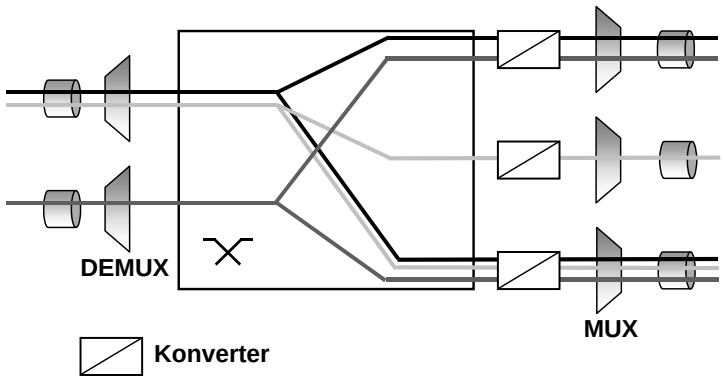


Abbildung 8: Optischer Cross-Connector

### 5.5 Konverter

Allgemein werden Konverter hier für die Umwandlung einer Wellenlänge in eine andere verwendet, um damit Wellenlängenkonflikte, die bei Cross-Connectoren oder Sternkopplern entstehen können, zu umgehen. Prinzipiell können Konverter alle Kanäle in Anspruch nehmen. Das Netzmanagement wird durch den Einsatz von Konvertern vereinfacht. Inbesondere das Routen der Signale wird einfacher, da keine durchgehenden Wellenlängenpfade gefunden werden müssen. Von Vermittlungsknoten zu Vermittlungsknoten können so jeweils unterschiedliche Wellenlängenkanäle benutzt werden. Ein weiterer Punkt ist, daß viele Konverter nicht nur die Signale auf eine andere Wellenlänge übertragen, sondern das Signal gleichzeitig noch erneuern.

Für die Realisierung von Konvertern gibt es verschiedene Möglichkeiten:

- Opto-elektronische Konverter wandeln Signale an einer Empfängerdiode in elektrische Signale um und senden sie mit einem Laser auf die Ausgangswellenlänge. Technisch gesehen ist es bisher die ausgereifteste Methode, nur die beschränkte Integrationsmöglichkeit fordert nach rein optischen Konvertern.
- Laser-Konverter sind rein optische Konverter, bei denen die optischen Eingangssignale direkt den Sendelaser steuern. Noch sind sie nicht ganz ausgereift, zu viele Störeinflüsse bei der Konversion begrenzen die Einsatzmöglichkeiten.
- Die Konversion mittels optisch gesteuerter Tore legt eine kontinuierliche Wellenlänge auf der gewünschten Ausgangsfrequenz gemeinsam mit dem Eingangssignal an ein optisch steuerbares Tor an. Das Eingangssignal dient dabei zur Ansteuerung für das Tor und läßt die Ausgangswellenlänge im gewünschten Takt passieren. Diese Konversionsart ist momentan der meistversprechende Ansatz für rein optische Konversion.

## 6 Übertragungssicherheit

Die Anforderungen an die Verfügbarkeit von Kommunikationsnetzen steigen. Um die Verfügbarkeit von Netzen auch dann gewährleisten zu können, wenn einzelne Fehler (etwa Ausfall einer Vermittlungskomponente oder Kabelunterbrechung) im Netz auftreten, sind entsprechende Schutzmechanismen erforderlich. Deren Bedeutung nimmt in optischen Netzen noch

zu, da man hier riesige Bitraten über einzelne Glasfasern und Netzkomponenten transportieren will.

Man unterscheidet grundsätzlich zwei Arten von Schutzmechanismen. Sie beruhen beide darauf, daß zusätzlich zu den Arbeitskapazitäten Reservekapazitäten im Netz vorhanden sind.

### 6.1 Protection-Verfahren

Bei Protection-Verfahren werden bereits bei der Netzkonfiguration, d.h. vor dem Eintreten eines Fehlers, für die Verkehrsströme im Netz zugehörige Reservekapazitäten definiert, auf die man dann im Fehlerfall umschalten kann. Folgende Mechanismen werden vor allem bei den Protection-Verfahren verwendet:

- Link-Protection sichert einzelne Übertragungsstrecken zwischen benachbarten Netzknoten ab, während Path-Protection Ende-zu-Ende-Pfade (die sich über mehrere Zwischenknoten erstrecken) schützt.
- 1+1-Protection richtet zu jedem Arbeitspfad einen zugehörigen Reservepfad ein und überträgt die Daten über beide Pfade. Der Empfänger wählt dann die fehlerfreien Daten aus.
- 1:1-Protection definiert wie das 1+1-Verfahren zu jedem Arbeitspfad einen Reservepfad, der aber nicht eine Kopie der Daten trägt, sondern im fehlerfreien Fall beispielsweise niederpriorisiertem Verkehr zur Verfügung steht. Im Fehlerfall schalten dann die Sender und Empfänger auf die Reservestrecke um.
- m:n-Protection stellt für n Arbeitspfade m Reservepfade zur Verfügung. Für geringe Fehlerwahrscheinlichkeit lassen sich so wesentlich Ressourcen einsparen.

### 6.2 Restoration-Verfahren

Das Restoration-Verfahren sucht erst beim Auftreten eines Fehlers nach momentan freien Ressourcen im Netz und versucht dann, den betroffenen Verkehr auf diese umzuleiten.

Protection-Verfahren arbeiten schneller als Restoration-Verfahren, da sie nicht erst nach freien Kapazitäten suchen müssen. Andererseits lassen sich Restoration-Verfahren effizienter realisieren, da sie auch freie Arbeitskapazitäten ausnutzen können. Daher kann es für einen Netzbetreiber vorteilhaft sein, nur hochpriorisierten Verkehr mit Protection-Mechanismen abzusichern, während für Verkehr niedrigerer Priorität Restoration-Verfahren ausreichen.

## 7 Probleme optischer Netze

Die optische Signalübertragung kann durch eine Vielzahl von Effekten störend beeinflußt werden. Hierzu gehören u.a. Dispersion und Polarisation, die z.B. begrenzend auf die optische Sendeleistung einwirkt. Reflexionen und unzureichende Isolationen verursachen weitere Probleme bei der Datenübertragung. Im folgenden werden einige Störeinflüsse beschrieben.

- Störungen durch Reflexion: Neben den Kostenvorteilen bei Einfaserbetrieb eines Netzes, in dem also die Datenübertragung in Hin- und Rückrichtung über eine Leitung erfolgt, hat man das Problem mit der Reflexion. Durch die nicht ideale Isolation zwischen Sende- und Empfangskanal kann es vorkommen, daß ein Anteil des gesendeten Nutzsignals reflektiert wird. Diese Signalanteile überlagern sich mit dem aus der Gegenrichtung kommenden Nutzsignal und können die Empfangsqualität störend beeinflussen.

- Nebensprechen durch Rayleigh-Streuung: Die Rayleigh-Streuung ist bei jeder Glasfaserübertragung vorhanden und kann auch durch keine technischen Maßnahmen unterdrückt werden. Das eingekoppelte Licht wird hierbei diffus gestreut, wobei bei bidirektionaler Übertragung auf einer Faser der Anteil des Streulichts, der zum Sender zurückgestreut wird, die Störungen verursacht, da er sich mit dem von der Gegenstation kommenden Signal überlagert. Dieser Effekt ist wellenlängenabhängig und baut sich mit zunehmender Faserlänge bis zu einem Maximalwert auf.

- Nebensprechen durch Fresnelreflexion (Fernnebensprechen): Während bereits bei der Herstellung der Glasfaser die Rayleigh-Streuung sozusagen 'eingebaut' wird, lassen sich feste Reflexionen an Übergängen grundsätzlich durch hinreichende Spezifikation der Komponenten und saubere Aufbautechnik beherrschen.

- Nebensprechen durch unzureichende Isolation: An die Komponenten zur Richtungstrennung, d.h. zur Weiterleitung des Signals an einer Vermittlungsstelle, sind besonders hohe Anforderungen hinsichtlich einer hinreichenden Isolation zu stellen, damit die Signale richtig weitergeleitet werden können. Aber auch allgemein ist eine gute Isolation der Glasfaser notwendig, um sozusagen den Austritt des Lichts aus der Leitung zu verhindern, was Schwächung oder auch Streuung des Signals zur Folge hätte.

- Nebensprechen durch nichtlineare Effekte: Bei optischer Datenübertragung über mehrere Wellenlängen kann es passieren, daß durch Wellenmischung zusätzlich optische Träger erzeugt werden, die unerwünscht in die Übertragungskanäle fallen.

Bei den hohen Bitraten wirkt sich ein Fehler, z.B. ein Faserbruch, in den höheren Schichten gravierend aus. Die Fehlerbehandlung führt zu einem großen technischen Problem. Im Falle eines Fehlers müssen die Signale neu durch das Netz geleitet werden, das erfordert einen extrem hohen Aufwand beim Netzwerkmanagement und führt zu einer höheren Netzbelastung. Da in den Knoten die Verluste der Übertragungsstrecke und die der passiven Komponenten mit optischer Verstärkung ausgeglichen werden müssen, ist wegen der Rauschakkumulation die Anzahl der Knoten begrenzt, die ein optisches Signal durchlaufen kann. Das bedeutet, daß die Signale über eine längere Strecke mehr und mehr Störungen unterliegen und es zunehmend schwieriger wird, das ursprüngliche Signal wieder herzustellen. Natürlich wird auch die Strecke, die ein Signal ohne Zwischenverstärker durchlaufen kann, durch die Sendeleistung der Laser begrenzt. Heutige kommerzielle Systeme mit Cross-Connectoren und Sternkopplern schaffen Signalübertragungen von 100 km ohne Zwischenverstärker [Hult96].

# 8 Aktuelle Projekte

Ein ehrgeiziges Projekt, das europaweit läuft [Reda99], ist die Verbindung größerer europäischer Städte, wie London, München, Frankfurt, Zürich, Mailand, Madrid, Paris und weiterer Wirtschaftsmetropolen mit einem superschnellen Glasfasernetz. Bis Ende des Jahres 2000 sollen 90 Prozent des Netzes stehen, durch welches 70 Städte in 17 Ländern miteinander verbunden werden. Die Kosten für die erste Ausbaustufe belaufen sich auf 2,9 Milliarden DM. Die Übertragungskapazität soll bis zu 300 Terabit pro Sekunde betragen. Insgesamt werden acht Millionen Kilometer Glasfaserleitung vergraben. Die Kabelbäume enthalten 192 Leitungspaare, dies würde bedeuten, daß ca. 700 Gigabit pro Sekunde über ein Kabel übertragen werden können. Auch wenn das Netz nicht rein optisch gestaltet wird, so werden auf Teilstrecken nur rein optische Komponenten verwendet.

# 9 Zusammenfassung

Wellenlängenmultiplex bietet nach heutigem Stand der Technik die beste Möglichkeit der effektiven Nutzung optischer Netze. Durch die Netzwerkelemente OADM, OMUX, ODEMUX und OCC kann eine vergrößerte Flexibilität, verbesserte Funktionssicherheit und vor allem eine größere Gesamtkapazität des Netzes realisiert werden. Engpässe jetziger Netze könnten bald durch ein Überangebot an Übertragungskapazität abgelöst werden, dies könnte aber zur Folge haben, das Bandbreiten verschwendet werden. Noch sind größere rein optische Netzwerke im Forschungsstadium, vielversprechende Konzepte und Teilrealisierungen existieren bereits.

## Literatur


[Hult96] H. Hultzsch. *Optische Telekommunikation.* Damm-Verlag, Gelsenkirchen. 1996.

[Mukh97] Biswanath Mukherjee. *Optical Commnunication Network.* McGraw-Hill, New York. 1997.

[Reda99] Süddeutsche Redaktion. Europa bekomme Super-Autobahn für Daten. *Süddeutsche Zeitung* (148), Juli 1999.

[SeHL98] J. M. Senior, M. R. Handley und M. S. Leeson. Developments in Wavelength Division Multiple Access Networking. *IEEE Communications Magazine* 36(12), Dezember 98, S. 28–36.

[Spät99] Jan Späth. Mehr Licht. *c't*, Januar 1999.

# Das LonTalk-Protokoll für Kontrollnetzwerke

Alexandre Grossoul

**Kurzfassung**

In dieser Arbeit ist der Begriff der Kontrollnetzwerke mit ihren Eigenschaften und Anwendungsbereichen eingeführt. Weiter werden das LonTalk-Protokoll definiert, sein Schichtenmodell im Vergleich zur OSI-Referenzmodell sowie die LonTalk-Adressen beschrieben. Es wird auch auf die einzelnen Schichten des LonTalk-Protokoll-Schichtenmodells eingegangen. Dabei werden Aufgaben der Schichten, evtl. ihre Unterschichten, Schnittstellen zu den benachtbaren Schichten sowie Rahmenformate beschrieben. Zum Schluß wird eine kurze Einführung in das Netzwerkmanagement und die Netzwerkdiagnostik gegeben.

## 1 Einleitung

Das LonTalk-Protokoll [Eche94] definiert die Grundregeln für die Kommunikation in einem LonWorks-System [DiLS98]. LON steht für Local Operating Networks (deutsch: Kontroll- oder Steuernetzwerke) und ist ein Feldbussystem, das sich für Aufbau und Betrieb leistungsfähiger und vor allem weitverzweigter dezentraler Netze eignet, und bei dem eine umfassende Unterstützung der Anwender geboten wird. Typische Merkmale für diese Art von Netzwerken sind:

- kurze Meldungen
- sehr geringe Kosten für einen einzelnen Netzknoten
- eine Vielfalt unterschiedlichster Kommunikationsmedien
- meist geringe Bandbreite auf dem Übertragungskanal
- geringe Wartungskosten sowie
- die Integrationsfähigkeit unterschiedlichster Hersteller in einem gemeinsamen Netzwerk.

Abbildung 1 stellt die wichtigsten Eigenschaften der LonWorks-Technologie dar.

LON-Systeme sind in der Regel verteilt aufgebaut (örtlich verteilte Hardware- und Software-Ressourcen, dynamisch konfigurierbare verteilte Anwendungen, Systemtransparenz und kooperative Autonomie) und können bis zu einigen zehntausend Knoten beinhalten. Die Knoten lassen sich für verschiedene Anwendungen entwickeln und im Betrieb konfigurieren und müssen das LonTalk-Protokoll in vollem Ausmaß unterstützen, da ansonsten Leistungsfähigkeit und die Offenheit des Systems nicht gewährleistet sind.

Das LonTalk-Protokoll wird gerade als Kommunikationssystem für die Gebäude- und Industrieautomatisierung standardisiert. Anwendungsbereiche für LON-basierte Systeme sind die Prozeßautomatisierung, Gebäudeautomation, elektrische Haushaltsgeräte sowie viele andere Produktbereiche mit dezentralen Meß- und Steuerkonzepten.

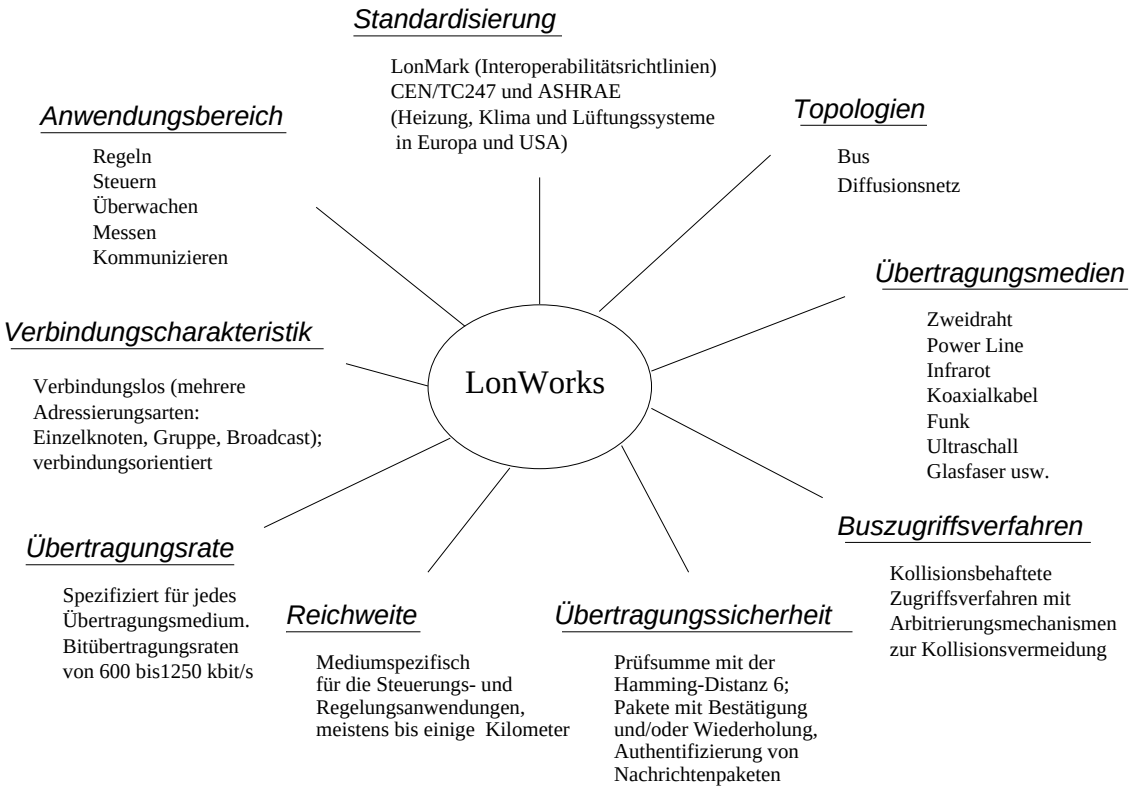


Abbildung 1: Eigenschaften der LonWorks-Netzwerktechnologie

## 2 Protokollübersicht

Bei der Beschreibung der LonTalk-Protokoll-Schichtenmodel wird die OSI-Standard-Terminologie benutzt. Alle sieben Schichten des OSI-Referenzmodells sowie die korrespondierenden Schichten des LonTalk-Protokolls sind in Abbildung 2 dargestellt. Zu jeder Schicht sind die verfügbaren Dienste in Stichworten angeführt. Ein Überblick über die einzelnen Schichten und Dienste erfolgt weiter unten.

## 3 Adressierung im LonTalk-Protokoll

LonTalk-Adressen sind hierarchisch strukturiert. Es gibt drei grundlegenden Adreßformate mit jeweils 3 Komponenten wie in Tabelle 1 dargestellt. Jedes transportierte Datenpaket beinhaltet sowohl Quelladresse als auch Zieladresse, wofür die Adreßformate aus der Tabelle 1 zu Adreßpaaren zusammengesetzt werden. LonTalk-Adressen sind kombinierte Schicht-3- und Schicht-4-Adressen. Es gibt keine Schicht-2-Adresse. Domain und Subnet werden beim Routing verwendet und können somit als Komponenten der Netzwerkadresse bezeichnet werden. Die Neuron-ID ist eine weltweit einzigartige, dem einzelnen Neuron-Chip zugeordnete Identifikationsnummer und wird als Knotenname bezeichnet und als Schicht-4-Adresse aufgefaßt.

Die Adreßinformation im LonTalk-Paketrahmen kann verschieden lang sein. Die Länge der einzelnen Komponenten ist aber, mit Ausnahme der Domain, festgelegt. Die Komponente *Subnet* umfaßt 8 bit und erlaubt 255 Subnetze innerhalb einer Domain (Subnet 0 ist reserviert). Die Komponente *Node* weist 7 bit auf, wobei der Wert 0 nicht verwendet wird. Die Komponente *Group* ist ebenfalls 8 bit breit, und es können somit bis zu 256 Gruppen innerhalb einer Domain gebildet werden. Das Feld *Member* (Gruppenmitglied) ist 6 bit lang, und

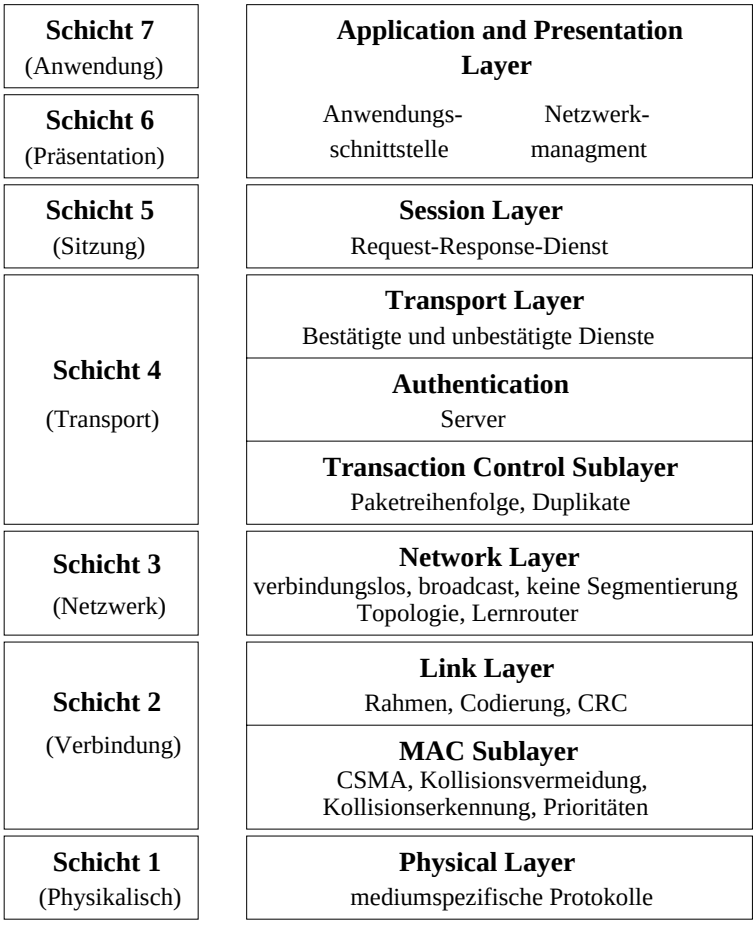


Abbildung 2: OSI Referenzmodell mit korrespondierenden LonTalk-Schichten

| **Adreßformat** |
|---|
| Domain, Subnet, Node |
| Domain, Subnet, Neuron-ID |
| Domain, Group, Member |

Tabelle 1: Drei Basisadreßformate in LonTalk

die *Neuron-ID* hat 48 bit. Die Komponente *Domain* kann je nach Netzwerkstruktur entweder 0, 1, 3 oder 6 byte lang sein. Insgesamt können somit bis zu

(Anzahl Subnets)*(Anzahl Nodes)=$(2^8 - 1)*(2^7 - 1) = 32385$

Knoten innerhalb einer Domain adressiert werden, und prinzipiell sind bis zu $2^{48}$ Domains möglich. Die wesentliche Merkmale der LonTalk-Adreßkomponenten werden in den folgenden Unterkapiteln beschrieben.

## 3.1 Domain

Eine Domain ist ein virtuelles Netzwerk, in dem die gesamte Kommunikation stattfindet. Im Gegensatz zum Internet unterstützt LonTalk keine Inter-Domain-Kommunikation (Kommunikation zwischen 2 oder mehreren Domain); diese muß mittels Gateways auf Applikationsebene abgedeckt werden.

Innerhalb der Domain findet Netzwerkmanagement und Netzwerkadministration statt. Darauf wird im Kapitel 10 eingegangen. Insbesondere werden Groups und Subnets vom lokalen Domain-Administrator vergeben und haben nur im Zusammenhang mit der Domain eine Bedeutung.

### 3.2 Subnet und Node

Das Subnet faßt eine Anzahl (0 - 127) von Netzwerkknoten (Nodes) zu einer Teilmenge der Domain so zusammen, daß innerhalb des Subnets kein Routing stattfindet. Subnets können als logische Kanäle aufgefaßt werden und müssen nicht mit physikalischen Kanälen korrespondieren. Ein oder mehrere Subnets können auf einen physikalischen Kanal oder auf mehrere, die mit Repeater oder Bridge miteinander verbunden sind, abgebildet werden.

Die Adreßkomponente Node kennzeichnet den einzelnen Knoten innerhalb des Subnets. Ein physikalischer Knoten kann bis zu zwei verschiedenen Subnets mit unterschiedlichen Knotennummern angehören unter der Voraussetzung, daß beide Subnets in unterschiedlichen Domains liegen.

### 3.3 Group und Member

Die Komponente Group faßt ähnlich wie das Subnet einen Satz von Netzwerkknoten innerhalb einer Domain zu einer Gruppe zusammen, innerhalb derer mit Hilfe der Komponente Member ein einzelnes Gruppenmitglied identifiziert wird. Group findet Verwendung bei der Gruppenadressierung. Ein einzelner Netzwerkknoten kann maximal 15 verschiedenen Gruppen angehören.

### 3.4 Neuron-ID

Die Neuron-ID wird bei der Produktion des Knotens unveränderlich und weltweit eindeutig festgelegt. Wird die Neuron-ID als Adreßinformation verwendet, so kann sie nur als Zieladresse, jedoch nie als Quelladresse dienen. Es steht dazu keine passende Rahmenstruktur zur Verfügung.

## 4 Physical Layer

Der Physical Layer unterstützt in LonTalk eine Vielzahl von unterschiedlichsten Übertragungsmedien (von kabelgebundenen Medien über Funkverbindungen bis hin zu optischen Übertragungsverfahren), wobei verschiedene Schicht-1-Protokolle in Abhängigkeit von eingesetzten Transceiver verwendet werden. Jedes Medium verlangt eigene Verfahren der Leitungskodierung sowie des Buszugriffes. Zum Beispiel wird Differential-Manchester-Codierung auf Zweidrahtleitungen eingesetzt, Spread Spectrum bei der Übertragung über das 230-V-Netz oder FSK-Modulation bei Funkübertragung.

## 5 Link Layer

Schicht 2 nach OSI gliedert sich bei LonTalk in den MAC (Media Access Control) Sublayer und in den Link Layer. Im MAC Sublayer wird ein kollisionsvermeidendes, jedoch nicht kollisionsfreies Buszugriffsverfahren, das predictive p-persistent CSMA (Carrier Sense Multiple

Access), mit optionaler Priorisierung und Kollisionserkennung bereitgestellt. Die Kollisionserkennung wird zwar vom LonTalk-Protokoll unterstützt, doch von heutigen Transceiver-Entwürfen nicht verwendet, da sie den Datendurchsatz am Netzwerk nur geringfügig verbessert.

Der Link Layer stellt einen subnetzweiten verbindungslosen Datenübertragungsdienst zur Verfügung, wobei seine Funktionalität auf das Erstellen des Paketrahmens sowie auf die Fehlererkennung, jedoch nicht auf Fehlerbehebung beschränkt bleibt. Erkannte Fehler werden an die höhere Ebene gemeldet und dort behoben. Fehlerhafte Rahmen werden am fehlerbehafteten CRC (Cyclic Redundancy Check) erkannt. CRC benutzt dabei das $''x^{16} + x^{12} + x^5 + 1''$-Polynom nach CCITT CRC-16 Standard.

Da die MAC- und CRC-Verfahren allgemein bekannt sind, wird hier darauf nicht weiter eingegangen.

## 5.1 Schnittstelle zu den benachbarten Schichten

Die Schnittstelle zwischen MAC Sublayer und Link Layer sowie zu den angrenzenden Schichten ist in Abbildung 3 dargestellt. Dabei gestaltet sich die Schnittstelle zum Physical Layer mit nur 3 Dienstprimitiven (`P_Data_Indication()`, `P_Data_Request()` und `P_Channel_Active()`) sehr einfach und universell.

Es gibt zwei Betriebsarten der Kommunikation mit dem Physical Layer:

- Direct Mode, dabei wird die Differential-Manchester-Codierung verwendet;
- Special Purpose Mode, dabei wird bidirektional seriell übertragen.

In beiden Betriebsarten wird eine 16-Bit-CRC-Prüfsumme beim Absenden und beim Empfang überprüft.

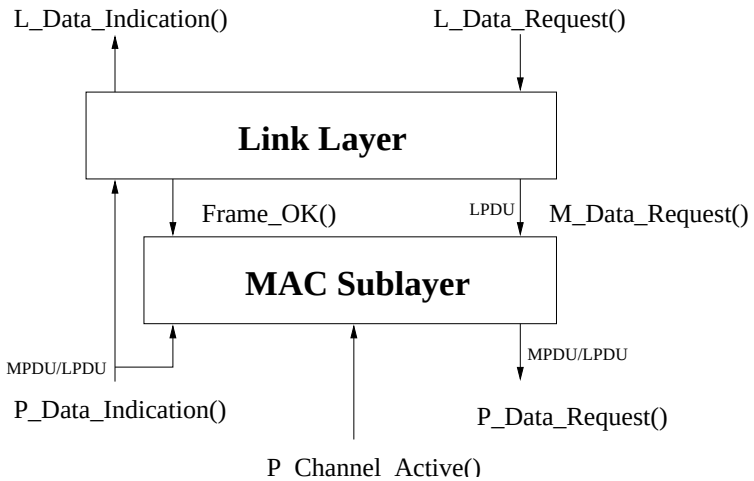


Abbildung 3: Schnittstelle zwischen MAC Sublayer, Link Layer und angrenzenden Schichten

Der Empfang eines Rahmens (MPDU/LPDU – MAC/Link Protocoled Data Unit) wird vollständig vom Link Layer mittels `P_Data_Indication()` durchgeführt. Nur der erfolgreiche Empfang eines Rahmens wird gemeinsam mit der Backlog-Information (siehe Abschnitt 5.2) mittels `Frame_OK()` an den MAC Sublayer gemeldet. Beim Absenden wird der Rahmen (MPDU/LPDU) mit dem `M_Data_Request/P_Data_Request` Service zuerst an den MAC Sublayer und dann an den Physical Layer weitergegeben. Die Übertragung beginnt dabei mit dem höchstwertigen Byte und ebenso mit dem höchstwertigen Bit.

### 5.2 Rahmenformat MPDU/LPDU

Der Rahmenaufbau von MPDU/LPDU ist in Abbildung 4 dargestellt. Die NPDU (Network PDU) wird von Rahmeninformation umschlossen. Zur Bit-Synchronisation wird eine Präambel bestehend aus „1“-Bits ausgesendet. Unmittelbar anschließend erfolgt die Byte-Synchronisation durch Aussenden eines „0“-Bits gefolgt von der Schicht-2-Header-Information mit dem *Priority-Bit*, dem *Alternate Path Bit* und dem *Delta-Backlog*. Mit dem *Priority-Bit* können Kanäle mit Prioritäten ausgestattet werden. Die Anzahl der Prioritätszeitschlitze kann für jeden Kanal separat im Bereich 0 bis 127 gewählt werden, doch müssen alle Knoten am Kanal die gleiche Einstellung treffen. Dem einzelnen Knoten wird nun ein Prioritätszeitschlitz zugeordnet, indem der Knoten seine priorisierten Nachrichten versenden kann. Die Entscheidung, ob eine Nachricht priorisiert oder nicht priorisiert versendet wird, liegt beim Knoten und muß vom Knoten für jede einzelne Nachricht neu entschieden werden. Als *Delta-Backlog* wird das Inkrement bezeichnet, das in jedem Knoten zum *Backlog* addiert wird, wobei man unter dem *Backlog* bei LonTalk den erwarteten Netzwerkverkehr in unmittelbarer Zukunft (predictive) versteht. Mit dem Alternate Path Bit kann bei erfolgloser Kommunikation auf einen alternativen Verbindungsweg umgeschaltet werden. Die Vorstellung der NPDU erfolgt im Abschnitt 6.

Anschließend wird die 16-Bit-CRC-Prüfsumme über das gesamte Paket berechnet. Im Direct Mode wird das Paket mir einer Code Violation (CV) der Differential-Manchester-Codierung beendet, die mindestens 1,25 Bit-Zeiten lang sein muß.

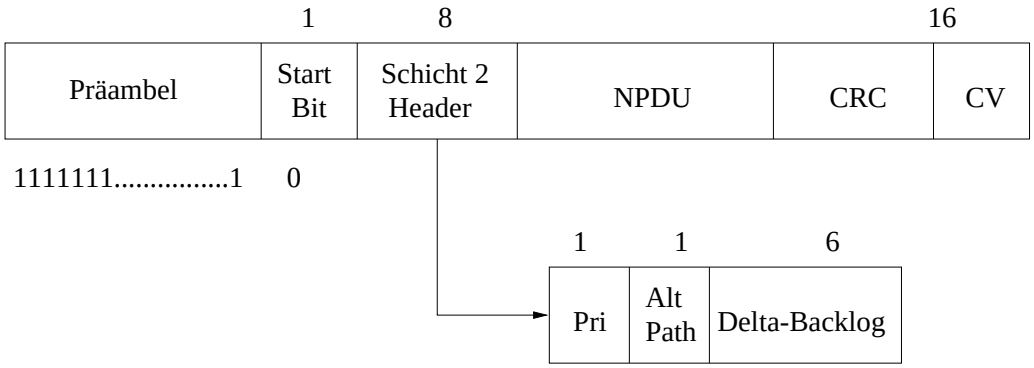


Abbildung 4: Rahmenformat von MPDU/LPDU

## 6 Network Layer

Der Network Layer stellt einen verbindungslosen, unbestätigten Nachrichtendienst für das Versenden von Datenpaketen innerhalb einer Domain zur Verfügung, den man mit folgenden Attributen charakterisieren kann:

- *Unbestätigter Unicast-, Multicast- und Broadcast*-Nachrichtendienst: Abhängig von der Zieladresse wird der LonTalk-Rahmen entweder an einen Knoten (Unicast), an eine Gruppe von Knoten (Multicast), an alle Knoten innerhalb eines Subnets oder an alle Knoten einer Domain (Broadcast) gesendet.
- *Rahmenverlust*: Der Network Layer stellt keine Rahmenwiederholung oder Rahmenbestätigung zur Verfügung. Durchläuft ein Rahmen mehrere Kanäle, so sinkt die Wahrscheinlichkeit der erfolgreichen Rahmenzustellung.

- Erhalt der *Rahmenreihenfolge*: Rückkopplungsfreie Topologien ohne Store & Forward Repeater, die nur auf einen Kanal wirken, garantieren die Beibehaltung der Rahmenreihenfolge.

- *Keine Nachrichtensegmentierung*: Nachrichten können im Network Layer nicht in Segmente aufgestellt und anschließend zu einer Gesamtnachricht zusammengesetzt werden.

## 6.1 Schnittstelle zum Transport Layer

Zwei Dienste werden vom Network Layer zur Verfügung gestellt, `Send_Packet() Request` und `Receive_Packet() Indication`, wie in der Abbildung 5 dargestellt. `Send_Packet()` übergibt der Schicht 3 Quell- und Zieladresse, den Typ des PDUs, die PDU selbst, Informationen zur Priortität, den Delta-Backlog sowie das Alternate Path Bit. In der Gegenrichtung liefert `Receive_Packet() Indication` sowohl Quell- als auch Zieladresse des empfangenen Rahmens, den Typ der PDU, die PDU und das Priority Bit aus Schicht 2.

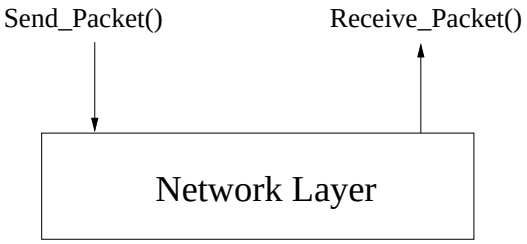


Abbildung 5: Schnittstelle zum Transport Layer

Grundsätzlich kann man die Funktionalität des Network Layer in zwei Teilbereiche gliedern:

- Erkennen und Vergleichen der Zieladresse im empfangenen Paketrahmen mit der eigenen Knotenadresse und Weiterleiten gültiger Datenpakete an den Transport Layer.

- Extrahieren der Routings-Information aus der NPDU und gezieltes Routen anhand von Router-Tabellen.

## 6.2 Rahmenformat der NPDU

Eine NPDU ist wie in Abbildung 6 dargestellt aufgebaut. Zwei Bits sind für die Protokoll-Version reserviert. Bit 4 und 5 kennzeichnen das Format der eingeschlossenen PDU. Die Zuordnung von Bit 4 und 5 ist in Tabelle 2 definiert. Bit 2 und 3 kennzeichnen das verwendete Adreßformat, und Bit 0 und 1 codieren die Länge der Domain-ID.

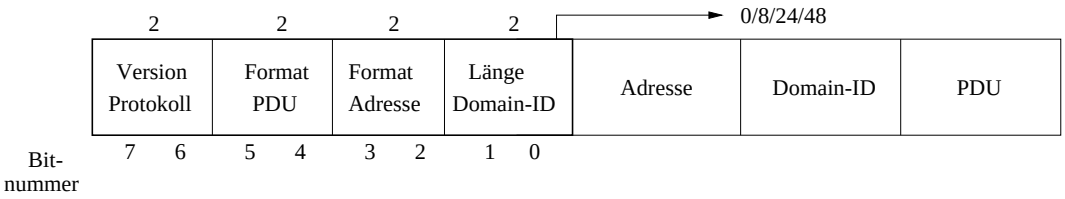


Abbildung 6: Rahmenformat der NPDU

| Bit 5 | Bit 4 | PDU Format |
|---|---|---|
| 0 | 0 | TPDU |
| 0 | 1 | SPDU |
| 1 | 0 | AuthPDU |
| 1 | 1 | APDU |

Tabelle 2: Codierung der PDU-Formate im NPDU-Header

# 7 Transport Layer

Die Schicht 4 ist bei LonTalk wie folgt aufgeteilt:

- Transaction Control Sublayer
- Transport Layer
- Authentication-Server für die Authentication des Absenders

## 7.1 Dienste im Transaction Control Sublayer

Der Transaction Control Sublayer hat folgende Aufgaben:

- Vergabe der Transaktionsnummern beim Absenden einer Nachricht: LonTalk arbeitet mit 4 bit langen Transaktionsnummern, die vom Absender vergeben werden und vom Empfänger zur Erkennung von duplizierten Paketen verwendet werden.
- Limitierung der gleichzeitig abgehenden Transaktionen: Die Anzahl der gleichzeitig aktiven abgehenden Nachrichten ist limitiert. Nur eine priorisierte und eine nichtpriorisierte abgehende Transaktion können gleichzeitig ablaufen. Die Anzahl der gleichzeitig ablaufenden Empfangstransaktionen ist auf maximal 16 beschränkt.
- Verantwortung für die geordnete Reihenfolge der eingehenden Nachrichten beim Empfang
- Erkennung von duplizierten Paketen (Duplikate): Voraussetzung für die erfolgreiche Erkennung von Duplikaten ist entweder ein Netzwerk, in dem die Paketreihenfolge beibehalten wird, oder aber, daß alle Pfade vom Quell- zum Zielknoten etwa gleiche Paketlaufzeiten aufweisen. Das Zusammenspiel verschiedener Schicht-4-Timer gewährleistet die zuverlässige Erkennung von duplizierten Paketen.

## 7.2 Dienste im Transport Layer

Das Transport-Protokoll stellt folgende Dienste zur Verfügung:

- Zuverlässige Multicast- und Unicast-Übertragung: Sowohl Multicast innerhalb einer Gruppe als auch Multicast von einem externen Absender an eine Gruppe werden unterstützt.

- Unbestätigte wiederholte (unacknowledged-repeated) Multicast- und Unicast-Übertragung: Dabei wird keine Bestätigung erwartet und der Rahmen auf alle Fälle so oft wiederholt wird, wie im Wiederholungszähler (Retry Count) angegeben ist.

Abbildung 7 stellt die Schnittstelle der Transportschicht zu den übergeordneten Schichten dar. Mit `Send_Message() Request` wird die Zieladresse, die APDU (Application PDU) und das Prioritätsbit der abzusendenden Nachricht übergeben, `Receive_Message() Indication` meldet eine eingetroffene Nachricht und übergibt die APDU an die höheren Schichten, und `Transaction_Completed() Indication` meldet eine abgeschlossene Transaktion.

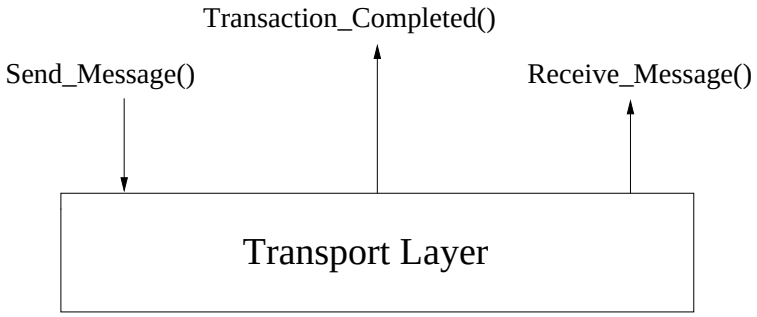


Abbildung 7: Schnittstelle zum Session Layer

## 7.3 Formate und Typen der Transport PDU (TPDU)

Die möglichen TPDU-Formate und die Typen der eingeschlossenen PDUs sind in der Abbildung 8 dargestellt. Das Schicht-4-Header-Byte beinhaltet das Authentication Bit, eine 3-bit-Codierung des eingeschlossenen PDU-Typs sowie die 4-bit-Transaktionsnummer. Im Anschluß erfolgt die Übertragung einer der 5 möglichen PDUs.

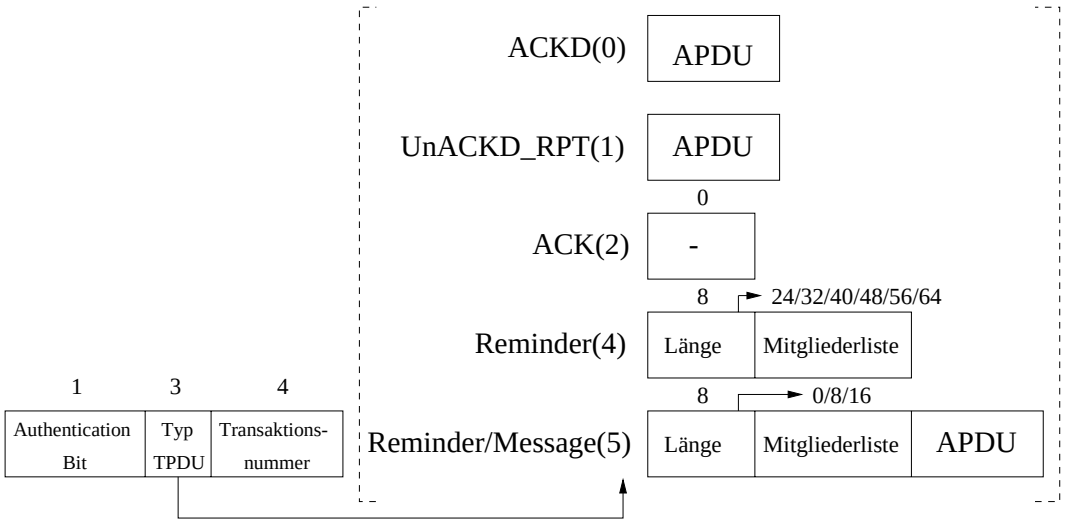


Abbildung 8: Formate und Typen der TPDU

## 7.4 Timer des Transport Layers

Das Transport-Protokoll bedient sich folgender Timer:

- Der *Transmit* Timer bestimmt den zeitlichen Abstand der Nachrichtenwiederholung auf Schicht 4.

- Der *Repeat Interval* Timer bestimmt den zeitlichen Abstand der Wiederholung für den unbestätigten Wiederholungsdienst (Unacknowledged Repeated).

- Der *Receive* Timer legt fest, wie lange ein Empfangsprozeß aktiv ist.

## 7.5 Authentication-Server

Das Authentication-Protokoll steht sowohl dem Transport Layer als auch dem Session Layer zur Verfügung. Die Grundlage bildet der Dienst der Erkennung von Duplikaten im Transport Control Sublayer.

Authentication (Beglaubigung) erlaubt dem Empfänger einer Nachricht, auf Wunsch die Identität des Absenders der Nachricht zu überprüfen. Dabei werden die beiden Teilnehmer mit Herausforderer (Challenger) und Herausgeforderter (Challengee) bezeichnet. Der Herausforderer initiiert den Prozeß der Authentication mit dem Absenden einer Zufallszahl Z an den Herausgeforderten. Dieser antwortet mit $V_G = S_G(Z, Nachricht)$, einer Verschlüsselung von Z sowie der ursprünglichen *Nachricht* unter Verwendung eines privaten Schlüssels $S_G$. Die Identität des Herausgeforderten wird nun überprüft, indem der Herausforderer die Verschlüsselung mit seinem eigenen Schlüssel durchführt $V_H = S_H(Z, Nachricht)$ und sein Ergebnis $V_H$ mit $V_G$ vergleicht. Verwenden sowohl Herausforderer als auch Herausgeforderter den gleichen Schlüssel $S_H = S_G$, so liefert das Authentication-Protokoll ein Pass-, im anderen Fall ein Fail-Signal.

### 7.5.1 Dienste im Authentication-Server

Die einzelnen Dienste im Authentication-Server stellt Abbildung 9 dar. Mittels `Initiate_Challenge()` wird der Herausforderer aufgefordert, den Authentication-Prozeß zu starten und eine 64-bit-Zufallszahl auszusenden. Der Herausgeforderte antwortet mit `Reply()` an den Herausforderer. Mit `Process_Reply()` wird der Vergleich der Ergebnisse durchgeführt und als Antwort Pass oder Fail zurückgemeldet.

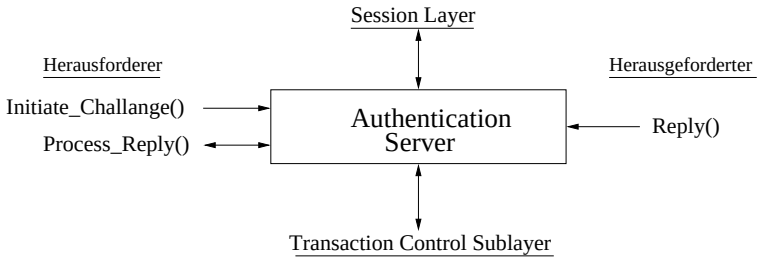


Abbildung 9: Die Dienste des Authentication-Server

### 7.5.2 Formate und Typen des Authentication-Server

Da der Authentication-Server sowohl dem Transport Layer als auch dem Session Layer zur Verfügung steht, obliegt dem Absender einer Nachricht, den Authentication-Prozeß durch Setzen des Authentication Bit in der TPDU bzw. SPDU (Session PDU) zu starten. Wird eine TPDU bzw. SPDU mit gesetztem Authentication Bit empfangen, so soll der Emfänger der Nachricht mittels Challenge-AuthPDU den Sender herausfordern. Dieser wiederum antwortet in einer Reply-AuthPDU. Die beiden AuthPDUs sind in der Abbildung 10 dargestellt.

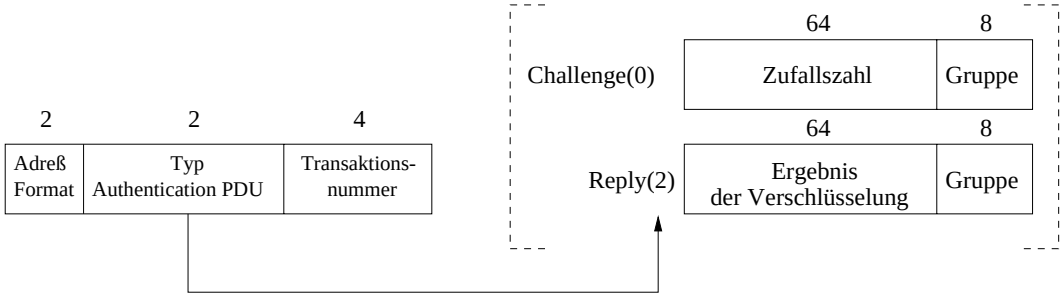


Abbildung 10: Formate und Typen der Authentication PDU

### 7.5.3 Der Verschlüsselungsalgorithmus

LonTalk verwendet nicht eine Verschlüsselung im eigentlichen Sinn. Die Nachricht wird unverschlüsselt übertragen, jedoch wird die Authtentizität des Absenders der Nachricht mit einem Schlüssel überprüft. Der Verschlüsselungsalgorithmus bedient sich dabei eines beliebigen 48-bit-Schlüssels S, einer 64-bit-Zufallszahl Z sowie der APDU von unterschiedlicher Länge.

Besondere Sorgfalt ist beim Setzen eines neuen Schlüssels mit dem Network-Management-Kommando erforderlich, da Nachrichten, wie bereits erwähnt, immer unverschlüsselt übertragen werden. Man kann dieses Problem verringen, indem man nur einen Differenzbetrag überträgt, der zum bereits bestehenden Schlüssel addiert wird.

# 8 Session Layer

Der Session Layer baut auf der korrekten SPDU-Reihenfolge sowie der Erkennung von Duplikaten im Transport Control Sublayer auf und stellt den Request-Response-Dienst zur Verfügung, der mit einem Remote Procedure Call (RPC) vergleichbar ist. Ein Client stellt eine Anfrage (Request) an einen Server und enthält von diesem eine Antwort (Response).

## 8.1 Schnittstelle des Session Layer

Die Schnittstelle des Session Layers hin zum Presentation Layer ist in der Abbildung 11 dargestellt.

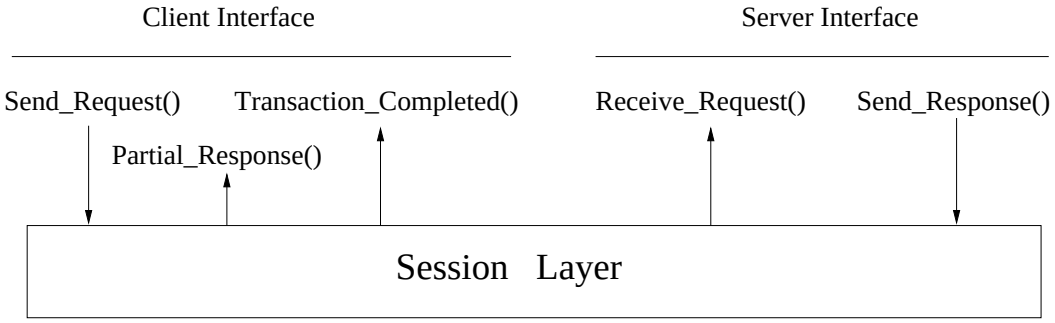


Abbildung 11: Schnittstelle des Session Layer

### 8.2 Formate und Typen der SPDU

Das Request-Response-Protokoll verwendet die vier PDU-Typen aus Abbildung 12. Das Rahmenlayout und die Protokollbearbeitung entsprechen den korrespondierenden PDU-Typen im Transport Layer (ACKD, ACK, Reminder, Reminder/Message).

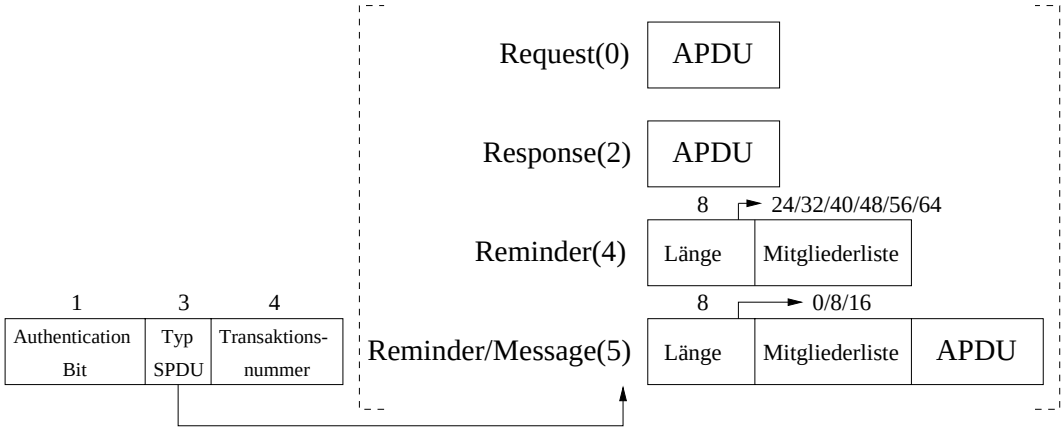


Abbildung 12: Formate und Typen der SPDU

### 8.3 Protokoll-Timer

Der Request-Response-Mechanismus bedient sich der beiden Timer Transmit und Receive. Die Funktionalität der beiden Timer ist identisch mit der Funktionalität der Timer im Transport Layer.

## 9 Presentation and Application Layer

Die auf einem Knoten laufende Applikation bedient sich der Dienste des Application Layers über die Dienstschnittstelle, die als Application Layer Interface (ALI) bezeichnet wird. Der unter dem Application Layer liegende Presentation Layer legt aufbauend auf einer korrekten APDU fest, wie die in der APDU enthaltenen Daten zu interpretieren sind. Die Dateninterpretation des Presentation Layer ist anwendungsunabhängig und gewährleistet den einfachen Nachrichtenaustausch zwischen Knoten verschiedenster Anwendungsgruppen.

Parallel zum Application Layer (siehe Abbildung 13) stellt LonTalk Network-Management- und Network-Diagnostik-Protokolle zur Verfügung. Diese werden im Abschnitt 10 beschrieben. Die beiden Protokolle und Presentation Layer fuktionieren aber nur dann, wenn alle darunterliegenden Schichten ordnungsgemäß arbeiten.

### 9.1 Funktionen des Presentation and Application Layers

Presentation und Application Layer unterstützen 5 Basisfunktionen:

- Network Variable Propagation: Diese Dienst sendet Nachrichten, die von dem/den Empfänger(n) als Netzwerkvariablen-Aktualisierung interpretiert werden.
- Generic Message Passing: Eine Anwendung kann eine beliebige Nachricht konstruieren, die dann mit allen zur Verfügung stehenden Adressierungsarten und Kommunikationsdiensten (siehe Abschnitt 7) versendet werden kann.

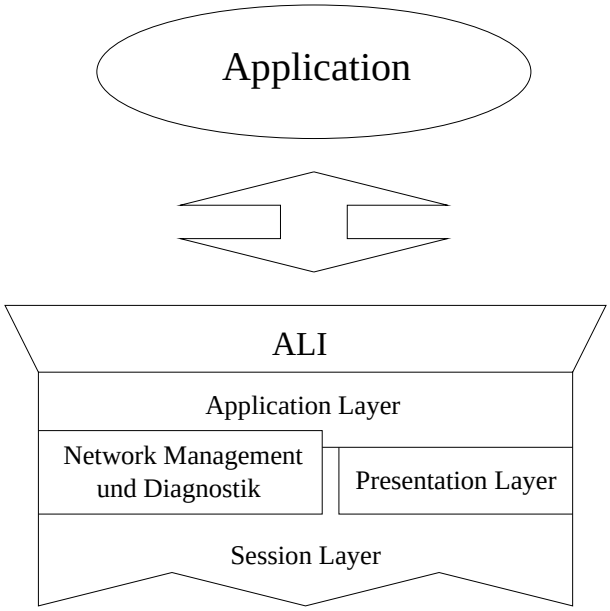


Abbildung 13: Einbettung von Application Layer, Network Management und Diagnostik

- Network Management Messages (NMM): Die von LonTalk zur Verfügung gestellten Dienste des NMM werden im Abschnitt 10 beschrieben.
- Network Diagnostik Messages: LonTalk unterstützt eine Reihe von Network-Diagnostic-Diensten. Diese werden auch im Abschnitt 10 beschrieben.
- Foreign Frame Transmission: LonTalk unterstützt damit die Übertragung von PDUs fremder Protokolle.

## 9.2 Formate und Typen der APDU

Die APDU besteht, wie Abbildung 14 zeigt, aus einem Header und einem Feld von Datenbytes.

## 9.3 Application Layer Interface

Das Application Layer Interface stellt dem Anwendungsprogramm die in Abbildung 15 präsentierten Dienste zur Kommunikation mit den anderen Knoten zur Verfügung. Die meisten Dienste bedürfen beim Aufruf keiner Parameter, sondern sie verwenden die Daten der Objekte msg_out, msg_in, resp_out und resp_in.

Die Syntax und die Funktionalität der Dienste des ALI sind wie folgt definiert:

- msg_alloc()
  Reservierung eines nichtpriorisierten Sendepuffer (Output Application Buffer)
- msg_alloc_priority()
  Reservierung eines priorisierten Sendepuffer (Priority Output Application Buffer)
- msg_send(msg_out)
  Sendung der in der Datenstruktur des Objekts msg_out übergebenen Daten.

0...n --> Die maximale APDU-Länge ist nicht definiert

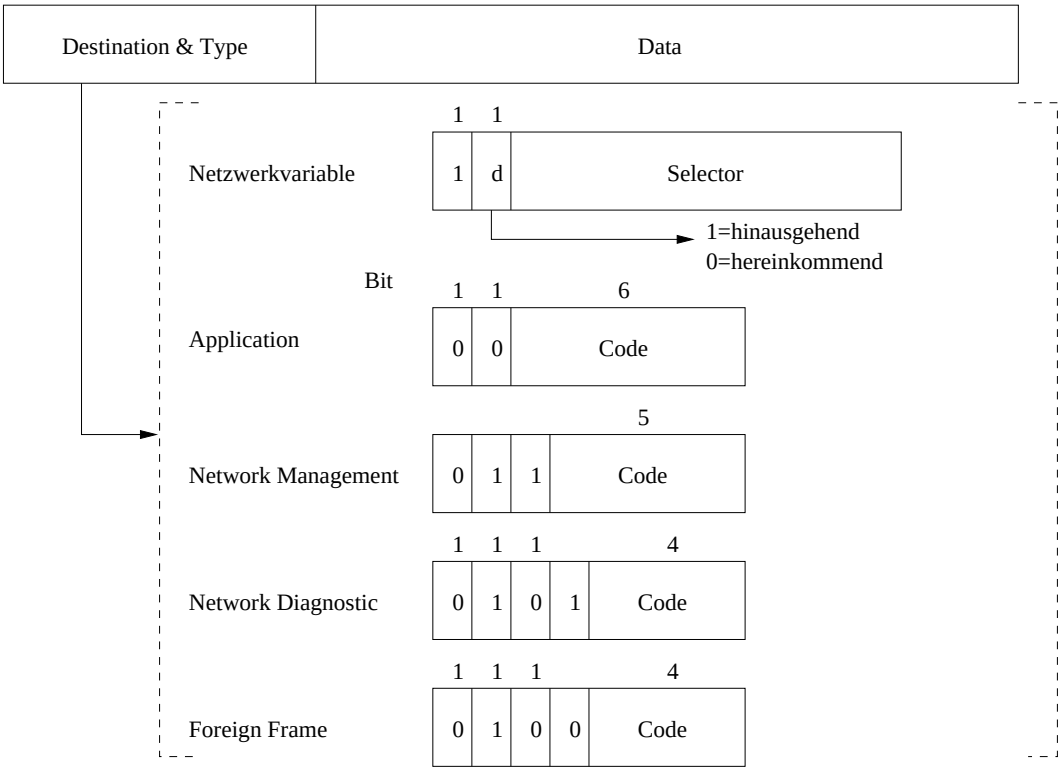


Abbildung 14: Rahmenformat der APDU

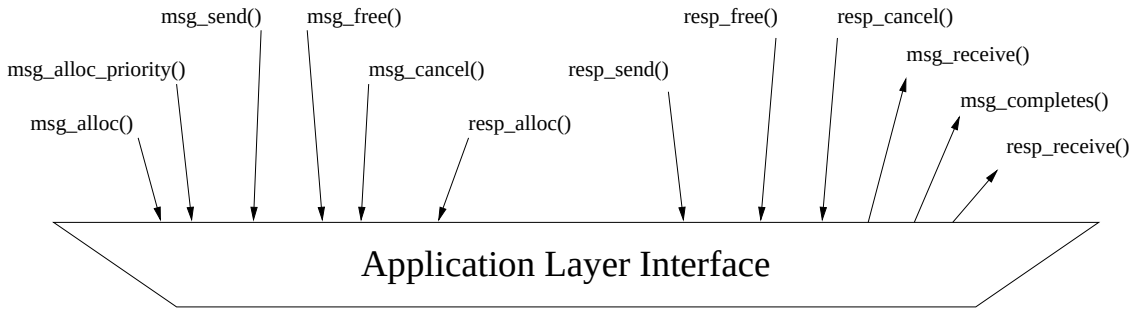


Abbildung 15: Festlegung der Dienste des ALI

- msg_cancel()
  Löschung der Nachricht, die gerade gebildet wird.
- msg_free(msg_in)
  Löschung aller Daten im Objekt msg_in.
- resp_alloc()
  Reservierung eines nichtpriorisierten Sendepuffer für eine zu sendende Antwort (Response).
- resp_send(resp_out)
  Sendung der in der Datenstruktur des Objekts resp_out übergebenen Antwort.
- resp_cancel()
  Löschung der Antwort, die gerade gebildet wird.
- resp_free(resp_in)
  Löschung aller Daten im Objekt resp_in.

- msg_receive(msg_in)
  Mitteilung der Anwendung über den Empfang einer neuen Nachricht.
- resp_receive(resp_in)
  Mitteilung der Anwendung über den Empfang einer neuen Antwort.
- msg_completes()
  Mitteilung über den Status einer bereits gesendeten Nachricht.

# 10 Netzwerkmanagement und Netzwerkdiagnostik

Netzwerkmanagement und Netzwerkdiagnostik-Protokolle (NM/ND) sind Protokolle, die auf dem Session Layer aufsetzen. Damit NM/ND in einem LON möglich ist, muß auf eine eindeutige, von LonTalk definierte, Semantik geachtet werden.

Mit wenigen Ausnahmen manipulieren NM/ND-Messages Speicherbereiche auf LonWorks-Knoten. Ein Teil der Dienste dient zur Bearbeitung von Datenfeldern, die die Konfigurationsdaten des LonWorks-Knotens enthalten. Die Adressen von Konfigurationsdaten im LonWorks-Knoten werden von den Knoten selbst verwaltet.

Die Netzwerkmanagement-Applikation stellt eine verteilte Anwendung nach dem Client/Server-Prinzip dar. Server-Funktionalität muß dabei auf allen Knoten gegeben sein, während Client-Funktionalität nur auf Netzwerkmanagement-Knoten vorhanden sein muß.

Es werden folgende Funktionalitäten von NM/ND unterstützt:

- Adreßzuweisungen,
- Abfragen von Knotenstatus und Knotenstatistiken,
- Bearbeiten von Router-Tabellen (Configured Router).

Netzwerkmanagementfunktionen sind mit wenigen Ausnahmen als RPCs über dem Session Layer zu implementieren. Für fast alle NM-Messages stehen alle Service-Typen (request/response, acknowledged, unacknowledged und unacknowledged/repeated) zur Verfügung.

# 11 Zusammenfassung

In dieser Arbeit wurden folgende Themen bzw. Begriffe erläutert:

- Kontrollnetzwerke, ihre Eigenschaften und Anwendungsbereiche,
- das LonTalk-Protokoll und sein Schichtenmodell,
- LonTalk-Adressen,
- einzelne Schichten des LonTalk-Protokoll-Schichtenmodells,
- das Netzwerkmanagement und die Netzwerkdiagnostik.

Zu den Besonderheiten, welche die Kontrollnetzwerke und ihr Protokoll von anderen unterscheiden, gehört, daß die Kontrollnetzwerke einfach zu realisieren, robust und vielseitig einsetzbar sind. Ihre Netzwerktechnik bietet die Möglichkeit, unterschiedlichste Übertragungsmedien zu nutzen. Das LonTalk-Protokoll enthält Sicherheits- und Transaktionsmechanismen. Im Protokoll ist auch Netzwerkmanagement definiert.

## Literatur

[DiLS98] Dietmar Dietrich, Dietmar Loy und Hans-Jörg Schweinzer (Hrsg.). *LON-Technologie. Verteilte Systeme in der Anwendung.* Hüthig Verlag Heidelberg. 1998.

[Eche94] Echelon Corporation. *LonTalk Protocol Specification. Version 3.0*, 1994.

# Universal Serial Bus – Funktionsweise und Protokolle

Andreas Jellinghaus

**Kurzfassung**

Zum Anschluß neuer Peripherie an den PC wurden in den letzten 15 Jahren immer wieder neue Lösungen entwickelt. Viele davon sind bis heute in Gebrauch, und so bietet ein aktueller Multimedia PC separate Schnittstellen für fast jedes Gerät – Maus, Tastatur, Lautsprecher, Mikrophon, Videokamera, Drucker, Modem, Scanner – wollen ihre eigenen Schnittstellen. Mit dem Universal Serial Bus (USB) wurde eine gemeinsame Schnittstelle geschaffen für all diese Geräte und viele mehr. Diese Schnittstelle bietet über Verteiler die Möglichkeit bis zu 127 Geräten anzuschließen und birgt viele Verbesserungen in sich, wie Hot Plugging (Zustecken von Geräten zur Laufzeit), Power Management, isochrone Datenströme für Multimedia-Anwendungen und mehr. 3 Jahre nach der Verabschiedung des USB Standard 1.0 beginnt USB sich langsam zu etablieren. Grund genug einen Blick auf die Funktionsweise und die verwendeten Protokolle zu werfen.

## 1 Einführung

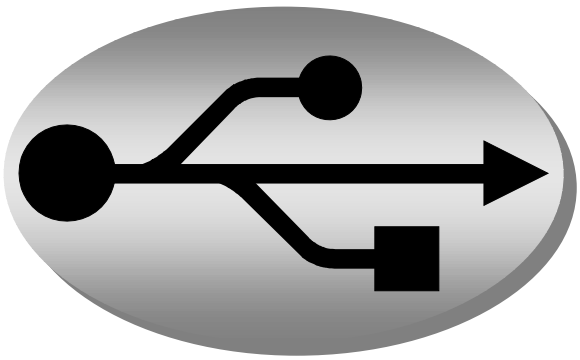

Abbildung 1: USB Logo

Der *Universal Serial Bus* (USB) verändert die PC-Welt durch einen neuen Ansatz zum Anschluß von Peripherie Geräten. Dies wirkt sich nicht nur auf Stecker und Kabel aus, sondern bietet neue Erweiterbarkeit, *Hot Plug und Play*, Kombigeräte, schnelle Datenübertragung, vier Klassen von Transferarten und ein geregeltes *Powermanagement*. USB bietet nicht nur neue Möglichkeiten und einfachere Benutzung, es sinken auch für Hersteller und Anwender die direkten und indirekten Kosten.

### 1.1 Stecker und Kabel

An der Rückseite des PC findet man ein knappes Dutzend Buchsen, und jedes Gerät benötigt eigene Kabel und Stecker. Fehler bei der Verkabelung sind üblich, teilweise – ohne gute Beschriftung – kann man nur durch Probieren den richtigen Anschluß finden.

Mit USB gibt es einheitliche Stecker und Kabel. Der PC wird nur noch zwei oder vier Anschlüsse bieten. Über welche Buchse ein Gerät angeschlossen wird spielt keine Rolle mehr. Es gibt nur einen Kabeltyp. Stecker und Buchsen wurden neu entworfen. Verwechslungen mit der alten Verkabelung sind dadurch ausgeschlossen.

Die Kabel besitzen nur noch vier Leitungen, Stecker haben vier Pins. Somit lassen sich beide günstiger herstellen als Kabel mit vielen Leitungen und komplexen Steckern.

## 1.2 Erweiterbarkeit

Bisher brauchen neue Geräte oft eine neue, schnellere Schnittstelle, und diese muß über eine Erweiterungskarte eingebaut werden. Jede Erweiterungskarte benötigt Ressourcen wie *Interrupt* Leitung, *I/O* Bereich oder einen eigenen Speicherbereich.

Bei aktuellen PC's ist die USB-Schnittstelle bereits Bestandteil der Hauptplatine. Die Schnittstelle verbraucht nur einmal Ressourcen, auch wenn es mehrere USB-Buchsen gibt, und unabhängig davon wieviel Geräte angeschlossen werden. Weitere Geräte können über Verteiler (*Hubs*) angeschlossen werden, es sind bis zu 127 Geräte in bis zu 6 Verschachtelungsebenen möglich.

## 1.3 Hot Plug and Play

Bisher mußten alle Geräte angeschlossen und eingeschaltet sein, bevor man den PC anschaltet. Wurde ein Gerät vergessen, so war oft ein Neustart notwendig.

USB stellt neue Anforderungen: Hardware und Software müssen das Hinzufügen und Abziehen von Geräten im laufenden Betrieb erlauben, müssen Änderungen selbständig erkennen und notwendige Treiber laden und konfigurieren.

## 1.4 Kombigeräte

USB erlaubt es, Geräte mit mehreren Funktionen zu entwickeln, zum Beispiel eine Tastatur mit eingebautem Lautsprecher und USB-Verteiler. Die Konzepte in USB sehen dies vor und vereinfachen die Ansteuerung solcher Geräte, zum Beispiel sind keine speziellen Treiber notwendig, es können die normalen Treiber verwendet werden.

## 1.5 Datenübertragung

Ein normales USB-Gerät kommuniziert in einem seriellen Verfahren mit einer Geschwindigkeit von 12 MBit/s (*Full Speed*). Geräte können aber auch ausschliesslich mit einer reduzierten Geschwindigkeit von nur 1.5 MBit/s (*Low Speed*) kommunizieren. Diese langsame Übertragung reicht für Geräte wie eine Maus aus und ist günstiger zu fertigen. Verteiler sind immer *Full Speed* Geräte und verstehen daher beide Geschwindigkeiten.

## 1.6 Transferarten

USB kennt vier verschiedene Transferarten:

*Control Transfer* dient zum Erkennen und Konfigurieren des Gerätes.

*Interrupt Transfer* erlaubt kleine Datenmengen schnell zu verschicken. USB wird jedoch nicht von den Geräten unterbrochen, sondern fragt die Geräte regelmässig ab (*Polling*). Wichtig ist hierbei, die Daten bald abzuliefern (Beispiele: Bewegung der Maus, Tastendruck, Bewegung des Joysticks).

*Bulk Transfer* überträgt viele Daten auf einmal und möglichst kompakt. Wichtig ist, die Daten schnell zu übertragen, kurze Pausen stören nicht und ein exaktes Timing ist nicht erforderlich (Beispiele: Drucker, Scanner, externe Massenspeicher).

*Isochronous Transfer* überträgt periodisch Daten, allerdings ohne Sicherung gegen Übertragungsfehler. Wichtig ist die konstante, zeitgenaue Übertragung, einzelne Bitfehler stören in Audiodaten nicht (Beispiele: Lautsprecher, Mikrophon).

Mit Ausnahme der isochronen Übertragung wird eine fehlerfreie Übertragung sichergestellt, d.h. fehlerhafte Übertragungen werden wiederholt.

## 1.7 Stromversorgung

Verteiler mit eigener Stromversorgung wie der PC liefern bis zu 500 mA pro Anschluß bei 5 V Spannung. Verteiler ohne eigene Stromversorgung liefern bis zu 100 mA. Nach 3 Millisekunden Inaktivität schläft ein Gerät ein und verbraucht maximal 2.5 mA Strom.

## 1.8 Kosten

Für den Benutzer sinken die indirekten Kosten, weil der Anschluß eines neuen Gerätes deutlich einfacher wird und Konfigurationsarbeit entfällt. Dies wird oft unter dem Stichwort *Total Cost of Ownership* angeführt.

Da alle Geräte die gleichen Kabel, Buchsen und Stecker verwenden, sinken die Kosten beim Hersteller gegenüber eigenen Lösungen deutlich, insbesondere wenn eine Steckkarte eingespart werden kann.

## 1.9 Beispielgeräte

Typische USB-Geräte sind Tastaturen, Maus, Drucker, Scanner, Joystick, Modem, Lautsprecher, Mikrophon, Videokamera und externe Laufwerke.

Eine Sonderstellung haben Monitor und Netzwerke inne: Monitore können USB-Hub (Verteiler) sein und über USB konfigurierbar sein, aber das Videosignal wird weiterhin über ein eigenes Kabel transportiert. Netzwerke benutzen weiterhin eigene Kabel, lediglich entsprechend zu bisherigen Nullmodemkabeln gibt es eine Möglichkeit für kleine Netzwerke. Dazu wird aber kein besonderes Kabel, sondern ein spezielles Gerät als Vermittler benötigt.

Reicht die Übertragungsgeschwindigkeit von 12 MBit/s nicht aus, so gibt es zwei mögliche Alternativen: USB Version 2.0 soll eine dritte, noch schnellere Übertragungsgeschwindigkeit von voraussichtlich 500 MBit/s einführen. Mit IEEE 1394 (*Firewire*) wird bereits ein konkurrierendes Verfahren eingesetzt.

## 1.10 Literatur

Eine umfassende Betrachtung von USB findet man in Buch von Hans Joachim Klein [Klei99], sowie in diversen Zeitschriften Artikeln ([Stra95], [Schn97], [Schn98], [Sier98]). Eine Implementierung findet man im Linux-Quellcode [Torv99]. Für eigene Entwürfe von Treibern oder Hardware sollte man aber unbedingt den USB Standard 1.1[Comp98] heranziehen.

# 2 Struktur

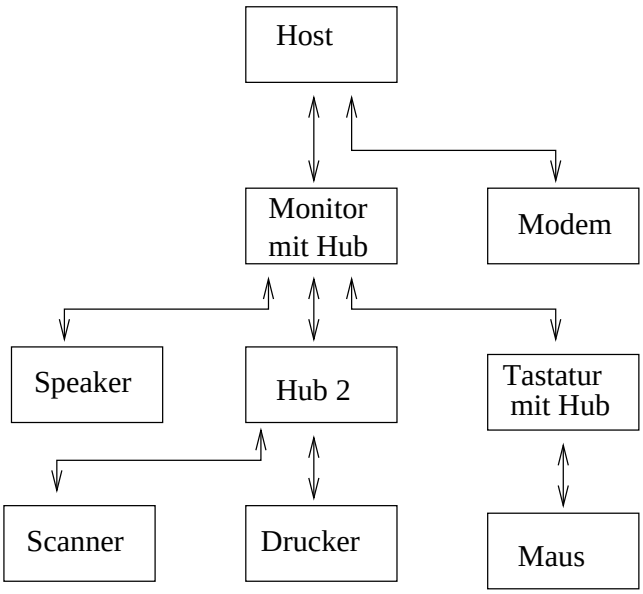


Abbildung 2: Physikalische Struktur

Physikalisch bildet USB einen Baum (oder Wurzel, oder Pyramide, oder *Tired Star*), beginnend beim PC als Wurzel über Verteiler (*Hubs*) als Knoten bis zu den Geräten als Blätter. Logisch dagegen bildet USB einen Stern – alle Geräte kommunizieren direkt mit dem Host. USB-Geräte können nicht untereinander kommunizieren, nur mit dem *Host*.

Ein Beispiel für eine USB-Verkabelung von PC, Monitor, Modem, Lautsprecher, Tastatur, Maus, Scanner und Drucker findet man in Abbildung 2.

# 3 Einteilung der Geräte

## 3.1 Funktion – Function

Ein einfaches Gerät besitzt eine Funktion. Bei einem Kombigerät, beispielsweise eine Tastatur mit eingebautem Lautsprecher und *Hub*, wird jedes Teilgerät als Funktion bezeichnet. Besagte Tastatur ist dann ein Gerät mit drei Funktionen.

Diese Unterscheidung wird benötigt, weil die Kommunikation fast immer zwischen dem Host und der Funktion stattfindet. Nur das Auslesen und Setzen von Konfigurationsparametern wird über die *Control Pipe* des Geräts – ein gemeinsamer Kanal für alle Funktionen – abgewickelt.

## 3.2 Der Verteiler – Hub

Der *Hub* ist selbst an einen anderen *Hub* oder den *Host* direkt angeschlossen. Der *Hub* kann eine eigene Stromversorgung haben oder über USB mit Strom versorgt werden.

Der Verteiler überwacht seine Anschlüsse und meldet alle Änderungen an den *Host*, zum Beispiel das Anstecken eines neuen Gerätes. Der Hub kann erkennen ob ein *Low-Speed*-Gerät oder ein *Full-Speed*-Gerät angeschlossen ist. Ein Anschluß wird erst auf Befehl des *Host*s aktiviert und kann durch Befehl des *Host*s abgeschaltet oder zurückgesetzt werden.

Jede Verbindung ist eine Punkt-zu-Punkt-Verbindung. Der Verteiler regeneriert das Signal vom *Host* und schickt es auf allen aktiven Anschlüssen wieder raus. Ein Paket wird jedoch nicht komplett zwischengespeichert und Prüfsummen werden beim Weiterleiten nicht beachtet. Grenzwerte für die Latenzzeiten finden sich im USB-Standard 1.1 [Comp98].

Erst nachdem ein *Low-Speed*-Zeichen empfangen wurde, schaltet der Verteiler auf die niedrige Übertragungsgeschwindigkeit und leitet das folgende Paket an *Low-Speed*-Geräte weiter. Die normale Datenübertragung mit *Full Speed* wird nicht an diese Geräte weitergeleitet.

Signale von angeschlossenen Geräten werden nur an den *Host* weitergeleitet, nicht an die anderen Anschlüsse.

Empfängt ein *Hub* vom übergeordneten Geräte ein *Reset* Signal, so gibt er es an alle Anschlüsse weiter und setzt auch sich selbst zurück.

## 3.3 Der PC als Host Controller

Der *Host* oder *Host Controller* verwaltet den gesamten Bus:

- jedes Gerät konfigurieren und aktivieren.
- genau jede Millisekunde ein Zeitsignal an alle Geräte schicken.
- aus den Anforderungen der Geräte einem exakten Zeitplan erstellen.
- alle Geräte gemäß dem Zeitplan regelmäßig abfragen.
- auf Entfernen und Zuschalten von Geräten reagieren.

Zudem handelt der *Host* auch gleichzeitig als ein Verteiler und muß seine eigenen Anschlüsse verwalten.

Kein Gerät sendet oder empfängt Daten ohne Anweisung des *Host*.

## 3.4 Adressierung

Jedes Gerät besitzt nach dem Einschalten zunächst die Adresse 0. Der *Host* liest die Konfiguration aus, und vergibt dann eine freie Adresse. Die Adressen liegen im Bereich 0 bis 127.

Adressiert werden sogenannte Endpunkte im Gerät. In jedem Gerät gibt es den Endpunkt 0, dieser wird für *Control Transfer* benutzt. Nur dieser Endpunkt ist bidirektional.

Jede Funktion kann einen oder mehrere weitere Endpunkte besitzen. Jeder Endpunkt ist festgelegt auf einen Typ: *Bulk Transfer*, *Isochronous Transfer* oder *Interrupt Transfer*, und auf eine Richtung: von *Host* zu Funktion oder von Funktion zu *Host*.

Zusätzlich zum Endpunkt EP0, der *Control Pipe*, sind 15 weitere Endpunkte der Richtung vom Host zur Funktion (*OUT*) und 15 weiter Endpunkte in Gegenrichtung (*IN*) möglich.

Die Numerierung der Endpunkte nimmt der Hersteller beim Erzeugen der Konfiguration vor, diese ist im USB Gerät gespeichert und wird durch den *Host* ausgelesen.

### 3.5 Timing

Der *Host* ist für exakte Zeitsteuerung verantwortlich. Genau jede Millisekunde muß ein Zeitsignal übertragen werden. Dannach werden isochrone Datenströme berücksichtigt, darauf folgen *Interrupt*- und *Control*-Übertragungen (soweit notwendig) und die verbleibende Zeit bis zum nächsten Zeitsignal kann für *Bulk Transfer* genutzt werden.

In Abbildung 3 ist ein Beispiel aufgeführt: Senden und Empfangen von Sprachdaten (*Tx* und *Rx Voice*), zur Telefonzentrale (*Tx* und *Rx Line*), Stereo Musik, diverse *Control* und *Interrupt* Daten (z.B. Tastatur, Verteiler), *Low-Speed*-Datenübertragung für die Maus und Nutzung der verbleibenden Zeit für den Datentransfer mit dem Scanner.

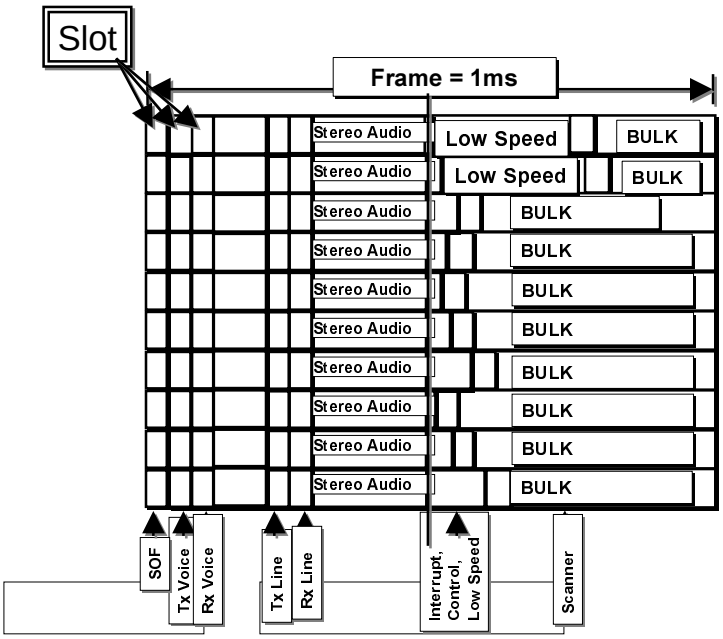


Abbildung 3: Slotverwaltung für exaktes Timing

## 4 Physikalische Schicht

| Leitung | Pin | Adern Farbe | Bedeutung |
|---|---|---|---|
| VCC | 1 | Rot | Stromversorgung |
| D- | 2 | Weiß | Datenleitung |
| D+ | 3 | Grün | Datenleitung |
| GND | 4 | Schwarz | Erdung |

Tabelle 1: Belegung von Stecker und Leitung

USB-Geräte werden über ein vieradriges Kabel verbunden: Eine verdrillte Doppelader zur differentiellen Datenübertragung (D+, D-), je eine Ader für Spannung und Erdung. Das Kabel für *Low-Speed*-Geräte kommt mit einer moderaten Isolation aus, für die *Full-Speed*-Geräte benötigt man eine bessere Isolation. Exakte Grenzwerte, Anforderungen etc. finden sich in der USB-Spezifikation [Comp98].

Ein Kabel verfügt immer über einen Stecker Typ B mit dem es an den Verteiler (Hub) angeschlossen wird. Der Verteiler besitzt mehrere Buchsen vom Typ B.

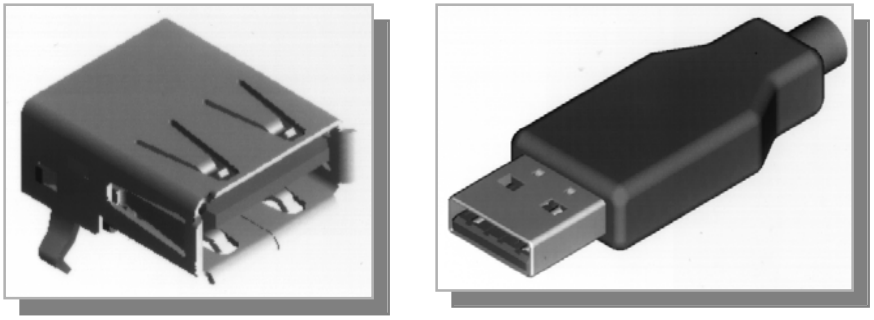


Abbildung 4: Stecker und Buchse USB Typ B

Bei *Full-Speed*-Geräten kann das Kabel abnehmbar sein und wird dann über Stecker und Buchse Typ A an das Gerät angeschlossen.

Bei *Low-Speed*-Geräten muß das Kabel fest am Gerät befestigt sein. Durch diese Einschränkung wird verhindert, dass versehentlich schwach isolierte Kabel für *Full-Speed*-Übertragung eingesetzt werden.

Kabel mit zwei Steckern Typ A oder zwei Steckern Typ B sind nicht zulässig. Stecker und Buchsen von Typ A und Typ B unterscheiden sich deutlich voneinander und von allen bisher bekannten PC-Steckern (Seriell 9, Seriell 25, Parallel, Centronix, DIN-Tastatur, PS/2, SCSI, SCSI-wide, Joystick, Busmaus, Audio, 10BaseT, 10Base2, 10Base5, Strom). Dadurch werden Verwechslungen vermieden, falsches Einstecken ist nicht möglich.

Es gibt auch keine Umstecker mehr wie bei seriellen Schnittstellen (25/9 Pin), nicht mehrere Sorten Kabel wie bei seriellem Anschluß und Nullmodem, oder Verwechselungsmöglichkeiten wie serielle und parallele Verlängerungen.

Jeder Verteiler (Hub) kann erkennen, ob Geräte angeschlossen sind, und von welchem Typ die Geräte sind: bei Geräten für die schnelle Datenübertragung wird die Datenleitung D+ über einen 1.5kΩ Wiederstand auf Masse gelegt, bei Geräten für die langsame Datenübertragung wird dies mit der Datenleitung D- gemacht. Daher kann auch erkannt werden, wenn kein Gerät vorhanden ist. Jede Änderung wird dem PC (Host) mitgeteilt.

Der Kabeltyp und die maximale Kabellänge werden im Standard festgelegt.

## 5 Signalkodierung

Die Daten werden mit NRZI-Kodierung (Non Return to Zero Inverted) in Signale umgesetzt. Dabei bedeutet eine 1 keinen Wechsel der Polarität, bei einer 0 wird die Polarität gewechselt. Damit der Takt auch bei einer langen Folge von Einsen erhalten bleibt, wird das Bit-Stuffing Verfahren eingesetzt: wenn sechs Einsen in Folge auftreten wird danach eine 0 eingefügt, und auf der Empfangsseite wieder entfernt.

Jedes Paket wird von einem *Sync*-Signal eingeleitet. Das *Sync*-Signal entspricht den DatenBits 00000001, wechselt also 7 mal die Polarität und ermöglicht so allen Geräten, auf diesen Takt zu synchronisieren.

Jedes Paket endet mit einem End of Packet (EOP) Signal. Dazu werden kurzzeitig beide Leitungen auf Masse gelegt. Eine Ausnahme ist die Signalisierung für *Low-Speed*-Datenübertragung: hier wird kein EOP Signal angefügt. Durch das EOP-Signal des folgenden Pakets endet die *Low-Speed*-Kommunikation.

Alle Daten werden mit *Little Endian* versendet. Bei 16-Bit-Werten wird also zuerst das Byte mit den niedrigen Wertigkeiten $2^0$ bis $2^7$ verschickt, danach das Byte mit den Wertigkeiten $2^8$ bis $2^{15}$.

# 6 Pakete

Jedes Paket beginnt mit einem *Packet Identifier*. Dieses Feld ist ein Byte, also 8 Bit lang. In diesem Feld steht der 4-Bit-Pakettyp, erst normal, dann invertiert. Dieses Feld wird bei jedem Empfang geprüft. Eine genaue Aufstellung findet man in Tabelle 2.

| Art | Name | Wert | Bedeutung |
|---|---|---|---|
| Token | OUT | 0001 | Adressierung, vom PC zum Gerät |
| | IN | 1001 | Adressierung, vom Gerät zum PC |
| | SOF | 0101 | Zeitsignal (Start-of-Frame) |
| | SETUP | 1101 | Konfigurationsbefehl von PC an Gerät |
| Daten | DATA0 | 0011 | Datenpaket |
| | DATA1 | 1011 | Datenpaket |
| Quitierung | ACK | 0010 | Bestätigung, Paket fehlerfrei empfangen |
| | NAK | 1010 | Fehler beim Senden oder Empfangen |
| | STALL | 1110 | Endpunkt ist angehalten oder *Control Pipe* nicht unterstützt |
| Speziell | PRE | 1100 | Präambel für das Senden mit *Low Speed* (1.5 MBit/s statt 12 MBit/s) |

Tabelle 2: Belegung des *Packet Identifier*

## 6.1 Token Pakete

*Token* sind Pakete mit der Kennung IN, OUT oder SETUP. Ein *Token* wird immer vom PC (*Host*) geschickt, um den Kommunikationspartner zu adressieren: bei OUT und SETUP wird bestimmt, wer die Daten erhält, bei IN, wer Daten schicken soll (siehe Abbildung 5).

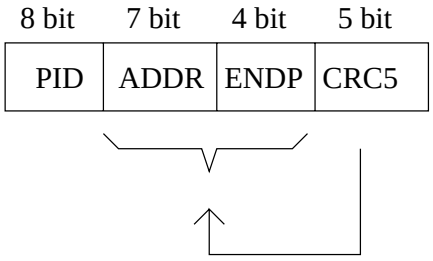


Abbildung 5: Aufbau eines Token-Paketes

Ein *Token* besteht aus 8 Bit für das PID-Feld, 7 Bit für die Adresse des Gerätes, 4 Bit für den Endpunkt und eine 5 Bit CRC Prüfsumme über Adresse und Endpunkt. CRC-5-Prüfsummen werden gemäß der Formel $G(X) = X^5 + X^2 + 1$ berechnet.

## 6.2 Start-of-Frame-Pakete

*Start-of-Frame*-Pakete werden vom PC (*Host*) genau jede Millisekunde ausgeschickt und von allen Verteilern (*Hubs*) und *Full-Speed*-Geräten empfangen. Mit diesen Paketen kann jedes

Gerät die Zeit exakt mitverfolgen. Geräte, die nicht an Zeitinformationen interessiert sind, können den Paketinhalt ignorieren.

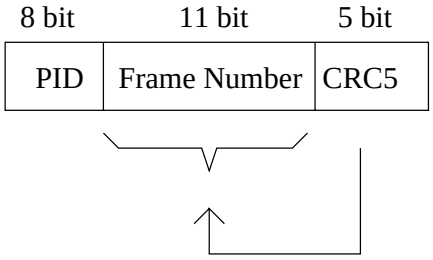


Abbildung 6: Aufbau eines Start-of-Frame Paket

Ein Start-of-Frame-Paket besteht aus 8 Bit für das PID-Feld, 11 Bit für eine fortlaufende Paketnummer und eine 5 Bit CRC-Prüfsumme über die Paketnummer.

### 6.3 Datenpakete

Datenpakete werden mit dem Paket Typ DATA0 oder DATA1 verschickt. Ein Datenpaket besteht aus einem Byte PID (8 Bit), 0 bis 1023 Byte Daten und einer 2 Byte (16 Bit) CRC Prüfsumme über die Daten. CRC-16-Prüfsummen werden gemäß der Formel $G(X) = X^{16} + X^{15} + X^2 + 1$ berechnet.

### 6.4 Quittungspakete

Ein Quitierungspaket besteht nur aus dem *Packet-Identifier*-Feld. Wenn ein *Token* oder ein Datenpaket beschädigt wurde, wird keine Quittung geschickt. Der PC (*Host*) kann eine Funktion durch einen *HALT*-Befehl anhalten. In diesem Zustand werden alle *IN*-Pakete und alle *OUT*-Datenpakete mit einer STALL-Quittung beantwortet.

Wird einem Gerät durch ein *IN*-Paket die Möglichkeit gegeben, Daten zu senden, so signalisiert es mit *NAK*: es kann nichts senden. Wurden einem Gerät ein OUT-Paket und ein Datenpaket geschickt und konnte das Gerät das Datenpaket nicht annehmen, so schickt das Gerät ein *NAK*. Der PC (*Host*) schickt niemals *NAK*-Kommandos. *SETUP*-Pakete werden niemals abgelehnt – die Funktion muß immer bereit sein, Befehle entgegenzunehmen.

Wenn alles funktioniert und ein Datenpaket korrekt übertragen wurde, schickt der Empfänger ein ACK. Dies gilt für *Host*s und Geräte. Auch wenn der PC (Host) eine *Control Pipe* fehlerhaft anspricht, antwortet die Funktion mit einem *STALL*-Paket.

## 7 Datenübertragung

Isochrone Daten zu senden ist ganz einfach: Adressierung durch ein *OUT*-Paket und anschließendem Senden der Daten durch den *Host*. Keine Quittung, keine Fehlerprüfung und keine erneuten Versuche.

Isochrone Daten empfangen ist ebenso einfach: Adressierung durch ein *IN*-Paket, Senden des Daten durch die Funktion. Keine Quittung, keine Fehlerprüfung und keine erneuten Versuche.

Das Senden von Daten an eine Funktion läuft in den drei folgenden Schritten ab: Adressierung durch ein *OUT*-Paket, Senden des Daten durch den *Host* und Quittierung durch die Funktion. Dieser Ablauf ist für *Bulk Transfer* und *Interrupt Transfer* identisch.

Daten von einer Funktion abzurufen läuft in den drei folgenden Schritten ab: Adressierung durch ein *IN*-Paket, Senden der Daten durch die Funktion und Quittierung durch den Host. Identisch für *Bulk Transfer* und *Interrupt Transfer*.

Ein normaler Befehl an eine Funktion erfolgt in zwei Phasen: In der ersten Phase wird der Befehl übermittelt und quittiert. Dazu schickt der *Host* ein *SETUP-Token*-Paket, der *Host* schickt ein Datenpaket und die Funktion schickt ein Quittungspaket. Damit wird sichergestellt, dass der Befehl korrekt übertragen wurde.

In der zweiten Phase wird geprüft, ob die Funktion den Inhalt auch verstanden hat. Dazu schickt der *Host* ein *IN-Token*-Paket, die Funktion schickt ein leeres Datenpaket als Bestätigung, und der *Host* bestätigt mit einem Quittungspaket.

Eine Befehlsanfrage an eine Funktion erfolgt in drei Phasen: In der ersten Phase wird der Befehl übermittelt und quittiert. Dazu schickt der *Host* ein *SETUP-Token*-Paket, der *Host* schickt ein Datenpaket und die Funktion schickt ein Quittungspaket. Damit wurde sicher gestellt, dass der Befehl korrekt übertragen wurde. In der zweiten Phase antwortet die Funktion mit dem gewünschten Inhalt. Dazu schickt der *Host* ein *IN Token* Paket, die Funktion schickt ein Datenpaket mit dem Inhalt und der *Host* bestätigt mit einem Quittungspaket.

Oft reicht ein Paket nicht aus, um alle gewünschten Daten zu liefern, daher wird die zweite Phase so lange wiederholt, bis alle Daten übertragen sind. Beide Seiten führen Buch darüber, welcher Teil der Antwort als nächstes Übertragen wird. Diese Information wird nicht übermittelt.

In der dritten Phase bestätigt der Host den korrekten Empfang der Daten. Dazu schickt der *Host* ein *OUT-Token*-Paket, der *Host* schickt ein leeres Datenpaket als Bestätigung, und die Funktion bestätigt mit einem Quittungspaket.

## 7.1 Funktion nicht bereit

Da der Host jede Transaktion einleitet, ist er immer bereit. Wenn eine Funktion nicht bereit ist, und Daten senden soll, so schickt sie statt Daten ein *NAK*-Signal. Wenn eine Funktion nicht bereit ist und (nicht isochrone) Daten empfangen soll, so antwortet sie auf die Daten mit einem *NAK*-Signal.

## 7.2 Funktion deaktiviert

Wenn eine Funktion deaktiviert ist und Daten senden soll, so schickt sie statt Daten ein *STALL*-Signal. Wenn eine Funktion nicht bereit ist und (nicht isochrone) Daten empfangen soll, so antwortet sie auf die Daten mit einem *STALL*-Signal.

## 7.3 Fehler bei der Übertragung

Keine Fehlerbehandlung für isochrone Übertragung, da verspätete Pakete als nutzlos angesehen werden. Ein Paket mit defektem PID-Feld, defekter CRC-Prüfsumme oder Bit-Stuffing-Fehler wird verworfen.

Bei *Control Transfer*, *Interrupt Transfer* und *Bulk Transfer* wird als letzter Schritt immer eine Quittung gesendet. Bleibt diese aus, wird ein Fehler erkannt. Erkennt der Host den Fehler, so wiederholt er den Datenaustausch.

### 7.4 Erkennung von Doppelten

Pro Endpunkt speichern Host und Funktion ein Bit. Dieses Bit bestimmt ob Data0 oder Data1 gesendet bzw. erwartet wird. Beim Senden und Empfangen eines *ACK-Token*-Pakets wird jeweils das Bit gekippt.

Bei einer *OUT*-Übertragung: ist ein Fehler im PID- oder Datenteil aufgetretten, wurde kein *ACK* versendet und das Bit ist noch auf beiden Seiten gleich. Bei der erneuten Übertragung gibt es keine Probleme. Wurde dagegen ein *ACK*-Befehl erzeugt, ist aber nicht angekommen, so stimmen die Bits nicht überein. Wenn nun die Übertragung erneut durchgeführt wird, fällt der Funktion das falsche Datenbit auf. Die Daten werden ignoriert, aber es wird ein *ACK* geschickt. Die Funktion ändert hierbei ihr Datenbit nicht. Sobald die Bestätigung ankommt, sind *Host* und Funktion wieder synchron.

*IN* Transaktionen werden mit entsprechend vertauschten Rollen, aber identisch, ausgeführt. Geht hier die Bestätigung des *Host*s verloren, wird dies erst beim nächsten Transfer bemerkt.

Eine *SETUP*-Transaktion wird genau wie eine *IN*-Transaktion ausgeführt. Die folgenden *IN*- und *OUT*-Transaktionen ebenfalls. Es kann sich jedoch ein Fehler einschleichen, wenn das *ACK*-Paket der letzten *IN*-Transaktion verloren geht. Dies wird folgendermaßen gelöst: eine *SETUP*-Transaktion benutzt immer Data0, die erste *IN*-Transaktion benutzt immer Data1. Bei 3-Phasen-Transaktionen benutzt die *OUT*-Transaktion immer Data1.

### 7.5 Low Speed Geräte

Möchte der *Host* ein *Low-Speed*-Gerät ansprechen, so schickt er vor jedem Paket ein *Sync*-Feld und ein *PRE-Token*-Feld. Anschließend wird eine kurze Pause eingelegt, in welcher alle *Hubs* ihre *Low-Speed*-Anschlüsse aktivieren und dann kommuniziert der *Host* mit *Low Speed*.

Auf das *PRE Token* folgt kein *EOP*-Signal. Erst am Ende des *Low-Speed*-Pakets kommt das normale *EOP*-Signal, woraufhin der *Hub* wieder die *Low-Speed*-Anschlüsse deaktiviert.

## 8 USB Datenstrukturen

Der PC (Host) muß angesteckte Geräte erkennen können. Ob ein Gerät angesteckt wurde, und ob es langsame oder schnelle Datenübertragung braucht, erfährt der PC vom Verteiler (Hub).

Um was für ein Gerät es sich handelt, kann der PC erfragen. Dafür sind im USB-Gerät Beschreibungen (*descriptor*) gespeichert, welche abgefragt werden. Es gibt fünf Arten von Standardbeschreibungen, je nach Gerätetyp können weitere hinzukommen.

Die Beschreibungen sind hierarchisch strukturiert: Jedes Gerät hat eine Geräte Beschreibung (*Device Description*). Das Gerät kann mehrere Konfigurationsbeschreibungen haben (*Confiuration Description*), pro solcher mehrere Schnittstellen Beschreibung (*Interface Description*) und pro Schnittstellenbeschreibung mehrere Endpunktbeschreibungen (*Endpoint Description*).

### 8.1 Geräte Beschreibung

Zunächst wird die Gerätebeschreibung (*Device Descriptor*) betrachtet. Jedes USB-Gerät besitzt genau eine. Jede Beschreibung beginnt mit den Feldern Länge und Typ, welche beide 1 Byte groß sind.

| Länge in Bytes | Name | Beschreibung |
|---|---|---|
| 1 | *bLength* | Länge der Beschreibung in Byte |
| 1 | *bDescriptorType* | Typ der Beschreibung |
| 2 | *bcdUSB* | USB Version des Geräts |
| 1 | *bDeviceClass* | Geräteklasse |
| 1 | *bDeviceSubClass* | Geräteunterklasse |
| 1 | *bDeviceProtocoll* | Protokoll |
| 1 | *bMaxPaketSize0* | Puffergröße für *Endpoint* 0 |
| 2 | *idVendor* | Hersteller ID. Wird von USB Forum vergeben |
| 2 | *idProduct* | Produkt ID |
| 2 | *bcdProduct* | Versionsnummer des Produkts |
| 1 | *iManufactur* | Zeichenkette für Hersteller |
| 1 | *iProduct* | Zeichenkette für Produkt |
| 1 | *iSerialNumber* | Zeichenkette für Seriennnummer |
| 1 | *iNumConfigurations* | Anzahl möglicher Konfigurationen |

Alle Felder müssen ausgefüllt sein, mit Ausnahme der Zeichenkettenfelder, welche optional sind. Wenn Zeichenketten gespeichert werden, wird hier nur der Index für eine solche Zeichenkette abgelegt, oder 0 wenn keine Zeichenkette gespeichert wurde.

Interessant ist das letzte Feld: Anzahl möglicher Konfigurationen. Ein USB-Gerät kann eine Vielzahl von Konfigurationen anbieten, aber nur eine kann aktiviert werden.

## 8.2 Beschreibung der Konfiguration

Auch diese beginnt wie alle Beschreibungen mit Längen- und Typenfeld. Dazu kommt die Gesamtlänge aller zugehörigen Beschreibungen (*wTotalLenght*, 2 Bytes), die Anzahl der Interfaces dieser Beschreibung (*bNumInterfaces*, 1 Byte), die Nummer für diese Konfiguration (*bConfigurationValue*, 1 Byte) und ihr Name (*iConfiguration*, 1 Byte), Attribute wie Stromversorgung und Remote-Aktivierung (*bmAttributes*) und der Stromverbrauch für diese Konfiguration (*MaxPower*, 1 Byte).

Die Gesamtlänge aller zugehörigen Beschreibungen ist wichtig, weil die Beschreibung der Schnittstellen nicht einzeln abgefragt werden, sondern zusammen mit der Beschreibung der Konfiguration übertragen werden.

## 8.3 Beschreibung der Schnittstelle

Auch die Beschreibung der Schnittstelle (*Interface Description*) beginnt mit Längen- und Typenfeld, danach folgt die Nummer der Schnittstelle (*bInterfaceNumber*, 1 Byte). Der nächste Wert ist interessant: *bAlternateSetting* erlaubt die gleiche Schnittstelle in mehreren Varianten zu definieren. Das macht zum Beispiel Sinn für eine Videokamera: diese will ihre Daten in festen Abständen abgeben, aber eben nur wenn sie aufnimmt. Vielleicht werden auch mehrere Qualitätsstufen angeboten, die verschieden große Datenmengen produzieren.

Zusätzlich gibt es noch Felder für die Anzahl der Endpunkte (*bNumEndpoints*, 1 Byte), Klasse und Subklasse der Schnittstelle (*bInterfaceClass* und *bSubInterfaceClass*, je 1 Byte) und das Protokoll (*bInterfaceProtocol*, 1 Byte). Schließlich kann auch hier eine Zeichenkette mitgeliefert werden: *iInterface* liefert den Zeiger darauf (1 Byte).

Klasse, Subklasse und Protokoll werden entweder in der Konfiguration oder bei der Schnittstelle gesetzt. An der anderen Stelle muß der Wert 0 gesetzt sein.

### 8.4 Beschreibung der Endpunkte

Zuletzt gibt es noch die Beschreibung des Endpunkts (*Endpoint Descriptor*). Es gibt keine Beschreibung für den Endpunkt 0, weil jedes Gerät diesen *Control Endpoint* besitzt und die Puffergröße schon in der Konfiguration steht.

Nach Längen- und Typenfeld kommt hier die Endpunkt Adresse (*bEndpointAddresse*, 1 Byte) und Endpunkt Attribut (*bmAttribut*, 1 Byte). Die Transferrichtung wird in der Endpunktadresse kodiert: Bit 7 auf 0 bedeutet *OUT*, Bit 7 auf 1 bedeutet *IN*. Das Attribut bestimmt die Art des Endpunktes: *Control* (0), *Isochron* (1), *Bulk* (2) oder *Interrupt* (3). Dazu kommen zwei Bytes für die Puffergröße (*wMaxPaketSize*) und 1 Byte für das *Poll Intervall* (*bIntervall*):

Bei *isochronen* und *interrupt*-Endpunkten soll der PC (*Host*) in regelmässigen Abständen den Endpunkt abfragen. Diese Intervalldauer (in Millisekunden) wird oft auf die nächste Zweierpotenz abgerundet und nicht exakt befolgt.

### 8.5 Zeichenketten

Auch Zeichenketten werden als Beschreibung abgelegt. Der Aufbau beginnt wie immer mit Längen- und Typenfeld. Danach folgt die Zeichenkette im 16-Bit-Unicode-Format (UTF16).

## 9 USB Anfragen (Requests)

Um Beschreibungen abzufragen und Funktionen zu konfigureren, gibt es eine Reihe von Anfragen, welche der PC an den Endpunkt 0 (*Control Pipe*) stellen kann. Allen Anfragen gemein ist das Datenformat: 8 Bytes lang: je ein Byte für den Typ der Anfrage und die Nummer der Anfrage, je zwei Bytes für Wert, Index und Länge. Der Typ der Anfrage ist Bitkodiert gemäß Tabelle 3.

| Bit-Position | Bedeutung |
|---|---|
| Bit 7 | Datentransfer zum Gerät (0) der vom Gerät (1) |
| Bit 6,5 | Request ist standard (00), klassenspezifisch (01) oder herrstellerspezifisch (10) |
| Bit 4..0 | Empfänger ist Device (00000), Interface (00001), Endpoint (00010) oder Speziell (00011). |

Tabelle 3: Kodierung des Anfragetyp

### 9.1 Status Abfragen

Status abfragen (*GetStatus Request*) muß immer unterstützt werden. Länge ist immer 2, Index ist die *Interface* Nummer bei Abfrage des *Interface*, bzw. die Endpunkt Nummer bei Abfrage des Endpunktes.

Das Gerät liefert ein Bitfeld zurück: versorgt sich das Gerät zur Zeit selbst mit Strom und ist *Remote Wakeup* aktiviert (sofern unterstützt). Die restlichen Bits sind reserviert. Das *Interface* liefert immer 0 zurück. Der Endpunkt liefert ebenso ein Bitfeld zurück. Hier ist nur ein Bit vergeben, und zwar ob der Endpunkt auf *STALL* geschaltet wurde, oder nicht.

### 9.2 Setzen und Löschen von Eigenschaften

Eigenschaften können gesetzt werden (*SetFeature Request*) oder gelöscht werden (*ClearFeature Request*). Bisher sind nur zwei Eigenschaften definiert: *Remote Wakeup* für das Gerät und *STALL* für den Endpunkt. Als Antwort wird ein Datenpaket der Länge 0 geschickt.

### 9.3 Adresse setzen

Nach dem Anschalten oder Ausführen eines Resets besitzt jedes Gerät die Adresse 0. Mit dem Kommando *SetAddress Request* kann die Adresse geändert werden. Als Antwort wird ein Datenpaket der Länge 0 geschickt.

### 9.4 Beschreibung abfragen

Mit einem *GetDescription Request* werden Beschreibungen des Gerätes (*Device Description*), der Konfiguration (*Configuration Description*) oder Zeichenketten (*String Description*) gelesen. Die Beschreibungen für Schnittstellen (*Interface Description*) und Endpunkte (*Endpoint Description*) werden mit der Konfiguration mitgeliefert und können nicht direkt gelesen werden.

Bei Zeichenketten kann über das *wIndex* Feld eine Sprache ausgewählt werden, sofern das Geräte die Zeichenkette in mehreren Sprachen bereitstellt. Über das Feld *wLength* wird eine maximale Länge angegeben. Längere Beschreibungen werden an dieser Länge abgeschnitten.

Der Puffer des *control endpoint* ist typischerweise nur 8 Byte lang. Weil Beschreibungen oft größer sind, müssen sie Stück für Stück ausgelesen werden: Mit IN-Befehlen werden die weiteren Stücke angefordert.

### 9.5 Schnittstelle setzen

Mit dem *SetInterface Request* wird zwischen den alternativen Modi der Schnittstelle gewählt.

## 10 Initialisieren des USB

Zu Beginn schlafen alle Geräte und alle Ausgänge der Verteiler sind gesperrt. Danach werden Ebene für Ebene des Ausgangs abgefragt, ob ein Gerät vorhanden ist. Wenn ja, wird der Ausgang aktiviert, und das Gerät konfiguriert.

Der *Host* kann den ganzen Bus zurücksetzen (*Reset*), indem beide Datenleitungen auf Masse gelegt werden. Danach müssen alle Geräte neu initialisiert werden.

## 11 Zusammenfassung

Der Universal Serial Bus (USB) wird sich durchsetzen, weil es sich um eine sinnvolle Neuordnung der PC-Peripherieanbindung handelt. Allerdings erfordert dies einen gewissen Zeitraum, in welchem sich Hard- und Softewarehersteller an die neue Technologie anpassen können. Heute werden nur wenige Tastaturen oder Mäuse als USB-Version verkauft. Erst langsam tauchen Drucker, Scanner und andere Peripheriegeräte mit USB-Anschluß auf dem Markt auf. Nur

die Firma Apple hat den Umstieg konkret umgesetzt: alle aktuellen Modelle kommen ausschließlich mit der USB in die Ladenregale.

USB ist kein Wunderwerk – schon lange mögliche Technik wurde endlich umgesetzt, um den Anschluß von Geräten am PC zu vereinfachen. Doch die USB-Technik ist nun knapp 3 Jahre alt, und während sich USB 1.0 und 1.1 noch nicht durchgesetzt haben, geht die Entwicklung weiter. Mit USB 2.0 soll ein noch schnellerer Nachfolger kommen, mit Firewire gibt es bereits eine ernstzunehmende Konkurrenz.

Neue Schnittstellen und damit reichlich Arbeit wie Steckkarten und Treiber installieren wird weiterhin nötig sein. USB hilft aber, weil alte Schnittstellen überflüssig werden. Zumindest bisherige Standardgeräte wie Maus, Tastatur, Scanner, Drucker, Modem, Joystick, Lautsprecher können mit USB auskommen und sind damit einfacher zu benutzen. Videokameras und externe Laufwerke stellen bereits heute hohe Anforderungen an die Übertragungskapazität und geben sich nur selten mit USB zufrieden.

Konkurrenz gibt es auch von anderer Seite. Während USB die Verkabelung einfacher macht, verdrängen andere Ansätze die Verkabelung vollständig durch Funk- oder Infrarottechnik (Beispiele: IrDA und Bluetooth).

Für den normalen Anwender ist USB vorerst die sinnvolle Anschlußvariante. Zumindest, wenn die PC-Hersteller USB in breitem Maße einsetzen.

## Literatur


[Comp98] Compaq, Intel, Microsoft, NEC. *Universal Serial Bus Specification 1.1*, September 1998.

[Klei99] Dipl. Ing. Hans Joachim Klein (Hrsg.). *USB – Universal Serial Bus.* Franzis Verlag. 1999.

[Schn97] Georg Schnurer. Die Schatten kommen. *c't – magazin für computer technik*, 2 1997, S. 292 – 298.

[Schn98] Georg Schnurer. USB – die Theorie. *c't – magazin für computer technik*, 1 1998, S. 77.

[Sier98] Peter Siering. Schubladen-Ware. *c't – magazin für computer technik*, 1 1998, S. 74 – 76.

[Stra95] Hermann Strass. Alles an einem Strang. *c't – magazin für computer technik*, 11 1995, S. 360 – 364.

[Torv99] Linus Torvalds. *Linux Kernel Source*, 1999.

# IrDA – Der Standard für Infrarotkommunikation

Urs Jetter

**Kurzfassung**

Für die Kommunikation mit mobilen Endgeräten ist eine berührungsfreie Schnittstelle optimal. IrDA ist der Kommunikationsstandard für Infrarotverbindungen. Die Arbeit vertieft die Protokollteile IrLAP (Link Access Protocol), IrMP (Link Management Protocol) und IAS (Information Acces Service) aus der Transportschicht. Es wird die Funktion und der Ablauf der Protokolle behandelt. Auch das Zusammenspiel der Protokolle wird betrachtet.

## 1 Einleitung

Der Standard für die berührungsfreie mobile Kommunikation mit Licht aus dem infraroten Spektralbereich wurde von der Infrared Data Association (IrDA) eingeführt. In dieser 1993 gegründeten Gruppe sind mehr als 150 Unternehmen organisiert. Die aktuelle Version heißt IrDA 1.1. In der Entwicklung befindet sich gerade die Version 2.0, die auch IrLAN enthält, das die Anbindung an ein LAN mit infrarotem Licht regelt. Seit Oktober 1995 liegt die Version 1.1 vor, die mehr Möglichkeiten bietet, als die Vorgängerversion [IrDA96]:

- Datenübertragungsraten von bis zu 4Mbit/s
- geringer Stromverbrauch
- geringe Kosten
- gerichtete Punkt-zu-Punkt-Kommunikation
- hohe Störungsunempfindlichkeit

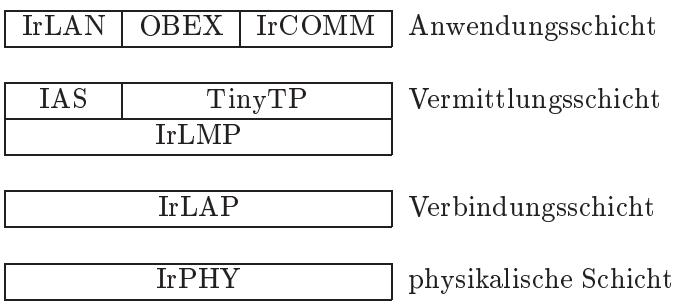


Tabelle 1: IrDA Schichtmodell

## 1.1 IrDA Schichtmodell

## 1.2 Anwendungen

Mit der fortschreitenden Verbreitung von Notebooks mußten die Anwender auch zunehmend Daten zwischen verschiedenen Rechnern abgleichen. Berührende Verfahren wie Kabelverbindungen oder Dockingstations[1] haben den Nachteil, daß die mechanische Steckverbindung verschleißt oder die Dockingsatation nicht dort ist, wo gerade ein Abgleich nötig ist.

Berührungsfreie Übertragungsverfahren wie Funkverbindungen, GSM-Telefonverbindungen oder Infrarotverbindungen haben den Charme, nicht nur günstig sondern auch am Einsatzort verfügbar zu sein.

Mittlerweile haben Drucker, Mobiltelefone, Handheld PC und Notebooks beinahe standardmäßig eine Infrarotschnittstelle. Bei der Kommunikation reicht häufig die physikalische Anwesenheit zum Zugriff aus. Beim Abgleichen von Daten wird manchmal noch eine Paßwortabfrage zwischengeschaltet.

IrDA Geräte lernen sich sofosrt kennen, wenn sie miteinander kommunizieren können. diese Funktion heißt Sniffing.

## 1.3 Notwendige Protokolle

Von den oben gezeigten Protokollen sind folgende für eine korrekte Kommunikation notwendig. Genaue Informationen zur Reduzierung des Codevolumens und der minimalen Unterstützung enthält IrDA Lite (siehe Abschnitt 2.7).

*IrPHY:* (Physikalische Verbindungsschicht) definiert physikalische Signale, Datendarstellung und mehr (siehe Abschnitt 2.1).

*IrLAP:* (Link Access Protocol) verantwortet eine einfache, gesicherte[2] Verbindung

*IrLMP:* (Link Management Protocol) Multiplex Datenübertragung und Adressmanagement mit IAS.

*IAS:* (Information Access Service) Stellt fest, ob bestimmte Dienste verfügbar sind und vergibt ggf. eine Adresse[3].

## 1.4 Weitere Protokolle

Ein Gerät, das den vollen Umfang an Funktionen anbieten soll (z.B. ein Computer oder auch eine Kamera[4]), muß noch folgende Protokolle für Anwendungen unterstützen:

*TinyTP:* (Tiny Transport Protocol) Stellt kanalweise Flußkontrolle zur Verfügung. Diese Funktionalitätist oft erforderlich.

*IrOBEX:* (Object Exchange Protocol) Stellt Anwendungen eine einfache, serialisierte Übertragung von Objekten zur Verfügung.

[1] Eine Dockingsation ist ein Einschubfach für Notebooks. Damit wird das Notebook um Steckkartenplätze erweitert und mit Peripheriegeräten verbunden.

[2] Die Verbindung ist bei multiplexer Übertragung nicht gesichert. Mehr dazu siehe Abschnitt 2.4

[3] Der IAS kennt nur 128 Adressen, die dynamisch vergeben werden. Die Entwickler gingen davon aus, daß nicht mehr als 128 Geräte für einen IAS sichtbar sind, was man auch leicht nachvollziehen kann.

[4] Solche Geräte, die als master an der Kommunikation teilnehmen (Abschnitt 3).

*IrCOMM:* (Emulation von Schnittstellen) Stellt älteren Programmen einen alternativen Übertragungsweg zur Verfügung. Dabei wird für sie eine serielle oder parallele Schnittstelle emuliert.

*IrLAN:* (Local Area Network access) Stellt die Basis für den direkten Anschluß an ein LAN. Dieser Standard ist noch nicht verabschiedet.

# 2 Funktionen der Protokolle

## 2.1 IrPHY – die physikalische Übertragungsschicht

Die physikalische Schicht enthält die dem Protokoll zugrundeliegenden Übertragungsmechanismen. Diese Schicht ist ganz oder teilweise in die Hardware integriert und somit am günstigsten von Chipherstellern zu erwerben. Oft wird der Hardwareteil von der sich schnell ändernden Software-Umsetzung des Protokolls getrennt. Der Softwareteil wird auch framer genannt, weil er die Umsetzung und Kontrolle der Übertragungsrahmen übernimmt. Die Spezifikation bestimmt folgende Parameter:

- optische Übertragung
- physikalische Form der infraroten Signale
- Kodierung von Bits, BOF, EOF, CRC etc.

IrPHY wird im Dokument *Serial Infrared Physical Layer Specification* der IrDA näher behandelt und ist nicht Thema dieser Ausarbeitung [Tajn98].

## 2.2 IrLAP – Schicht 2

Das Link Access Protocol liegt oberhalb der physikalischen Schicht (IrPHY) und entspricht der OSI Schicht 2. Es ist stark an die Übertragungsprotokolle High Level Data Link Control (HDLC) und Synchronous Data Link Control (SDLC) angelehnt. Sie werden um Eigenschaften für die berührungsfreie infrarote Kommunikation ergänzt.

IrLAP stellt eine zuverlässige Verbindung her. Die dabei benutzen Techniken werden in Abschnitt 3 beschrieben. Das IrLAP-Protokoll ist Bestandteil dieser Seminararbeit. Die benutzten Protokollmechanismen sind im einzelnen:

- wiederholte Paketübertragung (ARQ – Automatic Requeat Repeat)
- Flußkontrolle[5]
- Fehlererkennung
- Verbindungsaufbau und Verhandlung der Übertragungsparametern[6]

Durch die Sicherung der Übertragung auf sehr niedriger Ebene wird den übergeordneten Schichten zugesichert, daß die Daten ausgeliefert werden – oder zumindest eine Nachricht über den Ausfall erfolgt.

Diese Funktionen werden im Abschnitt 3 behandelt.

[5] IrLMP stellt eine multiplexe Flußkontrolle zur Verfügung, die in der Praxis verwendet wird.
[6] Puffergröße, Geschwindigkeit, Framegröße, etc.

## 2.3 IrLMP / IrIAS – Schicht 3

IrLMP ist der Schicht IrLAP übergeordnet und hängt von der ausgehandelten Übertragungsrate und Verbindungssicherheit ab. Es handelt sich hierbei um eine Schicht, die für den Verbindungsaufbau auf jeden Fall benötigt wird. Das IrLMP Protokoll stellt folgende Funktionalität zur Verfügung:

*Multiplexing:* mehrere virtuelle Verbindungen können über eine Verbindung laufen. IrLMP stellt den übergeordneten Schichten diese Funktionalität zur Verfügung.

*Management von Adresskonflikten:* Verfahren zur Vergabe von dynamischen Adressen.

*IAS:* (Information Access Service) Ein Dienstverzeichnis, das über das Vorhandensein von Diensten informiert und ggf. dynamische Adressen vergibt.

IAS stellt die Verbindung zwischen gewünschten und vorhandenen Diensten dar. In einer Tabelle werden die vorhandenen Dienste und Ausprägungen vorgehalten und mit den Wünschen der anfragenden Station verglichen.

IrLMP und IAS werden im Abschnitt 4 ausführlich beschrieben.

## 2.4 TinyTP – Übertragungsschicht

Das Tiny Transport Protocol (TinyTP, TTP) ist zwar eine wahlfreie Implementierung, jedoch ist es für multiplexe Datenübertragung sehr empfehlenswert. Das liegt daran, daß die Implementierung der Flußkontrolle im IrLAP vorsieht, daß nur eine von mehreren LMP-Verbindungen kontrolliert werden kann. Das hat zur Folge, daß jede weitere Verbindung ungesichert geschieht. TinyTP hilft hier durch eine weitere Flußkontrolle auf höherer Ebene ab. Es ist also anzunehmen, daß sich Programmierer von Anwendungen auf das Vorhandensein von TinyTP verlassen werden.

Der Flußkontrolle liegt ein kreditbasiertes System zugrunde. Die Höhe des Kredits hängt von der Puffergröße des jeweiligen Empfängers ab und wird beim Verbindungsaufbau verhandelt. Die Empfangsbestätigung wird im allgemeinen mit einem Datenpaket versendet.

Mehr über TinyTP kann in *TinyTP: A Flow-Control Mechanism for use with IrLMP* [Will96] nachgelesen werden.

## 2.5 IrCOMM – Schnittstellenemulation

Die Übertragung per Infrarot funktioniert grundlegend anders, als Kommunikation über serielle oder parallele Schnittstellen. Bei der Entwicklung von IrDA sollte jedoch weitverbreiteten Programmen[7], welche nur auf die serielle oder parallele Schnittstelle zugreifen, die Möglichkeit der Kommunikation ermöglicht werden. Folgende Übertragungstypen werden emuliert:

- Centronics-Emulation der parallelen Kommunikation über eine Centronicsschnittstelle (normales paralleles Kabel).
- serielle Kommunikation über ein neunadriges Kabel
- parallele und serielle Kommunikation über ein dreiadriges Kabel

[7] z.B. Laplink, Terminalprogramme etc.

Das Verwenden des IrCOMM-Pakets ist nicht unproblematisch. Neben der überflüssigen und damit doppelten Fehlerkorrektur erfolgt auch eine unnötige Serialisierung von Paketen, die sonst komplett übertragen würden. Die Kosten, die bei der Entwicklung gespart werden stehen einer ungenügenden Nutzung der Möglichkeiten von IrDA gegenüber.

## 2.6 OBEX – Object Exchange

In den letzten Jahren hat sich die objektorientierte Programmierung immer mehr durchgesetzt. Deshalb sind auch die Anforderungen an eine transparente Übertragung von Objekten gestiegen. Gerade der Ansatz von Java in Verbindung mit Jini sieht genau eine solche Übertragung vor. IrDA bietet mit OBEX eine Schnittstelle an, welche Objekte von der Anwendungsschicht eines Rechners in die Anwendungsschicht des anderen überträgt.

- arbeitet mit IrDA, ist aber unabhängig von der Transportschicht
- ist in kleinen Systemen mit weniger als 1kb Quellkode realisierbar
- anpaßbar an eigene Bedürfnisse

## 2.7 IrDA Lite – Das absolute Minimum

Durch die voranschreitende Integration von IrDA auf Chips ist es mittlerweile möglich IrDA mit sehr kleinem Speicherbedarf (< 10kB Programmcode, ca. 500bytes Speicher) zu realisieren. Die IrDA Lite Spezifikation gibt verschiedene Hinweise, welche Teile von IrDA realisiert werden sollen, um bestimmte Anwendungsbereiche zu optimieren.

Bei diesem Ansatz wird den unterschiedlichen Anforderungen der Endgeräte entsprochen. Für eine Armbanduhr ist eine Kommunikation mit dem Standard von 9600 bps sicher ausreichend, während eine Kamera mit 4Mbit/s arbeiten wird, selbst aber keinen großen Puffer benötigt.

## 2.8 IrLAN – die Vision

Die IrDA-Organisation hat IrLAN noch nicht verabschiedet. Es besteht zur Zeit nur als Vorschlag (Draft). Nichtsdestotrotz gibt es bereits erste Implementierungen[8], die zum späteren Standard möglicherweise kompatibel sein werden. Es sind drei mögliche Anwendungsbereiche von IrLAN vorgesehen:

*direkte Verbindung zum LAN:* Das Gerät wird über einen IrLAN Adapter direkt an das bestehende LAN angeschlossen. Der LAN Adapter übernimmt die Anmeldung und die Verbindung zum Netz (Bridge). Mit Jini ist auch ein Einsatz als Dockingstation im zugangsbeschränkten Umfeld denkbar [9].

*indirekte Verbindung zum LAN:* Über einen zweiten Computer wird die Verbindung zum LAN hergestellt. Der Computer übernimmt die Funktionen des LAN Adapters.

*virtuelles LAN:* Die Verbindung zwischen zwei Computern wird wie ein weiteres LAN aufgefaßt. Dadurch ist die Kommunikation zwischen den Rechnern wie in einem normalen LAN möglich. Alle LAN-Anwendungen werden dadurch nutzbar.

[8] Ein für Microsoft und Hewlett-Packard typisches Verhalten

[9] IrDA kennt zZ noch keine Zugangskontrolle. Daher können alle physikalisch erreichbaren Geräte auch genutzt werden.

Mehr Informationen über IrLAN bietet das Dokument *LAN Access Extension for Link Management Protocol (IrLAN)* [AxOR97] der Infrared Data Association. Wie gesagt ist dieses Dokument nur ein Vorschlag für die spätere Implementierung in IrDA.

# 3 IrLAP – Infrared Link Access Protocol

Die Kommunikationsschicht, die direkt auf der physikalischen Schicht (IrPHY, 2.1) aufsetzt, heißt IrLAP (Infrared Link Access Protocol). IrLAP wird von der IrLMP-Schicht (4) zur Kommunikation benutzt.

Neben der Problembehandlung, die die Hardware nicht leisten kann, setzt IrLAP auch den Verbindungsaufbau und die Rollenverteilung zwischen zwei Kommunikationspartnern um.

Die Vorgaben der Infrarot-Hardware sind:

*Punkt zu Punkt Verbindung:* Eine Verbindung kann stets nur zwischen zwei Geräten hergestellt werden.

*Reichweite:* Die typische Infrarotverbindung ist im Bereich bis 1m Entfernung realisiert. In Extremfällen sind 10m möglich. Das bedeutet eine Empfangsfläche von maximal $25\text{cm}^2$, die proportional zur Entfernung ist.

*Halbduplex:* Die Datenübertragung geschieht halbduplex. Vollduplexübertragung kann nur simuliert werden. Dazu müssen beide Geräte über geeignete Hardware verfügen[10].

*schmaler Übertragungsstrahl:* Der Infrarotstrahl wird in einem Winkel von 15 Grad abgestrahlt. Dadurch wird eine zu schnelle Schwächung des Signals vermieden. Auch werden Störeinflüsse minimiert.

*Unsichtbare Teilnehmer:* Ein Gerät kann so positioniert sein, daß es nur die Kommunikationsdaten eines dritten Gerätes empfängt. Dann tritt das dritte Gerät als Repeater auf.

*Störstrahlung:* Durch Interferenz kann es zu Störungen in der Übertragung kommen. Störquellen sind Leuchtstoffröhren, die Sonne, Schwarzlicht oder andere Infrarot-Geräte.

*Kollission:* Auf Hardwareebene gibt es keine Kollisionskontrolle. Dies behandelt IrLAP

## 3.1 Funktion

Bei der Implementierung von IrDA muß schon entschieden werden, ob es sich bei dem Endgerät um eine primäre oder eine sekundäre Station[11] handelt. Meistens fällt die Entscheidung nicht schwer. Die meisten mobilen Geräte sind primäre Stationen.

Eine primäre Station übernimmt die Führungsrolle bei der Kommunikation. Sie löst die Kommunikation aus, setzt die Verbindungsdaten (nach Absprache mit der sekundären Station) fest, behandelt unbehebbare Fehler (meistens durch eine Fehlermeldung an den Benutzer) und organisiert die Kontrolle und den Datenfluß. Primäre Stationen können immer auch als sekundäre auftreten. Typische Anwendungsfälle sind Computer, HandheldPCs oder ein Gerät welches einen Service benutzen will.

[10] Das Standard-Zeitfenster von 500ms ist ungeeignet, eine vollduplexe Kommunikation zu simulieren. Es muß durch Verhandlung auf Werte um die 100ms reduziert werden. Daneben ist auch eine Geschwindigkeitsanpassung sinnvoll.

[11] Die Begriffpaare primary/secondary, master/slave, etc. werden in der Literatur synonym verwendet.

Sekundäre Geräte sind Drucker oder andere Einrichtungen, die Dienste zur Verfügung stellen, aber nie selbst per Infrartoschnittstelle eine Verbindung aufbauen werden. Ihre Funktion beschränkt sich auf das Beantworten der Kommunikation einer primären Station.

Vor dem Verbindungsaufbau befinden sich beide Geräte im Normal Disconnect Mode (NDM). In diesem Zustand wird lediglich überprüft, ob es Anfragen an die eigene Station gibt. Andere empfangene Pakete werden ignoriert. Treten andere Übertragungen auf, muß ein Timeout von mindestens 500ms eingehalten werden, bevor ein Signal gesendet wird. Da jede Verbindungsanfrage mit 9600bps, 8bit und keiner Parität aufgebaut werden muß, kann das Überwachen der Kommunikation sehr stromsparend durch eine Funktion auf dem Chip ausgeführt werden.

Solange keine Verbindung aufgebaut oder abschließend inklusive Übertragungsdaten verhandelt worden ist, bleibt ein Gerät im NDM. Erst Anschließend wechseln beide Geräte in den NRM (Normal Response Mode), in dem die höherliegenden Funktionen aktiviert werden und damit auch der Stromverbrauch ansteigt. In diesem Übertragungszustand werden die eigentlichen Kommunikationspakete versendet.

IrLAP stellt vier Services zur Verfügung:

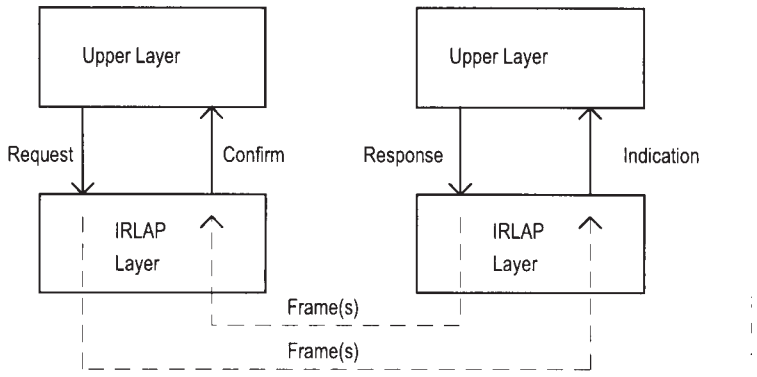


Abbildung 1: Kommunikationsweg

*Request:* Funktionsaufruf der übergeordneten Schicht an IrLAP.

*Confirm:* Meldung von IrLAP an die übergeornet Schicht als Folge einers Requests.

*Indication:* Anzeige eines Events oder einer von IrLAP ausgelösten Aktion.

*Response:* Antwort der übergeordneten Schicht an IrLAP als Folge einer Indication.

Einen Überblick über die angebotenen Dienste und deren Funktionen gibt Tabelle 2.

Unter Zuhilfenahme dieser Dienste kann die übergeordnete Schicht über die IrDA-Schnittstelle kommunizieren. Im allgemeinen wird diese Funktionalität von IrLMP genutzt.

## 3.2 Ausgewählte Techniken

Aus dem Angebot von IrLAP werden die Services Verbindungsaufbau, Übertragungsrahmen und Flußkontrolle näher beleuchtet, um die Kommunikation zwischen den Schichten zu erläutern.

| Verbindungslose Services | |
|---|---|
| Service | Funktion |
| Discovery | Erstellen einer Liste aller erreichbaren Geräte. |
| Adress Conflict | Verhindern von doppelten Geräteadressen im Kommunikationsbereich. |
| Unit Data | ungesicherter Datentransport an alle Geräte (max 384 byte) |
| **Verbindungsabhängige Services** | |
| Connect | Verbindungsaufbau |
| Sniffing | Erstellen einer stromsparenden Sniff-Verbindung |
| Data | wahlweise sichere sequentielle oder unsichere reihenfolgelose Übertragung von Daten. |
| Status | Anzeige ob der Verbinungsweg frei ist. |
| Reset | Zurücketzen von Zählern, Timern und Verwerfen von nichtbestätigten Datenpaketen. Findet nur bei beiderseitigem Einverständnis und nur für diese Verbindung statt. |
| Disconnection | Beendet die Verbindung und verwirft alle Informationen zur Verbindung ohne auf die Aktion der Gegenseite Rücksicht zu nehmen. |

Tabelle 2: IrLAP Dienstüberblick

### 3.2.1 Übertragungsrahmen

Ein Kommunikationsrahmen (Frame) im IrLAP-Umfeld hat je nach Übertragungsart den folgenden Aufbau:

| Asynchron 9600 – 115200 bps | | | | |
|---|---|---|---|---|
| n BOF | BOF | IrLAP Nutzbereich | FCS | EOF |

*n BOF:* n-faches Auftreten von C0h oder FFh. Im NDM ist n=10, im NRM wurde n bereits geschwindigkeitsabhängig ausgehandelt.

*BOF:* Begin of Frame einmaliges Auftreten von C0h.

*FCS:* 16 bit Checksumme nach CRC-CCITT über den Nutzdatenbereich.

*EOF:* End of Frame – Einmaliges Auftreten von C1h.

| Synchrone Übertragung 576 – 1152 kbps | | | | |
|---|---|---|---|---|
| STA | STA | IrLAP Nutzbereich | FCS | STO |

*STA* Start: 7Eh (01111110b)

*STO* Stop: 7Eh (01111110b) überprüfen!

*FCS* 16 bit Checksumme nach CRC-CCITT

Durch bit stuffing wird ein versehentliches Senden von 7Eh verhindert. Treten fünf Einsen in Folge auf, wird anschießend auf Senderseite eine Null eingefügt – auf der Empfängerseite wird sie wieder gelöscht.

| Modus | Event | Folgemodus |
|---|---|---|
| NDM | Connect-Request | Setup |
| | Empfang eines SNRM-Pakets | Connection |
| | sonst | NDM |
| Connection | Connect-Response | NRM(S) |
| | Disconnect-Request | NDM |
| | sonst | Connect |
| Setup | timeout | NDM |
| | Empfang SNRM-Paket | NRM(S) |
| | Empfang UA-Paket | NRM(P) |
| | Empfang disconnect-Kommando | NDM |
| | sonst | Setup |

Tabelle 3: Events beim Verbindungsaufbau

| 4 Mbps | | | | |
|---|---|---|---|---|
| 16 PA | STA | IrLAP Nutzbereich | FCS | STO |

*PA:* Pay Attention – Voraussenden von 16 bit : 1000 0000 1010 1000.

*STA:* Start: 32 bit : 0000 1100 0000 1100 0110 0000 0110 0000

*STO:* Stop 32 bit: 0000 1100 0000 1100 0000 0110 0000 0110

*FCS:* 32 bit Checksumme nach IEEE CRC 32

Durch das 4PPM Schema wird Codetransparenz erzielt.

### 3.2.2 Verbindungsaufbau

Vor dem Aufbau einer Verbindung befindet sich das Infrarotgerät im NDM (Normal Disconnect Mode). In diesem Modus sind nur Verbindungslose Dienste erreichbar. Voraussetzung für den Aufbau einer Verbindung ist das Vorhandensein einer gültigen Adresse.

Vom NDM aus kann durch Empfang eines Datenpakets mit Verbindungsaufforderung in den Connect-Modus oder durch einen Funktionsaufruf der übergeordneten Schicht in den Setup-Modus gewechselt werden. Sowohl der Connect- als auch der Setup-Modus stellen Übergangszustände dar, von denen aus nicht klar ist, ob der NRM (Normal Response Mode) oder wieder der NDM erreicht wird.

Vom Connect-Modus aus ist nur ein Sprung zum NRM als Secondary der Verbindung möglich. Nichtsdestotrotz können die Rollen später getauscht werden. Dieser Sprung geschieht nach dem Erhalten einer Response der übergeordneten Schicht. Alternativ kann die übergeordnete Schicht auch einen Rücksprung in den NDM durch ein Disconnect-Request verlangen.

Der Setup-Modus wird durch ein empfangenes Paket zum NRM verlassen. Dabei entscheidet die Art des Paketes, ob der NRM als Primary oder als Secondary angenommen wird. Der Setup-Modus wird nach Verstreichen eines Timeouts oder Erhalt einer disconnection-Aufforderung in den NDM verlassen.

Die einzelnen Aktionsmöglichkeiten zeigt folgende Tabelle 3:

# 4 IrLMP / IrIAS

Das Link Management Protocol (IrLMP) teilt sich in zwei Funktionseinheiten auf. Der Link Management Information Access Service (IAS) und der Link Management Multiplexer (LM-MUX) [SeWN96]. Dabei ist der Link Management Information Access Service auch für dem LM-MUX übergelagerte Schichten verfügbar.

## 4.1 Information Access Service (IAS)

Jeder IAS enthält Einträge von sich selbst und erkannten Geräten, so daß ein IrDA Gerät Informationen über die verfügbaren Services und die Geräte selbst erhalten kann. Die Informationen werden als Objekte in einer Datenbank gehalten. Die Informationen selbst werden nicht überprüft. Zusammen mit LM-MUX können auch Dienste auf demselben Endgerät angeboten werden.

## 4.2 Link Management Multiplexer (LM-MUX)

Der Link Management Multiplexer wird von IAS genauso wie von übergeordneten Transportschichten genutzt. Durch LM-MUX können mehrere Geräte miteinander kommunizieren – ohne eine direkte Sichtverbindung zu haben. Dabei werden wichtige Sicherungsdienste wie Datenflußkontrolle und Routing übernommen.

Dabei stellt der Multiplexer zwischen der IrLAP-Schicht und der übergeordneten Schicht (Transport-Dienst oder Anwendung) eine virtuelle Verbindung her, die der übergeordneten Schicht als gesicherte Punkt-zu-Punkt Verbindung erscheint. Die Flußkontrolle übernimmt dabei ebenfalls LM-MUX.

Die Nachfolgende Abbildung veranschaulicht die Verbindungsmöglichkeiten zwischen verschiedenen Endgeräten.

Folgende Verbindungswege stehen zur Verfügung:

*direkte:* Station A und B sowie Station A und C sind direkt verbunden. Alle zweipunktverbindungen können als direkt angesehen werden (X-P,X-I, etc.)

*indirekte:* B und C sind indirekt miteinander verbunden. Alle Verbindungen müssen über die Station A abgewickelt werden. P-Y-I, Q-Y-I beschreiben diese Möglichkeit.

*virtuelle:* Verbindungen innerhalb einer Station werden als virtuell bezeichnet. Die physikalisch gleiche Lokation ist transparent. Y-X, I-J sind Vertreter dieser Verbindungen.

*gemischte:* Gemischte Verbindungen entsprechen nicht dem optimalen Weg. Vielmehr wird versucht durch Schonen von Ressourcen zusätzliche externe Verbindungswege einzusparen. P-X-Y-I-J ist eine solche gemischte Verbindung. Sie ist äquivalent mit P-Y-I-J. Abhängig von der eigentlichen Implementierung und damit unbeeinflußbar wird hier ein Weg gewählt.

# 5 Ausblick und Schranken

Es hat sich gezeigt, daß die Euphorie, die IrDA entgegengebracht wird, in der Praxis schnell abebbt. Zum einen die hohe Störanfälligkeit bei größeren Entfernungen und die Notwendigkeit

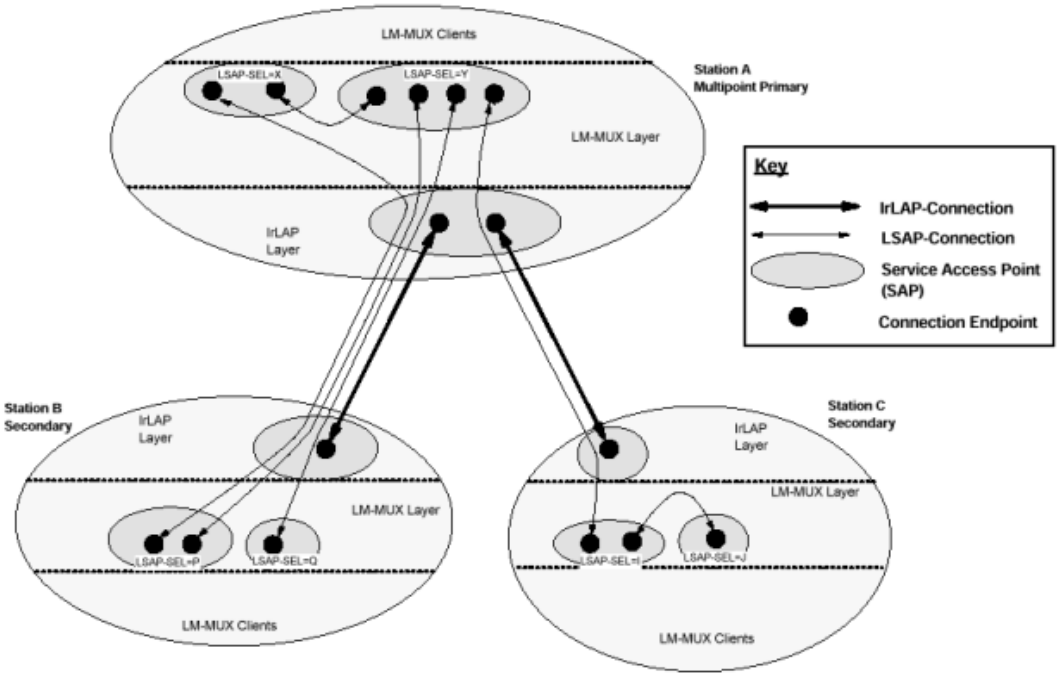


Abbildung 2: LM-MUX Verbindungswege

einer exakten Ausrichtung der Geräte setzt der Technik Grenzen. Anerkannt und eingesetzt wird heute der Gebrauch zum Abgleichen von Informationen zwischen Endgeräten. Für permanente Verbindungen (z.B. das verbinden von PDA über Handy ins Internet) sind solche Verbindungen nicht geeignet. Hier geht die Industrie wieder zu konventionellen Kabellösungen über. Der Abgleich der Telefonnummern findet jedoch immer mehr über IrDA statt.

Nach dem die Recherche für diese Seminararbeit abgeschlossen war, hat die IrDA am 26. Juni 1999 eine Reihe von weiteren Spezifikationen vorgestellt. Da die neuen Teile auf den bestehenden Sockel aufbauen und nur kleine Erweiterungen an den unteren Schichten vorgenommen wurden, ist keine Anpassung der Versionsnummer vorgenommen worden.

Neben kleinen Erweiterungen an den Standards IrPHY, IrLAP, IrLMP, TinyTP und IROBEX wurden folgende Protokollpakete neu vorgestellt bzw. IrLAN als Standard verabschiedet:

*IrLAN:* Anbindung von Geräten an ein bestehendes LAN

*IrDA Plug and Play:* Übergabe von Geräteinformationen an Betriebssysteme

*IrMC:* Standard für mobile Kommunikationsgeräte

*IrTrans-P:* Standard zur Bildübertragung zwischen Endgeräten

*IrDA Dongle Interface:* Schnittstelle für Schlüsselgeräte

*Jetsend over IrDA:* Implementierung des HP-Jetsend-Protokolls in IrDA-Geräte

Diese Aktivitäten zeigen, daß der Standard sich permanent verändert. Die IrDA-Organisation hat viel Mühe, den unterschiedlichen Interessen der Mitglieder gerecht zu werden. Microsoft will zum Beispiel seinen Standard IR 3.0 im nächsten Quartal vorstellen, aber darin auch Kompatibilität zum IrDA-Protokoll sicherstellen.

## Literatur


[AxOR97] Axtman, Ogus und Reilly. *LAN Acces Extensions for Link Access Management Protocol (IrLAN).* Infrared Data Organisation, http://www.irda.org, Draft 1.1. Auflage, January 1997.

[InHe99] Ingham und Helms. Infrared's Role in Wireless Communication Expands with IrDA. http://www.chips.ibm.com/products/infrared/news/article3/, 1999.

[IrDA96] IrDA. Technical Summary of 'IrDA DATA' and 'IrDA CONTROL'. *http://www.irda.org*, 1996.

[SeWN96] Seaborne, Williams und Novak. *Link Management Protocol.* Infrared Data Association, Version 1.1. Auflage, January 1996.

[Tajn98] Tajnai. *Serial Infrared Physical Layer Specification.* Infrared Data Association, Version 1.3. Auflage, October 1998.

[Tan96] Tan. IrDA Overview, 1996.

[WHNS+96] Williams, Hortensius, Novak, Smith, Suvak und Cremer. *Serial Infrared Link Access Protocol (IrLAP).* Infrared Data Association, Version 1.1. Auflage, June 1996.

[Will96] Suvak Williams. *TinyTP – A Flow-Control Mechanism for use with IrLMP.* Infrared Data Association, http://www.irda.org, 1.1. Auflage, October 1996.

# Anonymität und Vertraulichkeit im Internet

Roland Heinemann

**Kurzfassung**

Um nur den Inhalt einer Botschaft im Internet zu schützen, reicht es aus, diese mit den bekannten Kryptografieverfahren zu verschlüsseln. Will man jedoch auch das Vorhandensein einer Kommunikationsbeziehung zwischen Sender und Empfänger einer Nachricht oder das persönliche Bewegungsprofil beim Surfen im Internet vor anderen verbergen, dann reicht dies nicht aus. Auch das Verfahren, den ganzen Verkehr über einen einzigen Proxy zu leiten wie beim Anonymizer, kann durch die Kontrolle dieses Proxies durch einen Angreifer umgangen werden. Hier setzen die in diesem Seminarbeitrag vorgestellten Konzepte der Mix und der Crowd an und bieten durch ihre Verfahren dem Anwender die gewünschte Vertraulichkeit seines Benutzerverhaltens bis hin zur Anonymität. Anhand des Onion-Routing wird in diesem Seminarbeitrag eine Implementierung des Mix-Konzeptes vorgestellt.

## 1 Einleitung

Kommunikation im allgemeinen Sinn hat verschiedene Ausprägungen. Sie läßt sich nach Klassen kategorisieren, indem man betrachtet wer miteinander kommuniziert, über welches Thema geredet werden soll, mit welcher Dringlichkeit dieses Thema besprochen werden muß, auf welche Weise und mit welchen Sicherheitsvorkehrungen die Kommunikation stattfinden soll. Anhand dieser Betrachtungen wird jeder Kommunikationspartner den anderen und dessen Informationen anders beurteilen und dementsprechend einer anderen Dringlichkeitsstufe zuordnen. Auf Grund dieser Einteilung, werden an die Kommunikationsart unterschiedliche Ansprüche gestellt. Diese vielschichtige Thematik wird ausführlich in dem zweibändigen Werk [MüSt98] behandelt.

Der vorliegende Beitrag setzt sich mit zwei Ansätzen zur Bereitstellung von Vertraulichkeit und Anonymität in der Kommunikation und deren konkreter Umsetzung auseinander. Durch die Eigenschaft Vertraulichkeit wird der Inhalt einer Botschaft sowie das Vorhandensein einer Kommunikationsbeziehung zwischen Sender und Empfänger vor Dritten verborgen. Dabei stellt das Konzept der Crowds in Kapitel 3 Vertraulichkeit und Anonymität, das der Mix in Kapitel 4 noch zusätzlich Anonymität des Benutzers gegenüber seinem Kommunikationspartner zur Verfügung. Dadurch ist das Konzept der Mix komplexer und aufwendiger. Daß es dennoch möglich ist, akzeptable Verarbeitungsgeschwindigkeiten zu erreichen, zeigen die Angaben zum Onion-Routing in Abschnitt 5.2.

## 2 Grundlegendes

Obwohl die beiden Konzepte sich in ihren Ansätzen unterscheiden, bieten sie dem Anwender doch die Anonymität gegenüber dem Empfänger und Dritten sowie für den Empfänger die Nichtabstreitbarkeit von Nachrichten durch den Sender durch die in den folgenden Abschnitten beschriebenen Funktionen. Diese Funktionen wendet man auf eine Botschaft an, bevor man mit dieser wie in den Konzepten in Abschnitt 3 und 4 beschrieben verfährt.

## 2.1 Nichtabstreitbarkeit einer Botschaft

Manchmal, z.B. bei Verträgen oder anderen rechtlichen Angelegenheiten, ist es notwendig, den Absender einer Nachricht eindeutig identifizieren zu können. Dies kann durch eine Digitale Signatur erreicht werden. Dafür wird ein asymmetrisches Verschlüsselungsverfahren, z.B. RSA, verwendet. Nachdem der Sender ein Schlüsselpaar generiert hat, läßt er sich von einer vertrauenswürdigen Zertifizierungsstelle ein Zertifikat ausstellen, welches seinen öffentlichen Schlüssel sowie Angaben zu seiner Person enthält und von der Zertifizierungsstelle signiert ist; dieses Zertifikat kann in einem öffentlichen Verzeichnis hinterlegt werden. Dann verschlüsselt er die Botschaft $Y$ mit seinem eigenen geheimen Schlüssel $Inv(K_S)$. An diese Botschaft hängt er noch seinen öffentlichen Schlüssel $K_S$ an:

$$X = Inv(K_S)(C, Y), K_S$$

Das Anhängen von Teilen aneinander wird hier durch Kommas verdeutlicht; Klammern zeigen an, daß der Klammerinhalt als Argument einer Chiffrieroperation mit Schlüssel $K_i$ verwendet wird. $C$ ist eine zufällige Bitfolge, um das unerlaubte Entschlüsseln zu erschweren. Das Ergebnis $X$ dieser Operation verschickt der Sender an den Empfänger. Die Zugehörigkeit des mitgeschickten Schlüssels, der auch die Botschaft entschlüsselt, zum Sender kann der Empfänger durch den Vergleich mit dem vom Sender hinterlegten und durch die Signatur der Zertifizierungsstelle vor Manipulation geschützten Zertifikat überprüfen.

## 2.2 Anonymität

Manchmal ist es jedoch auch notwendig, gegenüber dem Empfänger anonym zu bleiben, wie z.B. bei Wahlen. Dies erreicht man, indem sich der Sender ein Pseudonym zulegt. Unter diesem tritt er nun auf. Falls auch die Nichtabstreitbarkeit einer Botschaft gewährleistet sein soll, müssen die Pseudonyme von einer Instanz zertifiziert werden, z.B. als zulässiger Wähler. Dann kann jeder die Stimmen zählen, ohne die Wähler identifizieren zu können. Ungültig sind dann die Stimmen, die denselben öffentlichen Schlüssel haben oder deren öffentlicher Schlüssel nicht in der Wählerliste steht. Da nur der zugelassene Wähler den geheimen Schlüssel besitzt, kann nur er eine signierte Stimme abgeben.

Seinen Schlüssel kann der Sender bei der betreffenden Instanz zertifizieren, indem er eine vertrauliche Botschaft nach dem Verfahren aus Abschnitt 4 schickt. In diese schreibt er die von der Instanz geforderten Angaben zu seiner Person und den öffentlichen Schlüssel, den er als Pseudonym verwenden will. Die Instanz kann diesen Antrag annehmen oder ablehnen. Die Antwort auf solch einen Antrag kann mit dem Verfahren der Rücksendeadresse (in Abschnitt 4.2 beschrieben) erfolgen. Nimmt die Instanz den Antrag an, so trägt sie den öffentlichen Schlüssel in ihre Liste der beglaubigten Personen ein. Diese kann von jedermann eingesehen werden. Falls aus rechtlichen Gründen die Person offengelegt werden muß, kann dies durch die persönlichen Daten, die bei der Instanz mit dem Antrag des Pseudonyms hinterlegt wurden, erfolgen. Auf Grund dieser Anforderungen können sich die Instanzen durch die erforderlichen Daten beim Antrag, sowie durch die Konditionen, unter denen die Identität offengelegt wird, und an wen diese Daten ausgehändigt werden dürfen, unterscheiden. Indem der Anwender eine ihm passende Instanz verwendet, lassen sich Anonymität, Vertraulichkeit und Nichtabstreitbarkeit individuell nach den eigenen Bedürfnissen skalieren.

## 2.3 Beobachter

Um die Verfahren besser beurteilen zu können, ist es sinnvoll, die möglichen Beobachter des Netzverkehres in drei Klassen einzuteilen.

*Lokaler Lauscher:* Kann als Angreifer die gesamte Kommunikation vom und zum Rechner des Nutzers überwachen. Zum Beispiel gehört der Netzwerkadministrator in diesen Bereich.

*Korrupte Mitglieder der in den vorgestellten Konzepten auftretenden Gruppen:* Diese können kooperieren und sich ihre erlangten Informationen teilen.

*Der Empfänger:* Bei Anfragen von Webseiten ist es der Server, der versucht, automatisiert Daten über das Benutzerverhalten zu ermitteln. Diese stellt der Betreiber des Servers gegebenenfalls benutzerspezifisch aufbereitet seinen Kunden zur Verfügung. Dafür kann er Cookies oder ein in Java, Javascript oder DirectX geschriebenes Programm benutzen.

Beobachtern dieser drei Klassen bieten sich unterschiedliche Möglichkeiten für eine Verkehrsanalyse[1], um den Aufenthaltsort und das Bewegungsprofil des Benutzers zu ermitteln. Dadurch ist die Anonymität des Senders gegenüber einem Beobachter von dessen Klasse abhängig.

# 3 Crowds

Hinter diesem einfachen Konzept [ReRu97] steckt die Idee, die Aktionen eines Einzelnen in den Aktionen einer größeren Gruppe zu verbergen. Durch die Bildung einer sogenannten Crowd soll die Verkehrsanalyse eines Mitglieds der Crowd durch einen Dritten oder ein Mitglied der Crowd vermieden werden. Dieses Verfahren wurde speziell für das Surfen im Netz konzipiert, um das Bewegungsprofil des Surfers in erster Linie vor dem Server zu verbergen, ohne dadurch eine größere zeitliche Verzögerung beim Surfen zu verursachen. Dies wird durch den geringen Overhead des Protokolls und die verwendete symmetrische Verschlüsselung erreicht.

## 3.1 Funktionsweise

Der Anwender wird durch einen Prozeß auf seinem Rechner repräsentiert. Dieser wird Jondo ("John Doe" gesprochen) genannt. Als eine Voraussetzung in der folgenden Überlegung wird angenommen, daß es durch die Struktur des Internets bedingt keinen globalen Beobachter geben kann, der alle im Netz kursierenden Nachrichten mithören und den Anwendern zuordnen könnte. Die Mitglieder einer Crowd sind in einer Liste eingetragen. Ein Anwender startet auf seinem Rechner einen Jondo-Prozeß. Diesen trägt der Anwender in seinem Browser als alleinigen Proxy ein, damit alle Anfragen des Browsers über diesen Proxy abgewickelt werden. Denn sonst gehen die durch einen Link auf einer Webseite gestarteten Downloads nicht über die Crowd und identifizieren den Benutzer beim Server.

Der Jondo tritt einer Crowd bei, indem er sich einen Account bei einer zentralen Stelle, genannt Blender, besorgt. Dafür übermittelt er diesem, gegebenenfalls verschlüsselt, seinen gewünschten Loginnamen und Paßwort, wie man in Bild 1 sehen kann. Der Blender trägt den neuen Jondo in seine Mitgliederliste ein. Daraufhin erzeugt der Blender für diesen Jondo und jedes Mitglied einen gemeinsamen Schlüssel für die symmetrische Verschlüsselung der Nachrichten. Mit Hilfe dieses Schlüssels kann der Jondo des Anwenders die Mitglieder der Crowd identifizieren und mit jedem einzelnen verschlüsselt kommunizieren. Der Blender schickt dem neuen Jondo eine Liste mit den IP-Adressen und den zugehörenden neu generierten Schlüsseln aller Mitglieder der Crowd. Außerdem schickt er jedem Mitglied die Adresse und den ihn betreffenden gemeinsamen Schlüssel des neuen Jondo. Auf diese Weise wird der neue Jondo den anderen Mitgliedern bekanntgemacht.

[1] Der vom Nachrichteninhalt unabhängige Versuch, zu ermitteln, wer, wann, mit wem, wieviel und wie häufig kommuniziert hat.

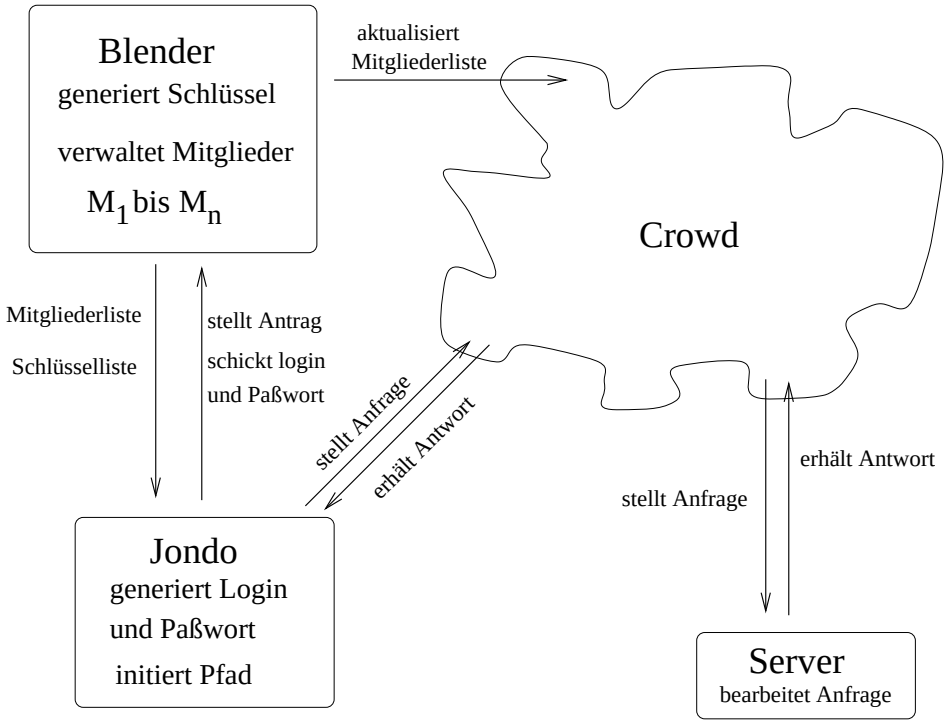


Abbildung 1: Das Konzept der Crowd

Daher führt jedes Mitglied der Crowd eine eigene Liste mit den IP-Adressen und einmaligen Schlüsseln für die Kommunikation mit den anderen Jondos. Diese Liste wird vom Jondo aktualisiert, wenn er Kenntnis vom Eintritt oder Verschwinden anderer Mitglieder erhält. Er kann auch selbständig Mitglieder aus seiner Liste entfernen, wenn er merkt, daß diese aus seiner Sicht die Nachrichten fehlerhaft weiterleiten. Daher können sich die Listen der Mitglieder zu jeder Zeit voneinander unterscheiden. Der Nachteil dieser Art der Mitgliederverwaltung einer Crowd ist, daß der Blender eine vertrauenswürdige dritte Person sein sollte. Da der Blender eine Liste aller $n$ Mitglieder und aller $n * (n-1)$ symmetrischen Schlüssel führt, kann jeder durch die Kontrolle des Blenders die abgefangenen Nachrichten zwischen den Mitgliedern lesen und protokollieren.

Nach der Anmeldung initiiert der Jondo einen zufälligen Pfad durch die Crowd, indem er seine Anfrage zusammen mit einer 128 Bit langen Pfadnummer an ein zufällig gewähltes Mitglied schickt. Dieses schickt die Anfrage dann nach dem Zufallsprinzip entweder über ein weiteres Mitglied oder direkt an den Server. Außerdem merkt es sich, von wem es diese Botschaft erhalten hat, an wen es sie weiterschickt, und welche zufällige 128 Bit große Pfadnummer der Botschaft von seinem Vorgänger zugeordnet wurde. Falls es jetzt ein weiteres mal eine Botschaft mit derselben Pfadnummer von einem Jondo, der nicht sein Nachfolger oder sein Vorgänger auf dem Pfad ist, erhält, so trägt es diesen Pfad in eine Pfadumsetzungstabelle ein. In dieser wird dem Pfad in Abhängigkeit vom vorangegangenen Jondo und der Pfadnummer der Nachricht eine neue Pfadnummer zugeordnet. Die Pfadnummer wird ebenfalls in der Nachricht geändert. Dies ist erforderlich, damit keine Endlosschleife entsteht, wenn dieser Jondo mehrmals auf dem Pfad enthalten ist. Dadurch wird bei einer wiederholten Übertragung über denselben Pfad die Schleife ausgelassen. Statistisch ist es sehr unwahrscheinlich, daß es mindestens zwei Pfade gibt, die sowohl einen gemeinsamen Pfadabschnitt als auch die gleiche Pfadnummer auf diesem Stück verwenden.

Optional kann der Verkehr auf einem Pfad mit einem Pfadschlüssel kodiert werden, damit kein Beobachter den Inhalt lesen kann. Diesen Pfadschlüssel ordnet der Jondo intern dem Pfad zu und reicht ihn an seinen Nachfolger weiter, falls er die Anfrage nicht direkt stellt. Da

er mit seinem Nachfolger einen gemeinsamen Schlüssel $K$ teilt, kann er den Pfadschlüssel $P_K$ mit dem gemeinsamen Schlüssel kodieren und dann an seinen Nachfolger schicken. Auf diese Weise erhält nur der Nachfolger den Schlüssel.

Nach diesem Verfahren werden statische bidirektionale Kommunikationskanäle aufgebaut. Dadurch wird der zeitliche Aufwand und das Datenvolumen im Netz, die durch den Aufbau eines neuen Pfades entstehen, eingespart, so lange die Pfade bestehen. Ein Pfad wird abgebrochen, wenn ein Jondo auf dem Pfad die Nachricht nicht weiterleitet oder nicht annimmt, weil er z.B. nicht mehr vorhanden ist. Durch das zugrundeliegende Protokoll TCP/IP kann ein sendender Jondo eines Datenpaketes feststellen, ob der Empfänger, der nächste Jondo oder der Server, die Nachricht angenommen hat. Der Pfad wird abgebrochen, indem der Jondo, der den Fehlschlag beim Versand erkannt hat, an seine Vorgänger auf dem Pfad eine Abbruchbotschaft schickt.

Wird ein neuer Jondo in die Crowd aufgenommen, ist er noch in keinem der bestehenden Pfade enthalten. Um seine Anfragen absetzen zu können, muß er einen neuen Pfad mit ihm als Ursprung initiieren. Damit der neuer Jondo nicht als Quelle einer Anforderung, die über einen neuen Pfad geschickt wird, erkannt werden kann, müssen alle Mitglieder die bestehenden Pfade vergessen. Dadurch erscheinen alle bei der einsetzenden Neuinitiierung der Pfade einander gleichwahrscheinlich als Ursprung der Pfade und der versendeten Nachrichten. Dies schützt den neuen Jondo. Damit nicht zu häufig eine solche Neuinitialisierung der Crowdpfade erfolgt, integriert der Blender neue Mitglieder periodisch in gewissen Zeitabständen.

Da über einen Jondo mehrere Pfade gehen, kann dieser sich einen beliebigen für seine Anforderungen aussuchen. Über den für den Versand verwendeten etablierten Pfad erreichen den Senderjondo die angeforderten Daten. Durch dieses Konzept steigt der Durchsatz und die Anonymität des Senders mit der Anzahl der Mitglieder und der dadurch größeren Anzahl an Pfaden. Die einzelnen Jondos einer Crowd sind dann über das Netz weiter verteilt. Und die Jondos können die Anfragen auf die bestehenden Pfade verteilen.

Die Echtzeitfähigkeit dieses Konzepts hängt jedoch allein davon ab, über wieviele Mitglieder die Nachricht und die Antwort geleitet werden, ehe sie am Bestimmungsort ankommen. Da der sendende Jondo auf die Pfadlänge sowie die Lebensdauer der Pfade keinen Einfluß hat, kann man nur eine statistisch ermittelte Antwortzeit abhängig von der Größe der Crowd angeben. Jeder Jondo hat aber die Möglichkeit, die sich für die über ihn führenden Pfade ergebende Verzögerung durch Testanfragen bei Servern auszutesten. Dadurch kann er den für seine Sicherheitsbedürfnisse kürzesten und geeignetsten Pfad für seine Anforderung ermitteln und nutzen. Diese Einteilung der Pfade hat aber nur solange Gültigkeit, bis die Pfade beim Eintritt eines neuen Mitglieds oder wegen eines erkannten Übertragungsfehlers abgebrochen werden.

## 3.2 Statistische Anonymität

Das vorgestellte Modell bietet dem Sender gegenüber den in Abschnitt 2.3 beschriebenen Beobachtern die in Tabelle 1 aufgeführte theoretische Sicherheit. In dieser Tabelle bedeutet $n$ Anzahl der Mitglieder einer Crowd, $p_f$ die Wahrscheinlichkeit, daß eine Anforderung von einem Jondo über ein weiteres Mitglied der Crowd statt direkt an den Empfänger geschickt wird. Diese Wahrscheinlichkeit $p_f$ sollte größer als $1/2$ sein, damit es für kooperierende Mitglieder schwieriger wird, den Ursprung eines Pfades auszumachen. Denn umso größer $p_f$ gewählt wird, umso länger sind die Pfade im Mittel und desto weniger Pfadjondos werden die kooperierenden Mitglieder statistisch gesehen auf diesem Pfad haben.

Durch das Konzept bedingt wird dem Sender vor einem lokalen Beobachter kein Schutz geboten, da alle ankommenden und abgehenden Nachrichten des Senders über Rechner versandt werden, die unter der Kontrolle dieses Beobachters stehen.

| Angreifer | Anonymität des Senders | Anonymität des Empfängers |
|---|---|---|
| lokaler Beobachter | keine Anonymität | $P$(Anonymität gewährleistet) $\longrightarrow 1$ (für n gegen unendlich) |
| c kooperierende Mitglieder $n \geq \frac{p_f}{p_f - \frac{1}{2}}(c+1)$ | Anonymität wahrscheinlich: $P$(absolute Anonymität) $\longrightarrow 1$ (für n gegen unendlich) | $P$(absolute Anonymität) $\longrightarrow 1$ (für n gegen unendlich) |
| Der Endserver | Anonymität gewährleistet | **N/A** |

Tabelle 1: Die Crowd bietet diese Formen von Anonymität gegenüber Angreifern

Ein lokaler Beobachter kann den Inhalt einer Nachricht nicht genau bestimmen, wenn die Nachrichten mit einem ihm nicht bekannten Pfadschlüssel kodiert sind. Denn er müßte schon die Botschaft entschlüsseln können, um die dekodiert gestellte Anfrage an einen Server mit dem kodierten Inhalt eines Pfades in Zusammenhang bringen zu können. Da der Server und ein Großteil der Pfadjondos nicht in seinem Blickfeld sind kann er nur die Adresse des Servers und den Inhalt lesen, wenn der Jondo die Anfrage direkt an den Server stellt oder der Beobachter selber einen Jondo auf dem Pfad hat. Umso größer die Anzahl der Mitglieder einer Crowd ist, desto größer ist die Wahrscheinlichkeit, daß sich mindestens ein einziger Jondo auf dem Pfad außerhalb des Blickfeldes des lokalen Beobachters befindet.

Dem Endserver bieten sich andere Möglichkeiten die Identität und die Adresse des Anwenders zu ermitteln, siehe Abschnitt 2.3.

Gegenüber $c$ kooperierenden Mitgliedern einer Crowd bietet das Konzept eine vom Anteil dieser an der gesamten Crowd abhängigen Sicherheit. Ist ihr Anteil gering, dann können sie sich nur wenig Informationen teilen und umgekehrt. Auch die Verlängerung des Pfades bringt hier keine zusätzliche Sicherheit, da jeder Jondo auf dem Pfad die über ihn versendeten Nachrichten dekodieren kann. Deshalb ist für sie die Adresse des Servers bekannt. Auch können sie den Inhalt einer Anfrage nach Belieben ändern. Durch die Kooperation zwischen korrupten Mitgliedern kann die Quelle einer Nachricht nur bis zum Vorgänger des ersten korrupten Jondos im Pfad zurückverfolgt werden. Je mehr Mitglieder die Korrupten in der Crowd stellen, desto sicherer ist dieser Vorgänger die Quelle. Alle anderen nichtkorrupten Jondos sind aus der Sicht der Korrupten mit gleicher Wahrscheinlichkeit sowohl der eigentliche als auch nicht der Ursprung der Nachricht. Für genauere Berechnungen der Vertraulichkeit des Senders in Bezug auf die Anzahl der korrupten Mitglieder, lese man in dem Bericht [ReRu97] nach.

# 4 Mixe

Das Konzept der Mixen [Chau81] entkoppelt bei der elektronischen Kommunikation die Zusammengehörigkeitbeziehung zwischen dem Sender einer Botschaft und dem Empfänger derselben, sowohl für den Empfänger wie auch für alle Pfadmitglieder und lokalen Beobachter. Darüberhinaus bietet es dem Benutzer verschiedene Grade von Anonymität durch die in Abschnitt 2.2 beschriebenen Verfahren.

## 4.1 Verfahren

Mit einer Kette von hintereinandergeschalteten Mixen kann man den Ursprung einer Botschaft $X$ verbergen. Dies erreicht der Sender, indem er seine Nachricht folgendermaßen *kodiert*. Er besorgt sich die öffentlichen Schlüssel $K_i$ und Adressen $A_i$ der Mixen $M_i$, über die

er die Botschaft leiten will, aus einer öffentlichen Tabelle. Um das Entschlüsseln mit Hilfe von Kryptoanalyse zu erschweren, wird vor die eigentliche Botschaft vor der Kodierung eine zufällige Bitfolge $R_0$ vorgeschaltet. Dieses Paket wird nun mit dem öffentlichen Schlüssel $K_E$ des Empfängers $E$ verschlüsselt. An das Ergebnis wird die Adresse $A_E$ des Empfängers gehängt. Vor dieses Paket wird wieder eine zufällige Bitfolge $R_n$ vorgeschaltet. Das Ganze wird jetzt mit dem öffentlichen Schlüssel der letzten Mix $M_n$ in der Kette verschlüsselt.

Das Ergebnis wird jetzt so verwendet, als ob die letzte Mix der Empfänger wäre. Also führt der Sender das Anhängen der Adresse der letzten Mix $A_n$, Vorschalten einer zufälligen Bitfolge $R_{n-1}$ und das Verschlüsseln mit dem öffentlichen Schlüssel $K_{n-1}$ der vorletzten Mix $M_{n-1}$ durch. Dies wiederholt er rekursiv bis er mit dem öffentlichen Schlüssel $K_1$ der ersten Mix $M_1$ in der Kette verschlüsselt hat. Dann verschickt er das Paket

$$K_1(R_1, K_2(R_2, \ldots K_{n-1}(R_{n-1}, K_n(R_n, K_E(R_0, X), A_E), A_n) \ldots, A_3), A_2)$$

an $M_1$. Das Anhängen von Teilen aneinander wird hier durch Kommas verdeutlicht. Klammern zeigen an, daß der Klammerinhalt als Argument einer Chiffrieroperation mit Schlüssel $K_i$ verwendet wird.

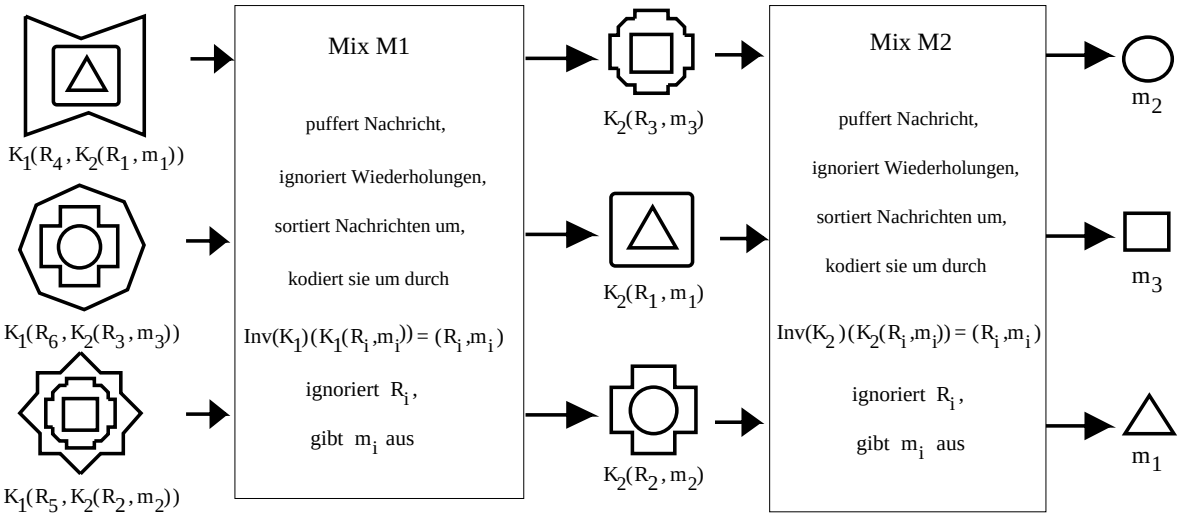


Abbildung 2: Weg einer Botschaft durch Mixkette

Erhält eine Mix $M_i$ eine Botschaft, so dekodiert sie diese mit ihrem geheimen Schlüssel $Inv(K_i)$, verwirft die zufällige Bitfolge $R_i$ und verschickt den Botschaftsteil an die Adresse $A_{i+1}$ bzw. $A_E$. Hinter dieser Adresse kann eine weitere Mix stehen, die wieder dieselbe Prozedur durchführt, siehe Bild 2. Ist es schon der Empfänger, so entschlüsselt dieser die Botschaft mit seinem geheimen Schlüssel $Inv(K_E)$ und liest sie dann. Weder die Mix noch ein Beobachter von außen kann entscheiden, ob der Adressat nur eine weitere Zwischenstation oder der Empfänger der Botschaft ist (siehe Abschnitt 2.2). Durch die asymmetrische Verschlüsselung ist gewährleistet, daß jede Mix nur ihren Vorgänger und ihren Nachfolger anhand deren Adressen, über die die verschlüsselte Botschaft geschickt werden soll, kennt. Nur der Empfänger ist in der Lage, ohne erheblichen Aufwand die Botschaft zu lesen.

Um jedoch zu verhindern, daß durch die Beobachtung der ein- und ausgehenden Botschaft deren Weg verfolgt werden kann (sog. Verkehrsanalyse) und mit Hilfe von Kryptoanalyse der geheime Schlüssel der Mix ermittelt werden kann, speichert die Mix die Botschaften in einem Puffer zwischen. Dabei entfernt sie Duplikate und an sie adressierte Nachrichten aus dem Stapel und ersetzt sie nach dem Dummyschema (beschrieben in Abschnitt 4.2.1). Wenn dieser Stapel voll ist werden die Botschaften in einer zufällig gewählten Reihenfolge versandt. Dies ist in Bild 2 veranschaulicht.

Die eingehenden Botschaften einer Mix $M_i$ sind nach dem bisher beschriebenen, um die zufällige Bitfolge $R_i$ länger als die versendeten. Dies ist wieder ein Ansatz, um die eingehenden Botschaften den abgehenden zuordnen und doch noch den Weg verfolgen zu können. Daher stellt man auch noch die Forderung, daß alle im Mixnetz kursierenden Nachrichten die gleiche Länge haben müssen. Da jede Mix ihren Nachfolger kennt, kann sie dessen öffentlichen Schlüssel verwenden, um die Nachrichten auf die geforderte Größe aufzublähen und mit diesem zu verschlüsseln. Dadurch wird diese Art der Zuordnung sehr erschwert. Der Aufwand steigt dadurch natürlich, da jede Mix nun jede Botschaft zweimal entschlüsseln und einmal verschlüsseln muß.

## 4.2 Nicht verfolgbare Rücksendeadresse

Manchmal ist es notwendig, auf eine anonym gestellte Anfrage eine Antwort oder eine Quittung zu erhalten, ohne dafür seine Identität oder seinen Aufenthaltsort preiszugeben. Das Konzept der Mix bietet hierfür einen weiteren Typ von Mixen an, die die Möglichkeit bieten, in der Botschaft eine nicht zurückverfolgbare Rücksendeadresse $A_X$ mitzuschicken. Um die Antwort zu verschlüsseln, wird ein Schlüsselpaar vom Sender generiert. Innerhalb der geschützten Botschaft werden der öffentliche Schlüssel $K_x$ und die Adresse $A_X$ mitgeschickt:

$$A_X, K_x \text{ mit } A_X = K_n(R_n, K_{n-1}(R_{n-1}, \ldots K_2(R_2, K_1(R_1, A_S), A_1) \ldots, A_{n-2}), A_{n-1})$$

$K_n$ ist der öffentliche Schlüssel der Mix $M_n$, der ersten Mix in der Rücksendekette vom Empfänger zum Sender. $K_i$ sind entsprechend die öffentlichen Schlüssel der übrigen Mixen, $A_i$ deren Adressen, und $A_S$ die Adresse des Senders der Anfrage. $R_1$ bis $R_n$ sind ebenfalls vom Sender in der Rücksendeadresse mitgeschickte Schlüssel, deren Inverse nur dem Sender bekannt sind. Damit chiffrieren die Mixen auf der Rücksendekette die Botschaft, um das Dekodieren für Dritte zu erschweren. Beachte: *Die Mixen in der Kette Sender-Empfänger und der Rücksendekette Empfänger-Sender müssen weder dieselben Mixen noch in der gleichen Reihenfolge oder derselben Anzahl sein. Auch darf eine Kette mehrmals über dieselbe Mix gehen, nur nicht direkt hintereinander.*

An die erste Mix $M_n$ sendet der Empfänger die mit dem öffentliche Schlüssel $K_x$ kodierte Antwort $P$, vor welche noch die Bitfolge $R_0$ gehängt wurde, zusammen mit der Rücksendeadresse $A_X$:

$$T = A_X, K_x(R_0, P)$$

Die erste Mix $M_n$ in der Rücksendekette versendet folgendes Paket:

$$K_{n-1}(R_{n-1}, \ldots K_2(R_2, K_1(R_1, A_S), A_1) \ldots, A_{n-2}), R_n(K_x(R_0, P))$$

Bei der Mix $M_1$ kommt an

$$K_1(R_1, A_S), R_2(R_3(R_4 \ldots (R_n(K_x(R_0, P)))))$$

An den Sender verschickt sie das Paket

$$R_1(R_2(R_3 \ldots K_x(R_0, P)))$$

Dieser dekodiert daraufhin seine Quittung bzw. Antwort zuerst mit dem geheimen Schlüssel zu $R_1$, dann rekursiv mit den anderen Schlüsseln $R_i$. Auf diese Art erhält der Sender seine Antwort.

### 4.2.1 Abwehr einer Verkehrsanalyse

Um einen Lauschangriff zu erschweren, schicken Mixen (z.B. in einem Ring, siehe Abschnitt 4.2.2) sich gegenseitig nicht nur ihre verschlüsselten Botschaften weiter. Für die Leerstellen in ihrem Ausgabepuffer generieren sie auch sogenannte Dummies, deren Botschaftsteil enthält, daß man sie als Dummy gefahrlos zerstören kann. Diese Dummies werden an eine zufällig gewählte Mix über einen zufällig gewählten Weg geschickt. Daher ist ein Dummy nur für die als Empfänger vorgesehene Mix von einer regulären Botschaft unterscheidbar. Diese kann nach Bedarf für den empfangenen Dummy einen neuen oder eine eigene Botschaft in ihren Ausgabepuffer ablegen.

Jede Mix sondiert die eingehenden Botschaften auf Dummies und an sie oder ein Mitglied ihres Subnetzes gerichtete Botschaften. Diese leitet sie an den Empfänger in ihrem Subnetz weiter. Korrupte Mixen könnten über eine Mix mehrere Botschaften hintereinander verschicken, um dadurch den Algorithmus für die Ausgabe der Nachrichten aus dem Puffer zu ermitteln. Mit diesem Wissen könnten sie doch noch die eingehenden den ausgehenden Botschaften einer Mix einander zuordnenden. Deshalb entfernt eine Mix gefundene Duplikate in ihrem Eingabepuffer und ersetzt diese. Die Mix versendet für diese freigewordenen Stellen eigene Botschaften oder generiert einen neuen Dummy, damit die Anzahl der eingegangenen gleich der versandten Botschaften ist. Dadurch kann die Mix nicht als Quelle einer Botschaft in dem abgehenden Strom von Botschaften ermittelt werden.

Die Botschaften haben durch die benötigten Routing-Angaben an der Startmix die größte Länge. Diese nimmt durch die Bearbeitung der Botschaften in den Mixen entlang der Kette zum Empfänger hin kontinuierlich ab. Mit dieser Information kann man zumindest durch den Vergleich der eingegangenen mit den versandten Nachrichten herausfinden, ob eine Nachricht an diese Mix gerichtet war. Wenn man jetzt noch fordert das alle kursierenden Botschaften gleiche Länge haben müssen, dann wird es für alle unmöglich, eine Verkehrsanalyse erfolgreich durchzuführen. Dies kann man dadurch erreichen, daß die fehlenden Bits einfach hinzugefügt werden. Damit niemand außer dem Adressaten die wahre Länge der Botschaft ermitteln kann, wird diese einfach noch einmal mit dem öffentlichen Schlüssel des Adressaten kodiert. Dieser muß deshalb die Botschaft zweimal entschlüsseln, ehe er an seine eigentliche Botschaft gelangt.

### 4.2.2 Ringe

Damit eine Mix nicht als Quelle einer Nachricht ausgemacht werden kann, fordert man, daß sie nur so viele Nachrichten versenden darf wie sie erhalten hat. Um eine konstante Anzahl von eingehenden Botschaften zu garantieren, können sich die Mixen zu Ringen zusammenschließen. Es ist natürlich sinnvoll, die Anzahl der Mixen für einen bestimmten Ring nach oben zu begrenzen, damit die statistische maximale Laufzeit einer Botschaft in diesem Ring ein erträgliches Maß nicht überschreitet. Eine Mix kann mehreren Ringen angehören und je nach Kapazitäts-, Sicherheitsbedürfnis und Verkehrsaufkommen einen oder mehrere der Ringe als Medium auswählen. Dies garantiert, daß die Mixen statistisch auch wirklich häufiger Nachrichten versenden können. Die einzige Ausnahme stellt der Beitritt in einen Ring dar. Die beitrittswillige Mix macht sich allen anderen durch eine an diese adressierte Botschaft bekannt. Dadurch werden in dem Ring von $n$ Mixen auch $n$ neue Platzhalter für Nachrichten geschaffen. Natürlich unterhält die Mix für jeden Ring einen eigenen Eingabe- und Ausgabestapel, sowie eine eigene Mitgliederliste. Für solche Gruppen gibt es viele dezentrale Lösungen des Mitgliedermanagements. Da es ausführliche Fachliteratur darüber gibt, sei an dieser Stelle auf diese verwiesen.

Wenn eine Mix Mitglied in einem Ring ist und sie eine Nachricht verschicken will, kann es vorkommen, daß sie trotz einer längeren Wartezeit keine Leerstelle im Datenpaketstrom findet.

Daher ist es einer Mix erlaubt, weiteren Ringen beizutreten oder einen neuen zu gründen. Wenn sie für die benötigten Ressourcen, Speicher und Rechenzeit, keine Kapazitäten offen hat, so kann sie sich aus einem der anderen verabschieden. Dafür schickt sie in beide Richtungen eine Abbruchbotschaft. Auch dieses Paket hat die vorgeschriebene Größe.

In so einem Ring von $n$ Mixen kursieren $n * (n-1)$ Botschaften inklusive den Dummies. Um die Dauerbelastung des Netzes durch solche Ringe gering zu halten, sollten sich nicht mehr als 20 Mixen zu einem Ring zusammenschließen. Wenn sich schon 20 Mixen zusammengeschlossen haben, schickt jede einer beitrittswilligen Mix eine Ablehnungsbotschaft. Dadurch verringert sich die Anzahl der kursierenden Botschaften um die $n$ Antragsbotschaften der beitrittswilligen Mix.

## 4.3 Ergänzungen

In den folgenden beiden Abschnitten wird darauf eingegangen wie man das Konzept echtzeitfähig machen und das Abbrechen einer Kette behandeln kann.

### 4.3.1 Echtzeitfähigkeit

Aufgrund der asymmetrischen Kodierung der Botschaft und deren Versand über verschiedene Routen, entsteht ein von der Länge einer solchen Versandkette, dem verwendeten Algorithmus und der Kommunikationsmenge abhängiger Zeitaufwand. Die daraus resultierende zeitliche Verzögerung kann reduziert werden, indem man einen Kommunikationskanal auf die in Abschnitt 4.1 beschriebene Art aufbaut und die eigentliche Kommunikation über diesen statischen Kanal bidirektional führt. Die Nachrichten brauchen dann nur noch mit symmetrischen Schlüsseln kodiert werden. Das genaue Verfahren zum Aufbau wird in Abschnitt 5 beschrieben. Die gewonnene Zeitersparnis hängt primär von der Kommunikationsmenge ab. Die Anzahl an Botschaften, ab der es sich lohnt, auf diese Weise über einen Kanal zu kommunizieren, hängt insbesondere vom Chiffrier-Algorithmus, der verwendeten Schlüssellänge sowie der Anzahl der beteiligten Mixen ab.

Da ein auf diese Weise eingerichteter Kanal eine feste Kette vorgibt, kann die Identität des Senders bei zu häufigen Nachrichtenwechseln über denselben Kanal durch einen Beobachter mit statistischen Mitteln wieder ermittelt werden. Dies kann man unterbinden, indem man eine Bandbreitenbegrenzung vornimmt. Ist die Anzahl der bisher über ein und denselben Kanal verschickten Botschaften über dieses Limit gestiegen, so wird dieser geschlossen. Die restlichen Botschaften versendet man dann über einen neu eingerichteten Kanal. Um diese Lösung echtzeitfähig zu machen, werden die wählbaren Parameter wie der Chiffrier-Algorithmus, die Länge der Versandkette oder der verwendeten Schlüssel sowie die Größe der Puffer und damit die maximale Verweildauer in einer Mix den Erfordernissen entsprechend klein gewählt. Dies bietet eine gegen Lauschen und Verkehrsanalyse resistente Umgebung.

### 4.3.2 Abbrechen einer Kette oder eines Übertragungskanals

Durch das Abbrechen einer Kette gelangen keine Nachrichten mehr vom Sender zum Empfänger oder umgekehrt. Ursachen für das Abbrechen können der Verlust von Paketen oder Mixen, der Überlauf der Eingangspuffer einer Mix in der Kette oder das Beenden des Kommunikationskanals durch einen der Endpunkte sein. Dies bemerkt die Mix, die sich in der Kette direkt vor der Störstelle befindet, da sie keine Empfangsbestätigung oder eine Zurückweisung durch das zugrundeliegende Übertragungsprotokoll im Netz innerhalb einer vorgegebenen Zeitspanne erhält.

Diese Mix schickt jetzt eine mit dem öffentlichen Schlüssel ihrer Vorgängerin, welcher ihr bekannt ist, verschlüsselte Abbruchbotschaft. Nachdem sie die Quittung durch das zugrundeliegende Protokoll für diese Abbruchbotschaft erhalten hat, streicht sie diesen Übertragungskanal aus ihrem Speicher. Auch dieses Paket hat die vorgeschriebene Größe. Dadurch kann es nur von der Vorgängerin nach dem Dekodieren von den eigentlichen Datenpaketen unterschieden werden. Diese Mix übermittelt wiederum ihrer Vorgängerin diese Abbruchbotschaft, mit deren öffentlichem Schlüssel verschlüsselt. Auf diese Weise erhält der Sender den Fehlschlag mitgeteilt und kann die Botschaft über eine neue Route verschicken.

Falls es sich um einen Kanal handelt, muß noch folgendes gelten: Jede Mix muß einen Kanal streichen dürfen, wenn über eine vorgegebene Zeitspanne keine Botschaft mehr verschickt wurde. Denn zum einen ist der Speicherbereich für die einen Kanal charakterisierenden Größen für jede Mix begrenzt und zum anderen kann es sein, daß der Kanal durch eine Mix vor ihr in der Kette unterbrochen wurde. Da jede Mix aber nur ihre direkte Nachfolgerin in der Kette kennt, kann sie eine Abbruchbotschaft für diesen Kanal nicht erreichen.

### 4.4 Integration des Konzepts in ein Mailsystem

In ein Mailsystem läßt sich das Konzept der Mixen sehr einfach integrieren. Dafür fordert man, daß jede Mail eine Kette von Mixen beliebiger Länge durchlaufen muß. Größere Mails werden nach der Kodierung mit dem öffentlichen Schlüssel des Empfängers in kleinere Happen zerstückelt. Diese werden wie separate Botschaften im weiteren Verfahren behandelt und über dieselbe oder über verschiedene Ketten von Mixen zum Empfänger geschickt. Entweder hat der Sender in seinem Mail-Client eine eigene Mix implementiert. Oder, wenn er Teilnehmer eines ihm vertrauenswürdigen Subnetzes oder Intranetzes ist, so kann die Mix-Instanz im Übergangsknoten ins Internet implementiert sein. Sie dient dann als Start-Mix beim Senden und Ende-Mix beim Empfangen in der Kette. Sie leitet die Mail für alle anderen Teilnehmer des Netzes und alle Arten von Beobachtern (siehe Abschnitt 2.3) bedingt durch das Konzept nicht nachvollziehbar weiter.

Die Sicherheit der Anonymität des Senders und die Schnelligkeit des Versandes kann skaliert werden. Durch das Konzept bedingt bedeutet eine sehr kurze Kette von Mixen den schnelle Versand einer Botschaft und unsichere Entkopplung des Senders vom Empfänger vor Dritten. Eine sehr sichere Entkopplung verursacht dementsprechend eine längere Kette von Mixen und eine größere Verzögerung. Der Versand einer sicheren Mail kann mit einer Mix automatisiert werden. Falls der Sender eine Antwort oder Quittung erwartet, generiert die Mix eine Rücksendeadresse (siehe Abschnitt 4.2) und schickt sie in der Mail mit. Dies und das Verschlüsseln nach Abschnitt 4 leistet die Mix für den Anwender transparent. Außerdem verhindert eine solche Mix die Verkehrsanalyse durch das in Abschnitt 4.2.1 vorgestellte Verfahren. Der Anwender/Sender braucht in seinem Mail-Client nur die Geschwindigkeit für den Versand seiner Mails zwischen "sehr schnell" und "sehr sicher", z.B. mit einem Schieberegler, zu wählen. Für die Mix bedeutet "sehr schnell" die Mail direkt an den Empfänger zu verschicken. Je mehr die Geschwindigkeit zu "sehr sicher" rückt, desto mehr Mixen werden in die Übertragungskette eingebaut. Als Standardwert wird wohl ein eher schnellerer von der Mix gewählt.

## 5 Onion-Routing

Für das Onion-Routing [Page99] wird das Konzept der echtzeitfähigen Mixen (siehe Abschnitt 4.3.1) verwendet. Diese implementierte Anwendung bietet bidirektionale Verbindungen. Dafür wird die aufwendigere asymmetrische Verschlüsselung mit den öffentlichen Schlüsseln nur zum Verbindungsaufbau verwendet. Da die Mixen im Internet unter verschiedenen Domains

stehen kann kein einzelner Onion-Router den Sinn des Onion-Routing untergraben. Da das Internet aus unabhängigen Netzen besteht und die Onion-Router über diese verteilt sein werden, müßten schon alle Mixen einer Kette kooperieren, um die Privatsphäre und vor allem den Aufenthaltsort des Senders durch eine Verkehrsanalyse bloßzulegen. Sobald es zwei nichtkorrumpierte Onion-Router gibt, den Startrouter und irgendeinen weiteren in der Kette, wird durch die Eigenschaften der Mix der Rückschluß sehr erschwert. In dieser Anwendung sind die im folgenden Onion-Router genannten Programmteile nur für das Weiterleiten der Botschaft zuständig. Daneben gibt es auch sogenannte Proxies. Diese bestehen aus drei logischen Schichten:

1. Einem optionalen anwendungsspezifischen privaten Filter, der den Datenstrom schützt. Er filtert die Personenangaben und die IP-Adresse des Anwenders aus den Nachrichten und Webanfragen heraus.

2. Einer anwendungsspezifischen Schicht, die den Datenstrom in eine vom Onion-Routing-Netz akzeptierte Form überführt. Dies beinhaltet das Aufteilen in Datenpakete der geforderten Größe und die Verschlüsselung mit dem Pfadschlüssel.

3. Einer Schicht, die die Verbindung aufbaut, die Datenpakete verschickt, empfängt und den Kanal abbaut, falls er nicht mehr benötigt wird.

## 5.1 Verlauf einer Datenübertragung

Eine Datenübertragung gliedert sich in drei Abschnitte:

1. Beim *Verbindungsaufbau* wird eine Onion-Datenstruktur initiiert. (Im weiteren wird sie einfach Onion genannt). Diese definiert den Verbindungspfad durch das Netz und stellt die Kontrollinformationen über das verwendete Chiffrierverfahren und den Schlüssel für die symmetrische Verschlüsselung, sowie die Adresse seines Nachfolgers jedem Knotenpunkt auf dem Transportweg im Onion bereit. Daraufhin wird dieser Onion mit dem öffentlichen Schlüssel des Knotenpunktes verschlüsselt. Da dies rekursiv die Kette abarbeitend für jeden Knotenpunkt wiederholt wird, kann jeder Knoten nur die für ihn bestimmten Daten lesen.

   Jeder Onion-Router entschlüsselt den erhaltenen Onion mit seinem privaten Schlüssel, merkt sich den Chiffrier-Algorithmus und den dazugehörigen symmetrischen Schlüssel in Zusammenhang mit den Adressen seines Vorgängers und seines Nachfolgers in der Kette. Dann bringt er den inneren Onion, der die Daten für seine Nachfolger enthält auf die richtige festgelegte Größe und schickt den oder die (falls er den Onion aufteilen mußte) Pakete an seinen Nachfolger in der Kette. Dies wird bei jedem Knoten wiederholt, bis der Endknoten des Kanals erreicht ist.

2. Beim *Datentransfer* dechiffriert jeder Onion-Router das erhaltene Datenpaket mit dem ihm für diesen Kanal übermittelten Algorithmus und dem symmetrischen Schlüssel und schickt das Ergebnis an seinen Nachfolger. Diese Datenpakete haben eine feste Größe, um die Verkehrsanalyse zu verhindern. Falls notwendig, werden die Datenpakete mit zusätzlichen Bits aufgefüllt, oder auf mehrere numerierte Pakete verteilt. Dies macht jeder Router in der Kette.

3. Der *Verbindungsabbau* kann sowohl von den Endpunkten als auch von einem dazwischenliegenden Router initiiert werden, falls dies notwendig werden sollte.

Auch hier kann ein Rückkanal über eine andere Kette aufgebaut werden, indem die in Abschnitt 4.2 erläuterte Rücksendeadresse beim Verbindungsaufbau mitgeschickt wird. Dafür erweitert man den Adreßteil um die Information über den für diese Antwort ausgesuchten Verschlüsselungsalgorithmus und den verwendeten symmetrischen Schlüssel. Dieser Rückkanal kann auch erst zeitlich versetzt aufgebaut werden, falls der Hinkanal durch einen Verbindungsabbau schon geschlossen wurde oder die Bearbeitung eine längere Zeit in Anspruch nimmt. Dafür wird die Rücksendeadresse vom Empfänger bis zur Benutzung zwischengespeichert. Durch die Verwendung der symmetrischen Kryptographie beim nachfolgenden Datenaustausch und der "fest verdrahteten" Kommunikationskanäle wird der Rechenaufwand auf ein erträgliches Maß reduziert, ohne dabei die Vorteile wie Nichtlokalisierbarkeit des Senders oder Anonymität und Vertraulichkeit der Kommunikation zu verlieren.

### 5.2 Daten zum Testlauf

Seit Juli 1997 läuft ein Netz aus Prototypen in den AT&T Labs. Dieses unterstützt mit seinen Proxies bisher folgende Protokolle: HTTP, FTP, SMTP, rlogin, telnet, NNTP, finger, whois und raw sockets. Anwendungen die diese Protokolle verwenden, können ohne Änderung über den Proxy den Onion-Routing-Dienst in Anspruch nehmen.

Das Prototypennetz lief auf einer Ultra2 mit zwei Prozessoren mit je 168 MHz. Das Netz umfaßt 9000 bis 9200 Proxies und Router. Mit diesem Netz wurden täglich mehr als 50000 Verbindungen bearbeitet, und am 31.12.1998 sogar der Spitzenwert von 84022 Verbindungen. Dies sind mehr als eine Million Web-Verbindungen im Monat über ein Backbone von nur 10MBit. In diesem Jahr wurde das Prototypennetz auf eine Ultra 450 mit vier Prozessoren mit je 300 MHz portiert. Diese wurde über eine 100 MBit Verbindung mit dem Internet verbunden. Bis heute wurden mehr als 15 Millionen Verbindungen über diesen Prototypen abgewickelt. Jeder, der Lust hat, kann selbst seine Verbindungen darüber laufen lassen. Wie das geht steht unter der URL in [Page99].

## 6 Zusammenfassung

Das Verfahren der Crowd nutzt die Netztopologie. Jedoch bietet es nur bedingte Entkoppelung von Sender und Empfänger. Gibt es zuviele korrupte Mitglieder, so ist die Anonymität nicht mehr gewährleistet. Gegenüber einem lokalen Beobachter bietet es keinerlei Schutz. Außerdem kann man keine festen Laufzeiten für die Nachrichten angeben, da die Pfade keine festlegbare Länge haben. Es ist aber einfacher zu handhaben.

Das Konzept der Mix ist gegen globale und lokale Beobachter resistent. Außerdem verbirgt es sowohl den Sender wie den Empfänger einer Kommunikationsbeziehung vor allen Beobachtern sehr effektiv. Allerdings ist es um einiges aufwendiger, da der Kanal von der sendenden Mix im vorhinein bestimmt und die Botschaft wie in Abschnitt 4.1 beschrieben kodiert wird. Dies bietet eine komplexitätstheoretische Sicherheit des Pfades und der übertragenen Daten.

Das Onion-Routing stellt dem Anwender transparent die Sicherheit des Mixkonzeptes zur Verfügung. Es ist wesentlich schneller, da es einen Kommunikationskanal nach dem Verfahren in Abschnitt 4.1 mit den öffentlichen Schlüsseln der asymmetrischen Kodierverfahren aufbaut und die Daten symmetrisch verschlüsselt über diesen überträgt. Durch die realisierten Proxies wird die Identität des Anwenders, die von den Anwendungsprogrammen in eine Nachricht oder eine Webseitenanfrage hineingeschrieben wurde, aus den Botschaften entfernt. Das Weiterleiten der Nachrichten wird in sogenannte Router ausgelagert. Dadurch werden die Kanäle unabhängig vom Anwender. Die Kanäle werden beständiger, da der Anwender offline gehen kann, ohne sie zu zerstören.

Es ist bei beiden anzumerken, daß der Anwender die Ausführung von Javascript, Java Applets, ActiveX deaktiviert haben sollte. Denn sonst kann ein Server sich durch ein kleines, in einer dieser Sprachen geschriebenes Programm die IP-Adresse des Senders übermitteln lassen.

Durch diese Konzepte wird der Schutz der Privatsphäre des Anwenders, z.B. eines Arbeitnehmers, vor den Beobachtern, z.B. seinen Mitarbeiter oder seinem Chef, gewährleistet. So wünschenswert die gewährte Anonymität und Vertraulichkeit sind, können sie für kriminelle Zwecke, wie z.B. Erpressungen und Mobbing, mißbraucht werden.

## Literatur


[Chau81] David L. Chaum. Untraceable Electronic Mail, Return Addresses and Digital Pseudonyms. *Communications of the ACM* 24(2), Februar 1981, S. 8.

[MüSt98] Günter Müller und Kurt-Hermann Stapf. *Mehrseitige Sicherheit in der Kommunikationstechnik.* Addison-Wessley. 1998.

[Page99] The Onion Routing Home Page. *http://www.onion-router.net*, 1999.

[ReRu97] Michael K. Reiter und Aviel D. Rubin. Crowds: Anonymity for Web Transactions. *DIMACS Technical Report* 97(15), August 1997, S. 18.

# Virtual Private Networks

Martin Treitz

**Kurzfassung**

Die Vernetzung von Computersystemen in Firmen und die damit verbundene Mehrfachnutzung von Resourcen wird heute immer bedeutender. So ist es heute üblich, dass jeder Mitarbeiter einen Computer an seinem Schreibtisch stehen hat und mit gemeinsamen Datenpools verbunden ist. Auch die firmeninterne Kommunikation über Rechnersysteme und computergestützte Teamarbeit werden immer üblicher. Bei der globalen Präsenz der Firmen stoßen eigene Netzstrukturen hier auf ihre Grenzen. Die Möglichkeiten, ein öffentliches Netz zu nutzen, um sein eigenes darauf zu simulieren, bietet die Virtual Private Network Technologie. Dabei spielen Aspekte der Sicherheitstechnologie, der Verschlüsselung und der Authentifizierung eine Schlüsselrolle. Auch werden Methoden zur Verfügung gestellt, die es erlauben, verschiedene Protokollbereiche zu verbinden.

## 1 Ein virtuelles Netzwerk mit realem Nutzen

### 1.1 Was ist überhaupt ein Virtual Private Network?

Obwohl der Begriff des *Virtual Private Network* (virtuelles privates Netzwerk, VPN) recht neu ist, bietet fast jeder Systemhersteller der Branche Komponenten und Software zum Aufbau eines *Virtual Private Network* (VPN) an. Ebenso wie das Angebot unterschiedlich ist, so viefältig sind auch die Ansichten, was man unter einem VPN verstehen kann.

Der fortschreitende Gebrauch und Nutzen von firmeneigenen Datenbanken, automatisierten Kontrollsystemen, e-commerce und computerunterstütztem Kundenservice ließ den Bedarf an der Vernetzung der Computer zur Konsistenz der Daten und der erleichterten Kommunikation 24 Stunden am Tag und 7 Tage die Woche stark ansteigen. Ließ sich dies in einzelnen Firmengebäuden und Produktionsstandorten noch recht leicht durch erprobte Netzwerktechnik verwirklichen, stieß man bei der heutigen globalen Präsenz der Firmen an die Grenzen herkömmlicher Netztechnik. Es ist nicht immer möglich und auch nicht unbedingt sinnvoll, ein eigenes privates Netz auf der Basis einer eigenen Infrastruktur aufzubauen. So ist es in den meisten Ländern verboten, auf öffentlichem Grund eigene Leitungen zu verlegen. Auch ist der Kostenaspekt von gemieteten Übersee-Standleitungen nicht zu verachten. Deshalb befindet sich der Markt für VPNs wohl am Beginn eines großen Wachstums.

Anstatt eine eigene statische Infrastruktur aufzubauen, bietet es sich vielmehr an, vorhandene öffentliche Kommunikationsnetze als Grundlage zu verwenden und darüber ein privates Netz zu simulieren. Im Idealfall sollten die Benutzer dabei nicht bemerken, dass das private Netz auf einem öffentlichen Netz basiert, das gleichzeitig auch von anderen genutzt wird. So stützt sich ein VPN auf ein öffentliches Netzwerk, das prinzipiell allen zugänglich ist.

Dies bietet sicherlich vielmehr Flexibilität, erhöhte Erreichbarkeit und in Zukunft wohl erleichtertes Netzwerkmanagement im Vergleich zu traditionellen Netzwerkstrukturen.

Die Nutzung vorhandener Resourcen bedeutet, dass das eigene Netz nicht wirklich privat ist, sondern nur privat scheint. Der Ausdruck privat bedeutet dabei, dass der Austausch von Daten über ein VPN in sicherer Art und Weise erfolgen muss [Brau99]. Die Informationsübertragung muss unverfälscht und unzugänglich für Nichtautorisierte erfolgen. Das heißt, dass der Empfänger davon ausgehen kann, dass die empfangenen Daten wirklich vom angegebenen Sender stammen und dass die Daten nicht durch Dritte erzeugt, gelesen oder verändert wurden. Und dies ist gleichzeitig auch der größte Flaschenhals derzeitiger VPN Technik.

Auch wenn die meisten Sicherheitssysteme sehr vertrauenswürdig erscheinen, ist momentan kein Produkt auf dem Markt, das sämtliche Bedenken ausräumt. Eine Studie geht davon aus, dass bis 2004 rund 72 Prozent des Sortware-VPN-Gesamtmarktes sich mit Sicherheitstechnik befassen wird. Und das bei einem Gesamtvolumen des Softwaremarktes von VPNs von geschätzten 1,26 Mrd. US Dollar in 5 Jahren [Brau99].

So stehen heute Krypthographie, Authentifizierung und Zugangskontrolle im Mittelpunkt bei der Konstruktion eines VPN.

Aber nicht nur Sicherheitsrisiken, sondern auch fehlende Dienstgüte (Quality of Service,QoS) in den meisten öffentlichen Netzen ist noch ein Hindernis bei der Verbreitung und effizienten Nutzung von VPN. Unternehmen sind teilweise darauf angewiesen, bestimmte Übertragungsraten (zum Beispiel bei Echtzeitunterstützung von Videokonferenzen) garantiert zu bekommen.

## 1.2 Grundsätzliche Systemstruktur

Es existieren drei unterschiedliche VPN Topologien, die natürlich beliebig kombiniert und erweitert werden können.

Zum einen kann man sich vorstellen, den Firmenhauptsitz mit verschiedenen Zweigstellen zu verbinden (LAN-to-LAN), diese Variante wird meist "*Intranet VPN*" genannt. Desweiteren ist es häufig wünschenswert, dass sich reisende Mitarbeiter von außerhalb in das Netz einloggen, als säßen sie an ihrem Arbeitsplatz. Diese Variante wird oft als "*Remote Access VPN*" bezeichnet. Die dritte Grundstruktur, als "*Extranet VPN*" bekannt, bezieht die Geschäftspartner, Zulieferer und Kunden mit in das Netzwerk ein [Aven99].

- *Intranet VPN*

  Ein Verbund der Netzwerke der verschiedenen Standorte einer Firma zu einem Netwerk ist heutzutage oft sehr wichtig. Die VPN intranets seien hier als semi-permanente Verbindungen über öffentliche Netze (zum Beispiel das Internet) verstanden. Diese Netze tragen das geringste Risiko, da Firmen gewöhnlich ihren Zweigstellen trauen und damit Ziel- und Quelladresse bekannt und kontrollierbar sind. Trotzdem darf der Schutz des Firmennetztes nach außen mittels Schutzmaßnahmen (z.B. firewalls) nicht aus dem Blickfeld geraten. Um die beiden Standorte zu verbinden, muss man also nur den Transport zwischen den beiden Orten herstellen und sichern (tunneling). Das Hauptaugenmerk liegt hier sicher eher auf der Performance des Netzes. Gewöhnlich werden hier wohl hohe Datenraten benötigt und eine Interoperabilität der eventuell verschiedenen Systeme erwartet. Durch Kostendegression in Netwerktopologien verschiebt sich immer mehr die Entscheidungskompetenz in kleinere Bereiche. Von daher kann es vorkommen, dass der Controlling-Bereich eines Unternehmens ein Ethernet hat und der Sales-Bereich einen FDDI-Ring.

- *Remote Access VPN*

  Firmen tendieren immer mehr dazu, das Internet als Backbonenetz für ihre reisenden Mitarbeiter zu nutzen. Die Unterhaltung von teuren Modem-Pools und die Berechnung

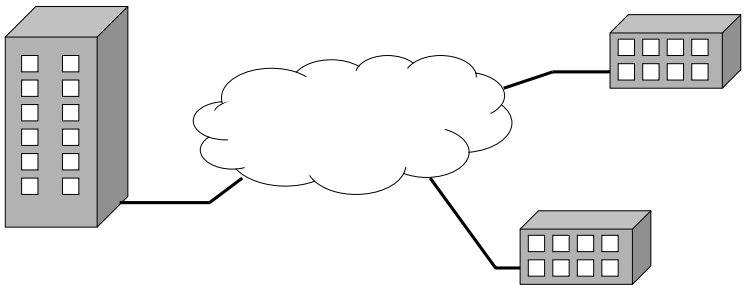

Abbildung 1: intranet VPN

von hohen Ferngesprächsgebühren weichen immer mehr einer VPN Lösung. Diese ist einfacher zu implementieren und zu pflegen. Für Mitarbeiter ist dabei der einfache Ge-

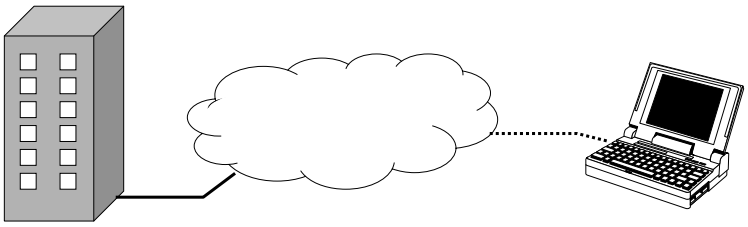

Abbildung 2: remote access VPN

brauch von Sicherheitssoftware wichtig. Es ist erwünscht, dass sich der Mitarbeiter von überall in das Firmennetz einloggen kann und von dort dann dieselben Zugriffsrechte und Möglichkeiten wie in seinem Büro hat. Hier spielen sicher Authentifizierung, Krypthographie und Tunneling eine wichtige Rolle.

- *Extranet VPN*

  Das Ziel bei einem extranet VPN ist die vertiefte Kooperation und strukturiertere Koordination zwischen einer Firma ihren Zulieferern und Kunden (business-to-business VPN). In dieser Topologie muss das virtuelle Netzwerk verschiedene Sicherheitsebenen und Zugriffsbereiche bieten und steuern, so dass es auf sensitive Daten keinen unauthentifizierten Zugriff geben kann. Da die meisten Firmen unterschiedliche Syste-

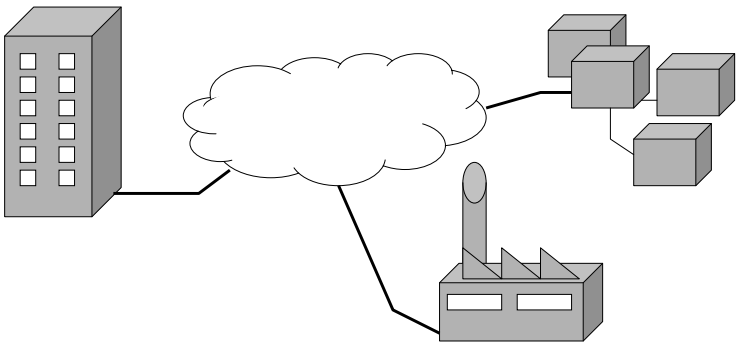

Abbildung 3: extranet VPN

mumgebungen benutzen ist es erforderlich, dass diese VPN Lösung ein äußerstes Maß an Kompatibilität mit den unterschiedlichsten Plattformen, Protokollen, Krypthographiemethoden und Authentifizierungsmechanismen leistet. Auch eine sehr feine Granularität der Zugriffsrechte und Sicherheitsebenen ist hier erforderlich.

## 1.3 Sicherheitsanforderungen an ein Netzwerk

Die Sicherheit des eigenen Netzes und der eigenen Datenbanken ist der Hauptaspekt eines virtuellen privaten Netzes. Die praktische Vernetzung, Kosteneinsparungen und die flexiblere Nutzung sind vergleichbar zweitrangig. Als Betreiber muss man sicher sein, dass die Integrität der Daten geschützt ist, zumal das Bemerken und Verfolgen von einem Einbruch in das Netz meist sehr schwierig ist. Außerdem können Viren als digitales Ungeziefer erheblichen Schaden anrichten.

Daher sind die Sicherheitsanforderungen an ein VPN vielschichtig. Zum einen muss man gewährleisten, dass das eigene Netz nach außen hin gegen unbefugten Zugriff geschützt ist und zum anderen muss man die Kommunikation über das Netz gegen Abhören oder Verfälschung der Daten schützen.

Zum einen sollte man sich bewusst machen, dass sofort das ganze Firmennetz mit all seinen Komponenten mit dem Netz verbunden ist, sobald nur ein einziger Computer des Unternehmensnetzes mit dem öffentlichen Netz verbunden wird. Deshalb sollte eine Kategorisierung der Daten und Systeme und eine Implementierung verschiedener Sicherheitshierarchien vor der Vernetzung erfolgen. Zum Beispiel könnten man sich verschiedene Sicherheitsstufen für Personendaten, Sicherheitsdaten wie Passwörter, Datenbanken, Produktentwicklung, etc. vorstellen. Deshalb sollte man den Verzicht der Vernetzung bestimmter Firmenbereiche nicht von vornherein ausschließen, da dies immer noch die beste Firewall ist. Aber von der praktischen Seite her gesehen sind es ja gerade oft die sensiblen, dynamischen Datenbestände, die von reisenden Mitarbeitern aktuell abgefragt werden müssen.

Zum anderen ist der Schutz des Datentransfers außerordentlich wichtig. Wenn man sensible Daten über ein öffentliches Netz schicken will, ist man an zwei Dingen interessiert: Zum einen will man wissen, mit wem man die Daten austauscht und zum anderen will man sicherstellen, dass sonst niemand mithören oder Daten verfälschen kann. So definieren drei grundsätzliche Schlagwörter die Sicherheit in einem VPN : *Verschlüsselung* (encryption), *Authentifizierung* (authentification) und *Zugangskontrolle* (access control).

Der Schutz des Netzes hängt von der Zusammenarbeit all dieser Komponenten ab. Ist eine Komponente fehlerhaft oder durchlässig, so ist es das ganze Netzwerk. Weiterhin soll eine transparente VPN-Technologie deren Teilnehmer schützen, ohne dass diese davon Kenntnis haben oder etwas dazu beitragen müssen. VPNs erlauben daher, im Gegensatz zu Sicherheitsmaßnahmen auf der Anwendungsebene, einen flächendeckenden Schutz. Zum anderen sollte man sehen, dass Sicherheit und Verfügbarkeit zusammenhängen. Sollen Daten überall möglichst schnell verfügbar sein, so wächst die Gefahr von Sicherheitslücken ebenfalls schnell.

So bilden Sicherheitstechnologien und -software sicherlich den Kern eines guten Netzes. Hierauf wird deshalb besonders ausführlich im Abschnitt 2 eingegangen. Auch wenn manche VPN heutzutage sicherer sind als traditionelle Technik, so ist dies trotzdem der sensibelste Punkt bei einer Entscheidung für ein VPN, da jeder Teilnehmer des öffentlichen Netzes mit seinem Standardaccount und den Standardprogrammen einen potentiellen Zugang zum Firmennetz hat.

## 1.4 Kostenaspekte

Die hohen Kosten und der komplexe Aufbau eines Netzwerkes mit eigener Infrastruktur sind nicht zukunftsweisend. Ferngesprächsgebühren, Gebühren für Standleitungen, Unterhaltungskosten für Modem-Pools und die ständigen aufwendigen Wartungskosten addieren sich täglich zu enormen Summen.

Im Vergleich hierzu sind feste Leitungen mit großen Bandbreiten zum nächsten Internet Service Provider günstiger. Auch die Wartung und Pflege nur eines einheitlichen Systems ist effektiver. Hier eröffnet sich nun die Option des geschickteren Outsourcings. Oft ist es ja das fehlende Wissen in einem Unternehmen, was zu veralteter Technik und damit zu Sicherheitslücken und unplanbaren Kosten führt. Mit Outsourcing hat man die Möglichkeit, für einen garantierten Leistungsstandard planbare Kosten zu haben. Diese Aspekte in konkrete Zahlen zu fassen, fällt dabei sehr schwer. Die dynamischen Tarife der Telefongesellschaften und die unterschiedlichsten Anforderungen je nach Unternehmen würden jede Tabelle in sich unschlüssig machen.

Die wirtschaftlichen Vorteile des remote access hängen natürlich davon ab, woher die Anrufe stammen. Ein VPN macht wenig Sinn, wenn die meisten Verbindungen im Ortstarifgebiet entstehen. In diesem Fall ist es am günstigsten, sich direkt über das öffentliche Telefonnetz in die Firma einzuwählen. Sind aber viele Mitarbeiter weltweit auf Verbindungen angewiesen, ist das VPN weit rentabler als Ferngesprächsgebühren. Somit hat man weltweite Erreichbarkeit und ein kostengünstigeres Wirtschaften in einem Konzept verwirklicht.

Einen weiteren Vorteil bieten die relativ geringen Einstiegskosten. Dabei ist es relativ einfach, das System ständig zu erweitern. So kann bei großen Unternehmen auch eine kombinierte Version bewährter Technik und VPN-Technologie erfolgen.

# 2 Basistechnologie

Dieses Kapitel erklärt und vergleicht die Basistechnologien, die nötig sind, um ein VPN aufzubauen. Dabei soll es zuerst um den Schutz des eigenen Systems nach Außen hin gehen (firewalls), dann um die Sicherheit beim Transfer der Daten (Krypthographie und Authentifizierung) und schliesslich um den Prozess des Übertragens von Daten mittels des Internet (tunneling).

Das eigentlich Neue und Herausfordernde an einem VPN ist, dass all diese Techniken, die ja bekannt sind, zu einem effizienten und sicheren System verknüpft werden müssen. Denn Sicherheit und Schutz ist der Dreh- und Angelpunkt jedes VPN. Und dieses Konglomerat nützlicher Techniken, welche am Anfang manuell zusammengestellt wurden, wird nun immermehr als Systemlösung für Unternehmen angeboten.

## 2.1 Firewalls

### 2.1.1 Arbeitsweise einer Firewall

Eine Firewall ist eine moderne Ausführung der mittelalterlichen Wälle und Mauern rund um Burgen und Ortschaften. So werden Firewall-Funktionen genutzt, um den Zugriff zwischen einem vertrauenswürdigen Netz und einem unsicheren zu kontrollieren. Die Firewall-Technik wird seit vielen Jahren in Netzen verwendet und bildet damit als erprobte Technik eine gute Grundlage für ein VPN.

Eine Firewall ist recht einfach zu kreieren, sie benötigt nur die Modifikation eines Gateway Routers. So können in einem Unternehmen mehrere LANs verbunden sein, wobei der gesamte Datenverkehr nach außen aber durch möglichst ein einziges Verarbeitungs-Gateway fließen sollte. Jedes Datenpaket, dass das Tor passieren will, muss kontrolliert werden.

Den Installationen steht es frei, eines oder mehrere Verarbeitungs-Gateways für spezifische Anwendungen einzurichten. Man sollte jedoch beachten, dass die Komplexität des Systems dramatisch zunimmt, wenn man mehrere Gateway-Router hat.

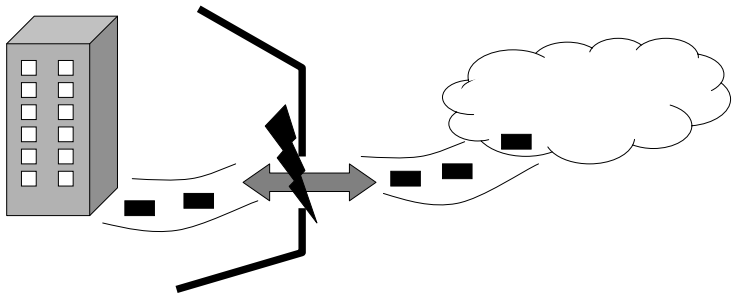

Abbildung 4: Arbeitsweise einer Firewall

In der Praxis ist es daher nicht unüblich, dass Unternehmen nur einen Verbindungsknoten einrichten und außer Email und evt. WWW alle anderen Funktionen für die Benutzer sperren, um eine bessere Kontrolle zu gewährleisten. Es gibt inzwischen auch Werkzeuge, welche die Aktivität überwachen, ist ein Benutzer einmal von außen in das Firmennetz eingeloggt. Obwohl die meisten VPN-Varianten keine Firewalltechnik enthalten, ist sie doch ein integraler Bestandteil eines VPN.

### 2.1.2 Verschiedene Funktionstypen

Eine Firewall erfüllt üblicherweise zwei Funktionen. Die erste ist, dass sie kontrolliert, welche Computer und Server von außen gesehen werden können und damit, welche Funktionen nach außen hin angeboten werden. Die zweite Funktion überprüft welche Maschinen im Internet von innen gesehen weren können und damit, welche Dienste genutzt werden können. Internet Firewalls erfüllen diese Aufgabe meist mit einer Überwachung der durchfließenden Datenpakete ("packet filtration").

Im folgenden sollen einige Ansätze vorgestellt werden, auch wenn es mit Sicherheit unzählige Varianten gibt. Worauf es allerdings hier ankommen soll, ist wie Firewall-Technik in ein VPN passt.

- *Packet Filtering Routers*

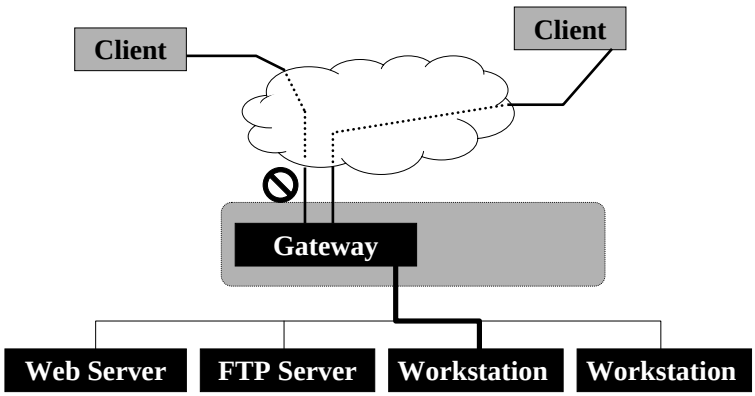


Abbildung 5: Packet Filtering Routers

Wenn ein paketfilternder Router als Gateway zwischen das eigene Netz und das öffentliche Netz geschaltet wird, arbeitet dieser mit einer wohldefinierten Tabelle von Regeln. Dabei entscheidet der Router nicht nach Inhalt eines Pakets oder gar warum ein Paket

zu einer bestimmten Schnittstelle gesendet wurde. Es wird lediglich überprüft, ob das Paket in das Regelwerk der Parameter passt, und danach wird der Transit gestattet oder verweigert.

Diese Entscheidungstabellen müssen nach den Sicherheitsbestimmungen des Unternehmens vorher festgelegt werden. Eine Möglichkeit (und sicherlich die häufigste) ist dabei, aufgrund der Ziel- und Quelladresse zu entscheiden. Dabei findet keinerlei Identifizierung statt, es wird allein aufgrund der Header entschieden.

- *Bastion Host*

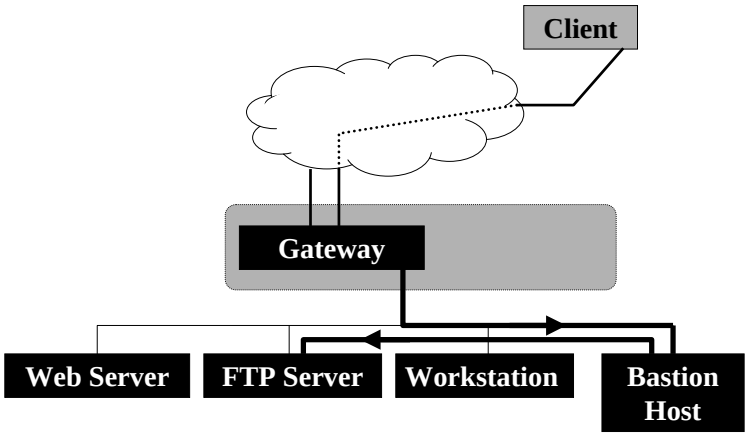


Abbildung 6: Bastion Host

Die Schwierigkeit bei einem paketfilternden Router besteht darin, das richtige Maß zwischen ausreichender Sicherheit und "alles ablehnen" zu finden.

Eine Lösung bietet die Bastion-Host Option. Hier wird Datenverkehr zum Bastion Host (der gleichberechtigt wie alle anderen Computer am Netz hängt) mit einer höheren Akzeptanz durchgelassen. Die grundsätzliche Sicherheitsstruktur bildet aber weiterhin der Router. Hier läuft die Kommunikation aber nicht direkt, sondern über den Bation Host, welcher den gesamten Datenverkehr auf höheren Schichten überwacht.

Gerade bei großen und dynamischen Netzwerken reduziert dies im Vergleich zum einfachen Modell den Verwaltungsaufwand erheblich. So ist das Hinzufügen eines neues Benutzers erheblich einfacher. Und ein zentraler Kontrollpunkt bedeutet gleichzeitig nur ein zentraler Punkt, wo Sicherheitslücken auftreten können. Eine zentralisierte Lösung hat natürlich auch ihre Nachteile. So benötigt ein großes Netzwerk mehrere Bastion-Hosts, die aufeinander abgestimmt sein müssen, um den erhöhten Datenverkehr zu bewältigen, was natürlich die Komplexität wieder erhöht und gegebenfalls auch zu Engpässen führt.

- *Perimeter Zone Network*

  Eine sehr populäre Variante ist die folgende. Hier wird zwischen das öffentliche Netz und das Unternehmensnetz ein Routernetzwerk geschaltet, das als sogenannte entmilitarisierte Zone zwischen den beiden Netzen fungiert.

  Es gibt in dieser Variante viele Sicherheitsvorteile. Zum einen sind mindestens zwei Router an der Kontrolle beteiligt. Einer bildet die Schranke zum Internet, der andere die Schranke zum Unternehmensnetz. Dazwischen liegen nur vertrauenswürdige Router oder Hosts. So können als konzentrische Kreise gedacht noch restriktivere Sicherheitsarchitekturen mit verschiedenen Sicherheitshierarchien kreiert werden.

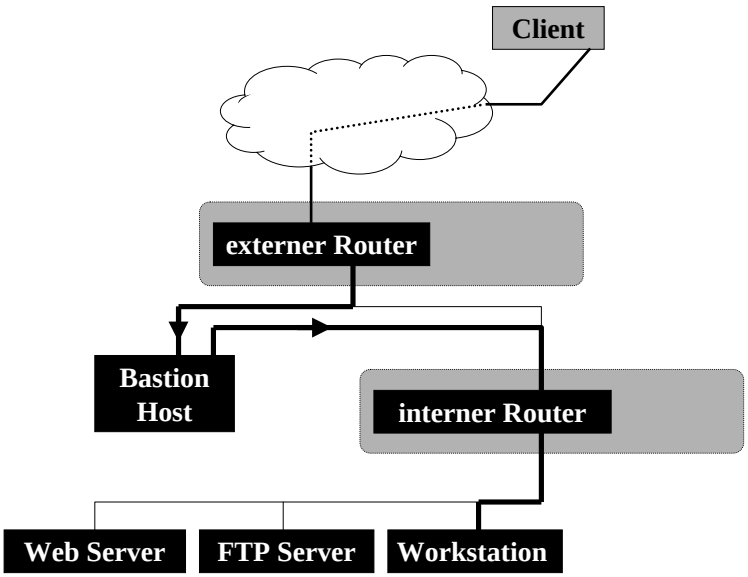


Abbildung 7: Perimeter Zone Network

Die straffste Sicherheit wird geschaffen, indem der gesamte Ausgangsverkehr des internen Netzes verboten wird und dem gesamten Eingangsverkehr der Zugang verweigert wird. Der gesamte Datenverkehr erfolgt nun in zwei Schritten über den Bastion Host.

- *Proxy Server*

  Das Proxy Server Konzept ist dem des Bastion Host sehr ähnlich, und manchmal überlappen sich beide fast komplett. Zum Vergleich: Ein gutes Beispiel ist elektronische Post. Wird eine Nachricht an ein Unternehmen gesendet, wird diese vom Gateway-Router an den Bastion host weitergeleitet, der nun die Weitervermittlung übernimmt. So schickt er sie nun wiederum in das interne Netz oder speichert sie, bis der Benutzer sie über POPmail abholt.

  Das Proxy Konzept soll hiervon insofern abweichend genannt werden, als dass es sich mehr als ein Transit-Kontrollpunkt versteht als ein Informationspool. Hier im besonderen beim Dateitransport (FTP), da dieser mit Zuweisung von flexiblen Portnummern besonders gefährdet ist.

### 2.1.3 Einsatz im Virtual Private Network

Firewalls bilden den Grundstock eines VPN. Um das eigene Netz zu schützen, werden um dieses Sicherheitsbarrieren errichtet. Diese erzeugen kleine Inseln im öffentlichen Netz. Das Konzept der VPN-Firewall-Technik schließt nun ein, dass spezifische Kanäle geöffnet werden, um von einer Insel zur nächsten verschlüsselte Daten über das öffentliche Netz zu senden. VPN-Software bietet dieses Routing transparent auf der Anwendungsebene an, so dass das Netzwerk an jedem Ende als ein zusammenhängendes Netz erscheint. So muss eine Firewall konzipiert, installiert und vor allem getestet werden, bevor Resourcen und Daten freigegeben werden können und man von den Vorteilen einer Vernetzung profitiert. In diesem Zusammenhang müssen die unterschiedlichsten Angriffsszenarien beachtet werden (so zum Beispiel das Überfliegen der Firewall im drahtlosen LAN bei Funknetzen). Die Kombination der entmilitarisierten Zone ist wohl heute das beste Konzept. Man muss sich aber stets bewusst sein, dass bei Vorstellung einer neuen Technik die Bekanntgabe des Knackens dieser Technik nicht lange auf sich warten läßt.

## 2.2 Krypthographie und Authentifizierung

Es ist offensichtlich, dass ein paketfilternder Router allein noch kein VPN ausmacht. So erschweren Firewalls wie getönte Scheiben nur die Sicht, aber durch Krypthographie und Authentifizierung muss das Netz wirklich abgeschottet und damit privat werden. Krypthographie bezieht sich dabei darauf, den Text während der Übertragung unkenntlich zu machen, dass selbst wenn jemand die Daten in die Hände bekommt, er nichts damit anfangen kann. In Verbindung damit bildet dann die Zugangskontrolle (Authentifizierung) einen wichtigen Aspekt im VPN. Das bedeutet, nur ausgewählte Nutzer sollen auf ein bestimmtes VPN zugreifen können. Ja sogar noch mehr. So soll ein reisender Mitarbeiter dieselben Zugriffsrechte und -beschränkungen haben, als ob er an seinem Arbeitsplatz sitzt. Geradeauch bei extranets, dem virtuellen Verbundnetz mehrerer Firmen, muss das verwendete VPN differenzieren können. Eine feine Granularität, welche den einzelnen VPN-Teilnehmern nur jeweils eingeschränkte Dienste zu bestimmten Datenbanken zur Verfügung stellt, ist hier unerläßlich.

### 2.2.1 Krypthographietechniken

Krypthographie (Verschlüsselung) ist eine mathematische Transformation, die einen Klartext (den eine Person oder ein Programm interpretieren kann) in einen verschlüsselten Text (den man nicht interpretieren kann) verwandelt. Dies geschieht mit einem sogenannten Schlüssel. Die verwendeten Algorithmen sind im Allgemeinen recht komplex und benötigen dazu in der Regel sehr große Primzahlen. Die modulo Funktion spielt außerdem eine große Rolle, da man bei der modulo Funktion im Nachhinein nicht erkennen kann, auf welche Zahl sie angewendet wurde, selbst wenn man deren Ergebnis kennt. (z.B. die modulo 3 Funktion mit Ergebnis 2. Welche Zahl war es ursprünglich? 11, 65 oder gar 12377 ?)

Je nachdem ob man für Ver- und Entschlüsselung den gleichen oder unterschiedliche Schlüssel verwendet, spricht man von symmetrischer oder asymmetrischer Krypthographie. Diese haben einen signifikanten Unterschied. Symmetrische Algorithmen sind relativ schnell, der Schlüssel ist recht klein und bei beiden Seiten bekannt. So werden diese Algorithmen immer eingesetzt, wenn große Datenmengen ausgetauscht werden müssen und sich die beiden Kommunikationspartner kennen. Asymmetrische Techniken finden dagegen oft bei kleinen Datenmengen statt, so z.B. beim Authentifizierungsprozess oder beim Austausch von Schlüsseln [Shiv98].

Ein recht bekannter und erprobter Algorithmus ist der *Data Encryption Standard (DES).* Dies ist ein komplexer, symmetrischer Algorithmus der immer 64-bit Blöcke mit einem 56-bit Schlüssel (mehr erlaubt der amerikanische Geheimdienst nicht) verschlüsselt. Um die Effektivität dieses Systems zu erhöhen, nutzt man nun Tripel Pass DES. Tripel Pass DES nutzt den DES Algorithmus mehrmals. Zuerst wird der Klartext verschlüsselt, dann mit einem falschen Schlüssel entschlüsselt (was natürlich nur Müll liefert) und dann zum zweiten mal mit dem ersten Schlüssel verschlüsselt. Dies hat zu Folge, dass jemand, der den Code entschlüsselt, es überhaupt nicht merkt, da er auf eine sinnlose Zeichenkette stößt.

*Outer Cipher Block Chaining (CBC)* ist eine weitere Technik, um die verschiedensten DES-Algorithmen zu verbessern. Diese Technik läßt per Zufall in den verschlüsselten Text "Zufallsfülldaten" einfließen. So wird der gleiche Text jedesmal in einer anderen Weise übertragen. Will man zum Beispiel seine Nachrichten auf dem Mailserver abrufen, ist die Pozedur jedesmal dieselbe. Da Protkolle nicht nur sehr gut definiert sind, sondern auch sehr bekannt, macht diese Zufallstechnik außerordentlich viel Sinn.

### 2.2.2 Authentifizierung in Rechnernetzen

Die gegenseitige Identifizierung beim Aufbau einer Verbindung ist existentiell für ein VPN. So muss nicht nur das Netzwerk nach außen geschützt werden, sondern auch der reisenden Mitarbeiter muss sich sicher sein, mit dem richtigen Server verbunden zu sein. Dieser Identifizierungsvorgang wird meist mit Authentifizierung bezeichnet (authentification). Die meisten Methoden beruhen dabei auf dem Prinzip von bestimmten Schlüsseln, die durch Hash-Algorithmen einen Hash-Wert generieren, der dann recht eindeutig und gut vergleichbar ist.

Drei Methoden sind dabei zu unterscheiden: *Certificate*, *Challenge Phrase* und *RADIUS*. Diese werden nun im folgenden vorgestellt.

- *Certificate*: Das digitale Zertifikat enthält Datenstrukturen, die Informationen enthalten, welche die positive Gültigkeit beweisen. Man kann sie mit einem Reisepass vergleichen. Es enthält Informationen über den Teilnehmernamen, den Zertifikatnamen, Gültigkeitsdauer, ... Neben den beiden kommunizierenden VPN-Komponenten existiert noch eine dritte vertrauenswürdige Autorität (als Softwareprogramm), die das Zertifikat ausstellt (wie eine Passstelle).

  Nachdem die Zertifikate ausgetauscht und verifiziert wurden, kann der Schlüssel für diese Nutzungsperiode ausgehandelt werden.

- *Challange Phrase*: Die Challange Phrase Technik ist ähnlich zu der Certificate Methode, nur dass es hier keine dritte Komponente gibt. Hier müssen die Berechtigungen "manuell" in den verschiedenen Komponenten implementiert werden, damit sich die verschiedenen Komponenten kennen, wenn sie zum erstenmmal miteinender kommunizieren.

- *RADIUS*: Auch diese Methode funktioniert vom Grundansatz ebenso wie die Certificate Methode. Auch hier existiert eine dritte unabhängige Autorität, die die Zertifikate vergibt. Bei der Certificate Methode tritt diese Autorität aber nur beim Ausstellen des Zertifikats in Erscheinung, wohingegen bei der RADIUS Methode bei jedem Aufbau einer Verbindung dieser Server kontaktiert wird, ob das aktuelle Zertifikat gültig ist. Ist diese Variante implementiert, besteht zusätzlich die Möglichkeit, einen Monitor zu installieren, der sämtliche aufgebaute Verbindungen zentral speichert.

### 2.2.3 Sicherheit im Virtual Private Network

Um den immer stärkeren Wunsch nach sicherer Internet-Kommunikation zu erfüllen, wurde der *IP Security Standard (IPSec)* definiert. IPSec standardisiert die Anwendung von krypthographischen Algorithmen zur Verschleierung und Authentifizierung. Um Interoperabilität zu gewährleisten, verlangt IPSec die Unterstützung von weitverbreiteten Algorithmen wie dem Data Encryption Standrad (DES) zur Verschleierung und dem Message-Digest-5-Algorithmus (MD5) zur Authentifizierung. IPSec kennt einen Tunnel (siehe nächsten Abschnitt) , dadurch kann IPSec zum Aufbau eines VPN benutzt werden. IPSec ist den heute verfügbaren IP-Routern der meistbenutzte Sicherheitsstandard.

## 2.3 Tunneling

### 2.3.1 Arbeitsweise eines Tunnels

Bei den bisherigen Betrachtungen blieb außer acht, dass es immer schwieriger wird, zwei verschiedene Netze miteinander zu verbinden. Die Vielfältigkeit der unterschiedlichsten Pro-

tokolle und Netztypen erschwert den Zusammenschluss. So sind firmeneigene Netze oft nicht auf IP -Basis, oder wenn sie es sind, will man bestimmt keine IP-Netze zum Datentransfer nutzen. Andererseits gibt es aber einen Sonderfall, der bei VPN oft zutrifft und der handhabbar ist. Das ist der Fall, in dem der Quell- und der Zielhost am gleichen Netztyp hängen, dazwischen aber ein anderer Netztyp liegt. Als Beispiel kann man sich ein internationales Unternehmen vorstellen mit einem auf TCP-IP basierten Ethernet in Frankfurt und Boston sowie einem dazwischenliegenden PTT-WAN. Eine Lösung für diese Art von Problemen bietet das sogenannte Tunneling. (Ist bei einem VPN allerdings die Rede vom sogenannten "tunneln" ist meist das Einkapseln von Datenpaketen in IP-Pakete gemeint, die über das Internet übertragen werden können.) Als Vergleich könnte man sich eine Autofahrt von Paris nach London vorstellen. So fährt man in Paris mit dem Auto los. An der (Netz)Grenze muss das Auto auf ein anderes Transportsytem wie die Bahn unter dem Ärmelkanal verladen werden. Danach kann das Auto wieder selbst nach London weitergesteuert werden [Tane97].

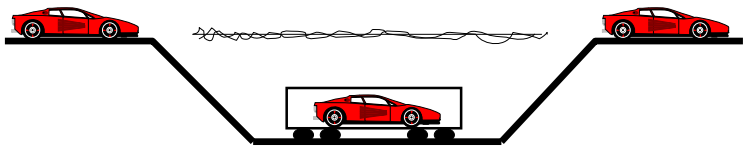

Abbildung 8: Arbeitsweise eines Tunnels

Das Einrichten eines Tunnels hat in den Routern zur Folge, dass sie über ihre Tunnel-Interfaces Pakete austauschen können, als ob sie über eine direkte Punkt-zu-Punkt Verbindung miteinander verbunden wären. Der Unterschied zwischen dem Tunnel-Interface und dem physikalischen Interface besteht darin, dass durch das Tunnel-Interface z.B. IP Pakete nochmals in IP Pakete eingekapselt werden. Das heißt, dass ein über einen Tunnel übertragenes Paket zwei Header hat.

*GRE (Generic Routing Encapsulation)* spezifiziert einen allgemeinen Mechanismus, um ein beliebiges Protokoll über IP zu tunneln. Zwischen den zu übertragenden Daten und dem IP-Tunnel-Header wird dabei ein GRE-Header eingefügt, der neben Kontrollflags eine Kennung des eingekapselten Protokolls enthält. Die sogenannten Security Gateways an den Tunnelendpunkten verbergen Quell- und Zielort sowie den Inhalt über das Internet verschickter Pakete

Die bisher beschriebene Technik des Tunnelns funktioniert also zwischen den Routern der verschiedenen Firmenstandorten. Um aber auch für reisende Mitarbeiter die Tunneltechnik verwenden zu können, muss es möglich sein, von einem Endsystem einen Tunnel aufzubauen.

Hier sollen im folgenden zwei grundsätzliche Protokollansätze vorgestellt werden, die beispielhaft die verschiedenen Ansätze verdeutlichen sollen.

### 2.3.2 Das Layer 2 Tunneling Protocol (L2TP)

Eine Möglichkeit besteht darin, dass man mit dem Endsystem eine PPP-Verbindung zum nächsten VPN-Router des eigenen Netztes aufbaut und dann die Pakete über das Internet tunnelt. Allerdings ist es oft nicht wünschenswert (und ja damit Ziel des VPN), eine PPP-Verbindung zu einem evtl. sehr weit entfernten VPN-Router aufzubauen.

Was man eigentlich möchte ist ja, dass man sich bei einem nahgelegenen Zugangsknoten des Internet einwählt und dann einen Tunnel aufbauen kann. Dies leistet das *Layer-2 Tunneling*

*Protocol (L2TP)*, das sich zur Zeit in der Standardisierung befindet. (Dies ist eine Weiterentwicklung des PPTP (Point-to-Point Tuneling Protocol) der Firma Microsoft und des L2F (Layer 2 Forwarding) der Firma Cisco.)

In einem auf L2TP basierenden VPN wählen sich die Endsysteme über PPP beim nächsten Zugangsknoten ein, der nun wiederum mit seinem Interface zum Internet verbunden ist (Dieser Zugangsknoten wird bei diesem Szenario oft L2TP Access Concetrator, LAC, genannt). In Abhängigkeit des Nutzers sucht nun der LAC den passenden VPN-Router aus (der in dieser Terminologie oft mit L2TP Network Server (LNS) bezeichnet wird). Nun etabliert der Zugangsknoten durch Austausch entsprechender L2TP Kontrollnachrichten zum VPN-Router einen IP-Tunnel. Typischerweise wird dabei der Zugangsknoten von einem Netzbetreiber verwaltet und der VPN-Router eher von der Firma selbst. Ist der Tunnel dann etabliert, werden

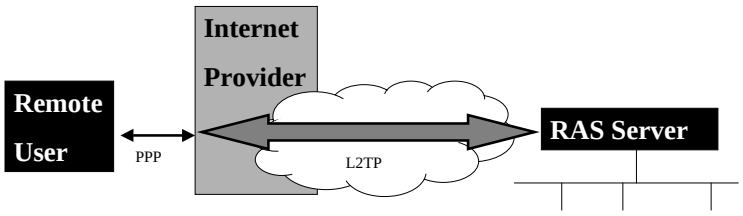


Abbildung 9: Providerabhängige Lösung

im Zugangsknoten die Pakete gekapselt und über IP an die Tunnelendpunkte übertragen. Durch den Tunnel entsteht dadurch für den Nutzer der Eindruck, als sei die PPP-Verbindung mit dem Router direkt aufgebaut worden. Im VPN-Router werden die Pakete dann wieder entpackt und an ein die entsprechenden Schnittstellen weitergeleitet.

Der Vorteil dieser Variante liegt sicher beim mobilen Endgerät, welches außer den PPP Protikollimplementationen keine weiteren Ausstattungen benötigt.

Der Nachteil, der auch schon zu der zweiten Variante überleitet, ist sicherlich der, dass der Zugangsknoten mit der entsprechenden Software ausgestattet sein muss. Wählt man sich mit seinem Endgerät an einem Zugangsknoten ein, der diese Tunnelvariante nicht unterstützt, hat man keine Möglichkeit, einen Tunnel aufzubauen.

Ein Ziel sollte es sicherlich sein, dass man mit seinem Endgerät unabhängig vom Provider und dessen Implementierungen einen Tunnel zum eigenen VPN aufbauen kann. Dies sollte aber trotzdem für den Benutzer transparent sein. Da die meisten reisenden Mitarbeiter nicht Computerspezialisten sind, sollte der Aufbau des Tunnels also nicht aufwendiger sein, als der Aufbau einer PPP-Verbindung.

Diese providerunabhängige Lösung bietet einen Tunnel direkt vom Endgerät bis zum firmeneigenen VPN-Router. Hier baut der Mitarbeiter eine PPP-Verbindung zum nächsten Network

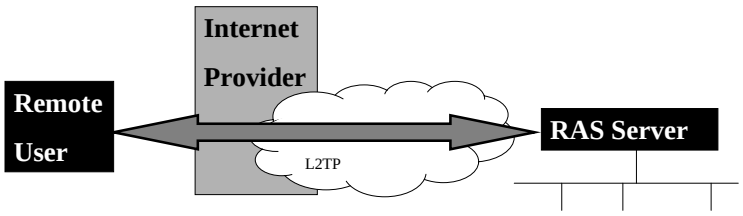


Abbildung 10: Providerunabhängige Lösung

Service Provider auf. Ist diese Verbindung etabliert, wird das dial-up Programm zum zweiten

mal gestartet. Diesesmal wird als Telefonnummer die IP-Adresse des Firmennetztes mit dem VPN-Port eingetragen. Der eindeutige Nachteil ist sicherlich bei dem L2TP Konzept, dass dies alles nur mit Windows NT-Systemen möglich ist und es für andere Systeme keine Option gibt.

### 2.3.3 Der Alta-Vista Tunnel als Alternative

Nachdem im letzten Abschnitt ein typisches Tunnel-Szenario dargestellt wurde, soll nun das Alta-Vista Tunnel-Konzept angerissen werden. Es liefert einige alternative Betrachtungsweisen zum L2TP. Im Allgemeinen kann man sagen, dass es auf diesem Gebiet sehr viele Neuerungen gibt und geben wird und sich noch keine Version als ausreichend und optimal durchgesetzt hat. Gerade in Punkto Sicherheit gibt es hier Bedenken, da durch die Tunneltechnik die Firewall durchlässig wird.

Benötigen die meisten Tunnelprotokolle feste IP-Adressen, so arbeitet der Alta-Vista Tunnel mit flexiblen virtuellen IP-Adressen. Jedes Alta-Vista Tunnel Netwerk besteht aus zwei Seiten, die mit dem Internet verbunden sind. So wird der Tunnelverkehr, der für das VPN bestimmt ist (hier kann man zwischen unterschiedlichem Datenverkehr gleichzeitig differenzieren), mit der Tunnelsoftware von der physikalischen IP-Adresse zur virtuellen VPN-IP gesendet. Diese virtuelle IP-Adresse ist nun über das VPN mit der virtuellen IP-Adresse des Servers verbunden, der nun wiederum an die physikalische IP weiterleitet. Alta-Vista arbeitet hierbei mit dem Konzept von Arbeitsgruppen, Passwörtern und natürlich auch Verschlüsselung.

So bietet der Alta-Vista Tunnel den Vorteil von flexiblen IP-Adressen, die jedesmal anders sein können. Die Sicherheitsbedenken sind durch die Software nicht gänzlich aus dem Weg zu räumen und bilden heutzutage sicherlich das größte Problem bei dieser Option.

# 3 Perspektiven

## 3.1 Vor- und Nachteile eines Virtual Private Network

Die Vorteile des VPN übertreffen in bestimmten Bereichen die herkömmliche Technik. Für sogenannte Neueinsteiger sind die relativ geringen Einstiegskosten interessant, und für Unternehmen mit bisher eigenen Leitungen ist eine kombinierte Version aus gemieteten Standleitungen und der Nutzung des öffentlichen Netztes als Erweiterung der bisherigen Möglichkeiten zu beachten.

Mobile Mitarbeiter können sich über Internet-Verbindungen zum Ortstarif ins Unternehmensnetzwerk einwählen. Standorte lassen sich kostengünstig über IP-Backbones anbinden (lokale Fest- oder Dial-up Verbindungen). Auch die Errichtung eines effizienten Extranets mit einer feinen Granularität wird durch VPN-Technologie erst richtig möglich.

Zum einen steigt damit die *Flexibilität* und *weltweite Erreichbarkeit* sprunghaft und zum anderen kann man eine *Kostenreduktion* im Unternehmen durch Einsparungen von Mietleitungen erreichen. Auch das Management der eigenen Server läßt sich bei Bedarf nun auch leichter einem Provider übertragen (outsourcing).

Als Nachteil und als grundsätzliches Problem ist die Sicherheit eines VPN, das über das öffentliche Netz errichtet werden soll, anzusehen. Die Abschottung des eigenen Netzes vom öffentlichen ist vielfach nicht effektiv genug. Hier spielen auch die internationalen rechtlichen Beschränkungen einer hochwertigen Verschlüsselung hinein. So gestatten amerikanische Exportbeschränkungen nur geringe Schlüssellängen von bis zu 56 bit. So hängt in der Zukunft

der Erfolg von VPNs bestimmt auch von den Regelungen und Gesetzen der Krypthographie ab. So kann bei erforderlicher Sicherheit und Integrität der Daten nur von einem VPN abgeraten werden.

Ein weiterer Nachteil, der aber mit der Zeit wohl besser in den Griff zu bekommen ist, ist die Dienstgüte in öffentlichen Netzen und das Angebot von Differentiated-Services [Hurw97]. Gerade im Internet stellt die Garantie von bestimmten Bandbreiten ein Problem dar, da es oft viele Netzübergänge zu den unterschiedlichsten Providern gibt. Aber gerade die garantierte Bandbreite für Echtzeitunterstützung ist oft für Unternehmen ausschlaggebend ("Wenn es darauf ankommt, muss man sich drauf verlassen können"). Hier werden sich wohl mit B-ISDN über ATM mit QoS und dem Prinzip der virtuellen Standleitung in Zukunft neue Optionen ergeben.

## 3.2 Ein Ausblick

Im Zuge der allgemeinen Popularität der Internet-Technologien ist eindeutig ein Trend zu speziellen IP-VPNs zu erkennen, die als öffentliches Netz das Internet nutzen.

Heutigen VPNs gemeinsam sind einige Schwachstellen. Hier stehen wohl drei Aspekte im Blickfeld: *Management, Sicherheit und Dienstgüte*

Meist ist das VPN-Management aufwendiger als die Anbieter von VPN-Technologie versprechen. So ist das Management von aufwendiger erprobter Technik mit Erfahrung wohl doch meist viel schneller als neueste Technik mit hohem Update- und Wartungsbedarf. Einheitliche, plattformunabhängige VPN-Managementsysteme sind kaum verfügbar. Das erweist sich speziell dann als Problem, wenn Netzkomponenten unterschiedlicher Hersteller eingesetzt werden. Eine Alternative, um den Management-Aufwand eines VPNs zu reduzieren, besteht im Outsourcing des VPN-Managements vom Anwender zum Netzbetreiber. (Was auch die Begeisterung von Providern zu VPNs verständlich macht.)

Auch die Sicherheit des Netzes hat oft viele Lücken. So sind Hacker den Firewall Herstellern meist schon einen Schritt voraus. Eine Firma, die absolute Sicherheit benötigt, ist immer noch besser beraten, ein eigenes Netz zu betreiben.

Ein weiteres, sehr signifikantes Poblem ist die mangelnde Dienstgüteunterstützung in der IP-Protokollarchitektur. Selbst wenn ein Unternehmen bereit ist mehr zu bezahlen, bekommt es seine Daten nicht schneller durch das Netz übertragen. So ist von entscheidender Bedeutung, wie sich der Ansatz der Differentiated Services Architekturen entwickelt und international durchgesetzt werden kann.

So bietet ein VPN für viele Unternehmen eine interessante Option zur Erweiterung der eigenen Flexibilität. Und auch für Unternehmen, die nicht auf hohe Datenraten und hohe Sicherheit, aber wohl auf Erreichbarkeit angewiesen sind, ist ein VPN eine innovative Lösung. Da der mobile Zugriff zum Unternehmensnetzwerk zur Verbesserung der Kommunikation und Steigerung der Produktivität wohl immer bedeutender wird, kann die VPN-Technologie in Zukunftwohl ein großes Wachstum verzeichnen.

## Literatur


[Asce97] Inc. Ascedend Communications. Virtual Private Networks, 1997.

[Aven99] Corporation Aventail. Making sense of Virtual Private Networks, 1999.

[Brau99] T. Braun. Virtuell aber real. *NET*, April 1999, S. 36ff.

[Hurw97] M. Hurwicz. A virtual private Affair. *BYTE Magazine*, july 1997.

[Shiv98] Corporation Shiva. Networking Concepts, Juni 1998.

[Tane97] Tanenbaum. *Computernetzwerke*. Prentice Hall. 1997.

# Transport von Signalisierungsnachrichten über IP

Alfons Maas

**Kurzfassung**

Zukünftig werden die beiden Kommunikationsnetze Internet und Telefonnetz immer weiter zusammenwachsen. So wird es erforderlich, Signalisierungsnachrichten des klassischen Telefonnetzes auch über IP-Netzwerke zu transportieren. Die IETF Working Group „Sigtran“ beschäftigt sich mit der Umsetzung. Um einen Überblick über die grundlegenden Ideen für den Transport von Signalisierungsnachrichten zu geben, wird zuerst die Struktur des Zeichengabesystems Nr. 7 und die damit verbundene Problematik vorgestellt. Der Transport von Signalisierungsnachrichten über IP bedarf eines transparenten nachrichtenbasierten Signalisierungsprotokolls. Die Schnittstelle zwischen dem öffentlichen Telefonnetzes und dem Paket-orientierten Netz bildet das sogenannte „Backhaul“. Das Telefonnetz stellt in Bezug auf Ausfallsicherheit und Verfügbarkeit weit höhere Anforderungen als ein IP-Netz. So sind weitreichende Maßnahmen wie der Session Manager für eine redundante Netzwerkkonfiguration oder der Gebrauch von RUDP, ein zuverlässiges UDP, notwendig. Um Anwendungen den Transport von Signalisierungsnachrichten zu erleichtern und die notwendigen Anforderungen zu gewährleisten ist MDTP, ein experimentelles Protokoll, entwickelt worden.

## 1 Motivation

Zur Zeit existieren noch zwei wesentlich unterschiedliche Kommunikationsnetze: das Internet und das Telefonnetz. Signalisierungsnachrichten werden – insbesondere mit dem „Zeichengabesystem Nr. 7“ der ITU-T – in den klassischen öffentlichen Telefonnetzen in einem eigenen Netz mit eigenem Protokollstack eingesetzt. Signalisierung, auch als Zeichengabe bekannt, wird für die Vermittlung, also Auf- und Abbau, der Nutzkanäle sowie für die Steuerung von Leistungsmerkmalen, z.B. für besondere ISDN-Dienste, verwendet. Der im folgenden vorgestellte Vorschlag der IETF Working Group „Sigtran“ soll eine Basis für kommerzielle Anwendungen sein, welche verbindungsorientierte Netze mit paketorientierten koppeln und/oder ganz spezielle Anforderungen an das Netz stellen, wie z.B. die Internet-Telefonie. Auch in Blick auf die Zukunft, in der mit Sicherheit diese Netze weiter zusammenwachsen werden, ist diese Funktionalität notwendig, da derzeitige Internet-Protokolle die hohen Anforderungen an Telefonnetze in Bezug auf Verfügbarkeit, Ausfallsicherheit und Erreichbarkeit nicht erfüllen können. Die einzelnen Kommunikationsknoten zur Kopplung der beiden Netze, die Gateways, müssen in der Lage sein, gleichzeitig mehrere tausend Kommunikationsverbindungen zu verwalten und zu vermitteln, welche einen echtzeitkritischen Charakter besitzen. So dürfen bei einem Telefongespräch keine großen Verzögerungen bei der Übermittlung der Sprachdaten auftreten oder die Verbindung einfach unterbrochen werden. In paketorientierten Netzen gibt es weiterhin das Problem, daß oftmals „Pakete“ doppelt oder in falscher Reihenfolge ausgeliefert werden, was für diese Art der Kommunikation absolut unbrauchbar ist und die Vermittlungsknoten unnötig belastet. Gerade kommerzielle Anwendungen stellen einen hohen Anspruch, welchen das Internet heute noch nicht erfüllen kann.

# 2 Die IETF Working Group „Sigtran“

Die IETF Working Group „Sigtran“ (Signaling Transport) beschäftigt sich mit der Umsetzung des Transportes von Signalisierungsnachrichten über IP-Netzwerke. So muß das IP-Netzwerk den Anforderungen der PSTN-Signalisierung (Public Switched Telephone Network) in Hinblick auf Funktionalität und Leistung genügen und in der Lage sein, Nachrichten des „ZGS Nr. 7“ (SS7) zwischen IP-Knoten und Knoten des öffentlichen Telefonnetzes auszutauschen. Dies ermöglicht eine Kopplung des Telefonnetzes mit dem Internet, um z.B. die Internet-Telefonie an das weltweite PSTN-Netz anzubinden und auch Wählverbindungen anzubieten.

Das existierende Telefonnetz besitzt sehr hohe Anforderung in Bezug auf Ausfall und Verzögerung bei der Übermittlung von Signalisierungsnachrichten. Diese müssen für eine Umsetzung auch auf IP-Netzwerken gewährleistet werden. Die Working Group möchte eine Möglichkeit ausarbeiten, um mittels TCP oder UDP ein Signalisierungstransportprotokoll zu definieren. Dabei wird auf vorhandene Ansätze der IETF für Dienstgüte und Sicherheitsmechanismen zurückgegriffen. Weiter sollen keine neuen Rufsteuerungs/Rufabwicklungs-Protokolle (Call Control) kreiert werden.

Um die Besonderheiten und Konzepte in Bezug auf den Transport von Signalisierungsnachrichten über IP zu verstehen, werden im folgendem Abschnitt die wesentlichen Merkmale und Einsatzgebiete des Zeichengabesystems Nr. 7 (ZGS Nr. 7) vorgestellt.

# 3 Das Zeichengabesystem Nr. 7 (ZGS Nr. 7)

## 3.1 Überblick und Entwicklung

Das derzeitige Haupteinsatzgebiet des ZGS Nr. 7 [BGGH+95],[Sieg99] ist die Informationsübertragung für den Aufbau von Nutzkanalverbindungen. Die Signalisierung für den Auf- und Abbau von 64-kbit/s-Nutzkanalverbindungen und zur Steuerung von ISDN-Diensten erfolgt auf der Basis des ZGS Nr. 7 Protokollstapels – insbesondere ISUP auf der Anwendungsebene – und nicht auf der Basis des D-Kanalprotokolls, das lediglich auf Teilnehmerseite verwendet wird. Das Zeichengabesystem Nr. 7 wurde erstmals von der CCITT, jetzt ITU-T, im Gelbbuch (1980) veröffentlicht und bereits für digitale Vermittlungsstellen in USA Anfang der achtziger Jahre eingesetzt. Es war das erste Zentralkanal-Zeichengabesystem, d.h. mit diesem Verfahren wurden zum ersten Mal die Nutzwege von den Signalisierungswegen getrennt behandelt. Kennzeichnend für das Verfahren ist die Übertragung von Signalisierungsinformationen für viele Nutzkanäle gemeinsam in separaten Signalisierungskanälen, den zentralen Zeichengabekanälen (ZZK), siehe Abbildung 1, unten.

Bei den kanalgebundenen Systemen mußten die Signalisierungszeichen je Übertragungsschnittstelle generiert werden und auf der Empfängerseite von den Nutzsignalen getrennt sowie das Signalisierungszeichen erkannt werden (siehe Abbildung 1, oben). Über die Signalisierungskanäle werden nun die Steuerrechner der einzelnen Netzknoten direkt miteinander verbunden.

Die wichtigsten Funktionen des ZGS Nr. 7 sind:

1. Abschnittsweise Übertragung von Signalisierungsnachrichten zwischen den beteiligten VSt über zentrale Signalisierungskanäle.
2. Überwachung und Steuerung des Signalisierungsnetzes.
3. Ende-zu-Ende-Signalisierung, d.h. Austausch von Signalisierungsnachrichten zwischen Ursprungs- und Zielvermittlungsstelle für die Rufabwicklung und Unterstützung von ISDN-Dienstmerkmalen.

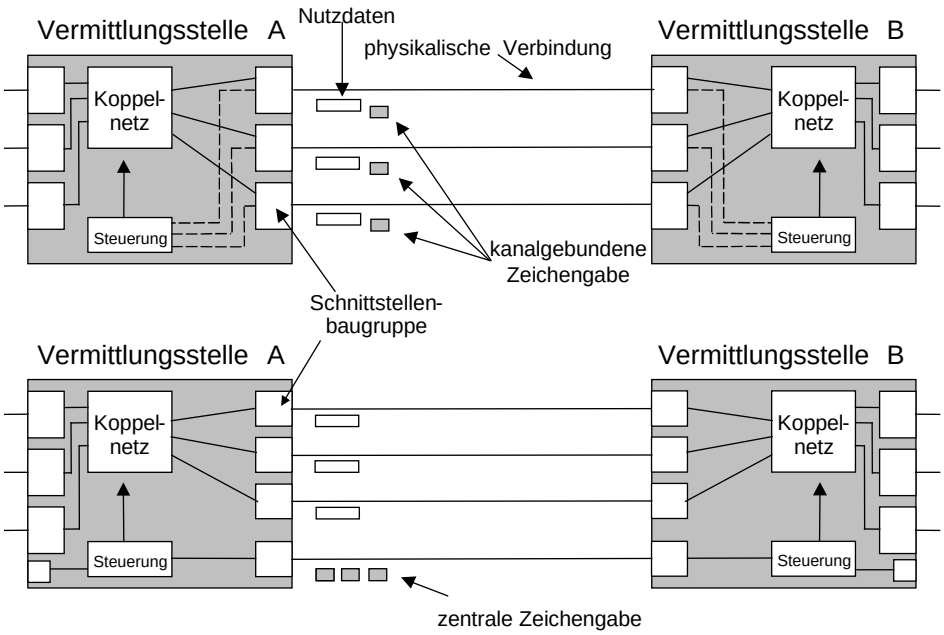


Abbildung 1: zentrale Zeichengabe

### 3.2 Das Signalisierungsnetz

Die Signalisierung geschieht in getrennten Kanälen, die den Nutzkanälen eines Primärsystems entsprechen. So können die Signalisierungskanäle wie Nutzkanäle behandelt werden (z.B. Vermittlung durch Koppelnetze ohne Bearbeitung der Signalisierung). Dadurch ergeben sich Signalisierungspunkte (Signaling Points – SP) und Signalisierungstransferpunkte (Signaling Transfer Points – STP). Die SP sind die Instanzen der an der Signalisierung beteiligten Vermittlungsstellen, d.h. sie verarbeiten Signalisierungsnachrichten, während die STP die Instanzen der weiterleitenden Vermittlungsstellen sind. Die Rollen von SP und STP können sich so je Verbindung ändern. Die Instanzen der Nr. 7-Signalisierung bilden zusammen mit den Signalisierungskanälen ein eigenes Signalisierungsnetz. Die Verbindungen im Nutzwegenetz werden durch die ausgetauschten Signalisierungsinformationen im Signalisierungsnetz gesteuert. Die Gespräche werden digital (PCM-Verfahren) im Zeitmultiplex in den Leitungen des Nutzwegenetzes übertragen.

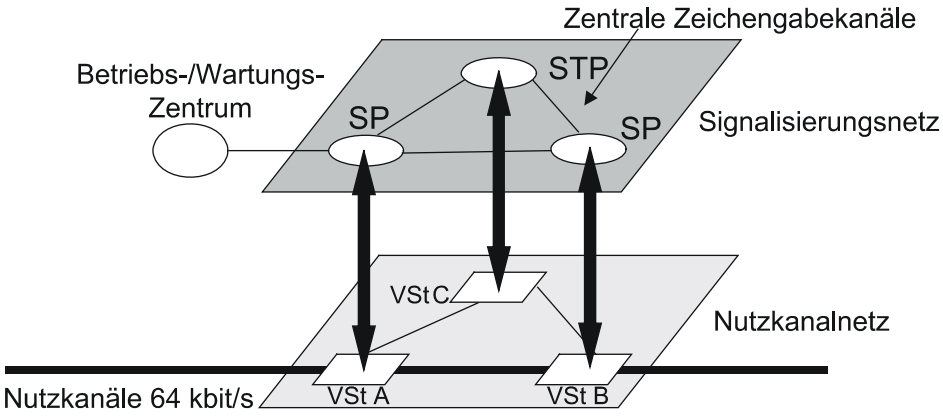


Abbildung 2: Signalisierungspunkte

Ausfälle im Transferbereich der Signalisierung können relativ einfach durch entsprechende Umwegschaltung der Signalisierungskanäle behoben werden. Vermittlungsstellen, welche die

SPs innerhalb des Signalisierungsnetzes bilden, müssen aus Sicherheitsgründen immer an zwei unterschiedlichen Vermittlungsstellen (STPs) angeschlossen sein.

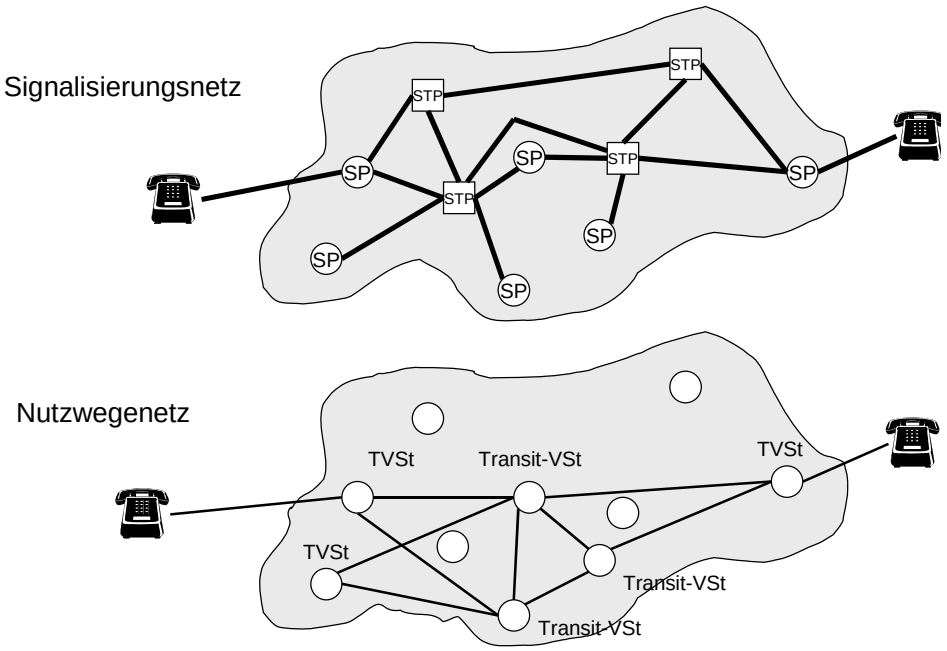


Abbildung 3: Signalisierungsnetz

Das weltweite Fernsprechnetz setzt sich aus unterschiedlichen nationalen Netzen zusammen. Die Grundlage für den logischen Zusammenhalt der einzelnen Netze sind ZGS Nr. 7-Signalisierungskanäle, welche die einzelnen Netze miteinander verbinden. Über diese Signalisierungskanäle werden auch die Elemente des Intelligenten Netzes erreicht. Diese erlauben es, Dienste statt Endeinrichtungen zu adressieren (z.B. Service 0800 oder persönliche Rufnummern). Für diese Steuerung sind nur Signalisierungskanäle notwendig, die Vermittlung der Nutzkanäle erfolgt mit den Netzelementen des jeweils beteiligten Netzes.

### 3.3 Aufbau

Das ZGS Nr. 7 ist wie das OSI-Referenzmodell in mehreren aufeinander aufbauenden Ebenen eingeteilt (siehe Abbildung 4). Diese Ebenen sind allerdings nicht genau deckungsgleich mit den Schichten im OSI-Modell, da beide zeitlich parallel entwickelt wurden.

zu den einzelnen Komponenten:

- MTP (Message Transfer Part): Der MTP bildet die unteren drei Ebenen – also die Basis – des ZGS Nr. 7. Seine Funktionen müssen in jedem Knoten des Signalisierungsnetzes implementiert sein. Er unterteilt sich in die Ebenen:
  - Ebene 1 (Zeichengabekanal):
    - Zugriff über Koppelnetze
    - physikalische Bitübertragung
  - Ebene 2 (Zeichengabestrecke):
    - gesicherte Zeichenübermittlung
    - Rahmensynchronisation
  - Ebene 3 (Zeichengabenetz):

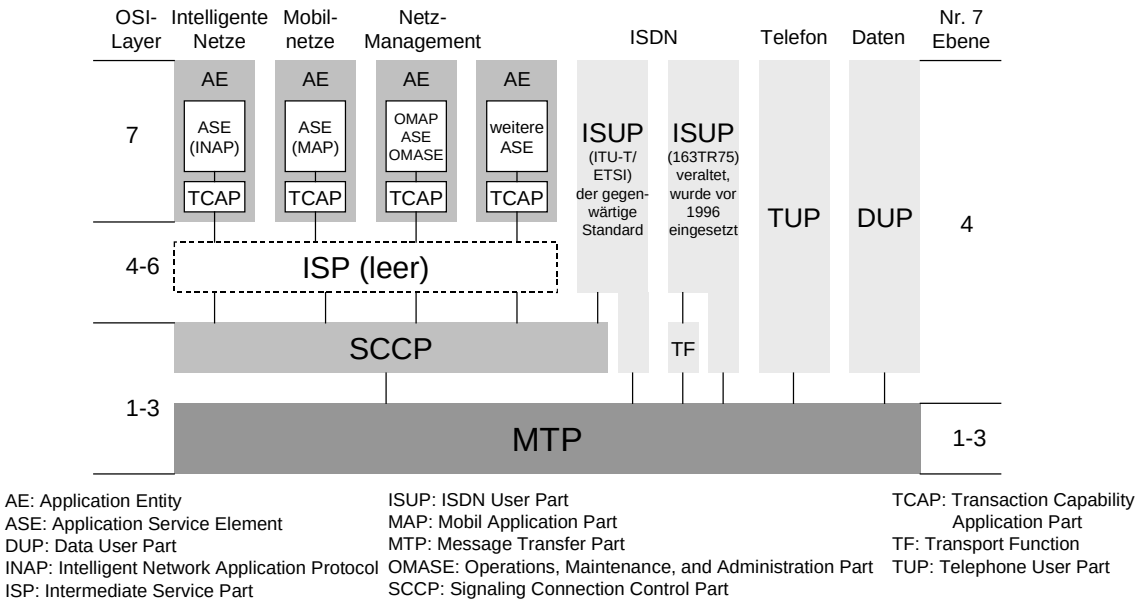


Abbildung 4: Komponenten des ZGS Nr. 7

  * Netzmanagement
  * Nachrichtenverteilung
  * Nachrichtenleitweglenkung

- SCCP (Signaling Connection Control Part): Der SCCP wird benötigt, da der MTP nicht alle Funktionen der OSI-Netzschicht beinhaltet. Er gehört daher noch zur OSI-Schicht 3, obwohl er Ebene 4 angehört.
- TUP (Telefon User Part): Der TUP ist besonders für den Einsatz von Fernsprechanwendungen (vor ISDN) ausgelegt, das heißt für die Übermittlung der Informationen, die in einem Fernsprechnetz zur Steuerung der Verbindungsleitungen zwischen der Vermittlungsstellen benötigt werden.
- DUP (Data User Part): Der DUP wurde im deutschen Netz nie unterstützt, der Einsatz ist auch nicht geplant.
- ISUP (ISDN User Part): Ähnlich dem TUP, jedoch mit der Möglichkeit, Informationen über unterschiedliche Dienste zu übertragen, um die ISDN-typischen Leistungsmerkmale zu erbringen.
- TCAP (Transaction Capabilities Application Part): Komplexere Anwendungen verwenden den TCAP, welcher räumlich verteilte Transaktionen unterstützt (z.B. Datenbankabfragen).
- INAP (Intelligent Network Application Protocol): Dieses Protokoll ermöglicht Zusatzdienste im Bereich des TCAPs, z.B. Zugriff auf Funktionen der „Intelligenten Netze“.

# 4 Kopplung des ZGS Nr. 7 mit IP-Netzen

## 4.1 Funktionale Anforderungen und Definitionen

In diesem Abschnitt wird die wesentliche Struktur und Umgebung für die Realisierung des Transportes von Signalisierungsnachrichten über IP-Netzwerke vorgestellt. Wichtig sind die

Beziehungen zwischen den physikalischen und funktionalen Einheiten, welche Signalisierungsnachrichten austauschen, wie z.B. Media Gateways und Signaling Gateways, und wie der Transport genutzt werden kann [OnRy99].

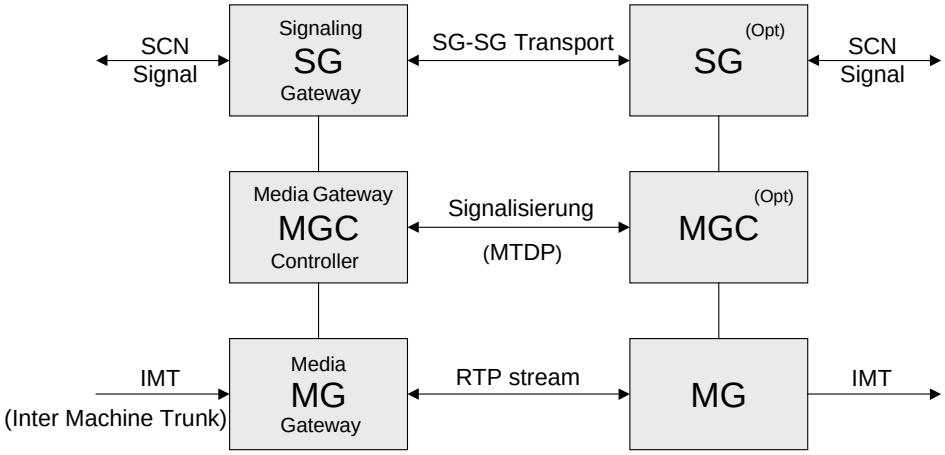


Abbildung 5: Allgemeine Struktur der Gateways

Der Begriff SCN (Switched Circuit Network) beschreibt die öffentlichen, leitungsvermittelten Telefonnetze mit Signalisierungsnachrichten, z.B. ZGS-Nr. 7. Diese Nachrichten werden im Signaling Gateway (SG) teilweise terminiert, übersetzt und in das IP-Netz weitergeleitet oder aus diesem empfangen. Das Media Gateway (MG) verwandelt den SCN-Datenstrom (PCM-codierte Nutzkanladaten) in einen paketorientierten und liefert diesen in das paketorientierte Datennetz aus oder in umgekehrter Reihenfolge aus dem Datennetz in das öffentliche Telefonnetz. Die Steuernachrichten der Gateways, also des MGs oder SGs, werden über den Media Gateway Controller (MGC)gesendet. Dieser behandelt die Registrierung und das Management der Gateways. Der MGC muß die Daten für die lokalen Telefonnetzwerke autorisieren. Für den Signalisierungstransport zwischen den MGCs wird das MTDP-Protokoll vorgeschlagen, auf das später noch eingegangen wird. Die Kommunikation zwischen den MGs kann mittels RTP-Streams betrieben werden. RTP/Streams enthalten die Nutzkanaldaten. Ein STG ist ein SG welcher Signalisierungsinformationen über ein unterliegendes Netzwerk transportiert, z.B. ISUP über IP anstelle von ISUP über ZGS-Nr.7-Ebenen. Ein SG stellt also die STG-Funktionalität zur Verfügung.

Der Transport von Signalisierungsnachrichten über IP bedarf eines transparenten nachrichtenbasierten Signalisierungsprotokolls. Dieses muß Kapselungsmethoden, Ende-zu-Ende Mechanismen und Gewährleistungen für funktionale Ansprüche sowie ausreichende Leistung bereitstellen. Außerhalb dieses Transport-Protokolls liegen Rufzuweisung und Rufzuweisungsumwandlung, auf die hier nicht näher eingegangen wird.

In Abbildung 6 sind Implementierungsbeispiele aufgezeigt, wie die Funktionalität zwischen ZGS Nr.7- und IP-Netzwerken genutzt werden kann. Für die Kopplung mit einem ZGS Nr.7-Netzwerk terminiert der SG den ZGS Nr.7-Link und transportiert die Informationen zum MGC. Das MG hingegen terminiert den IMT und kontrolliert diesen anhand der Informationen vom MGC (a). Das SG und das MG können auch in einer Einheit untergebracht werden (b).

In Abbildung 7 ist der alternative Fall aufgezeigt, in dem der ZGS Nr. 7 Link in derselben Einheit terminiert wird wie der IMT. Hier ist die SG- und die MG-Funktionalität in der selben Einheit untergebracht. Für die Rufsteurerung besteht ein „Backhaul“ der Signalisierungsnachricht mit dem MGC. Die MTP-Ebene 1 und 2 werden hier terminiert und die restlichen Signalisierungsnachrichten an den MGC gesendet. Da die Gateways diese Nachrichten nicht weiterverarbeiten müssen, wird damit eine höhere Skalierbarkeit erreicht. Dieses

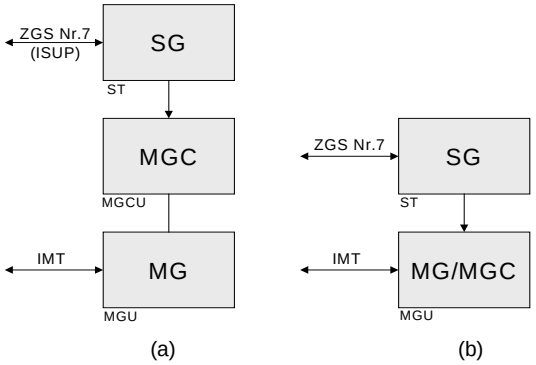


Abbildung 6: Implementierungsbeispiel (a) und (b)

„Backhauling“ wird auch für bestimmte Rufsteuerungs- bzw. Ruflenkungsfunktionen, welche das MGC zur Verfügung stellen muß, benötigt.

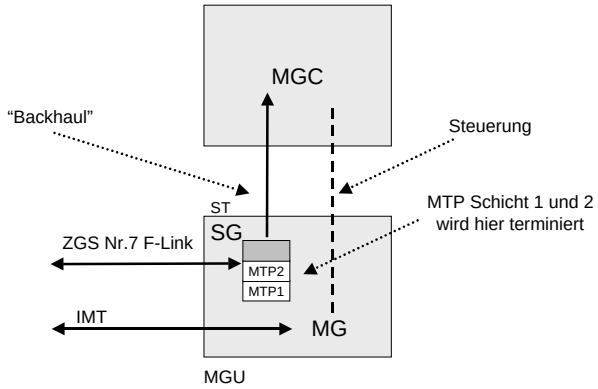


Abbildung 7: Implementierungsbeispiel (c)

Abbildung 7 beschreibt exemplarisch die Protokollarchitektur eines ZGS-Nr.7 Zugriffs auf ein IP-Netzwerk. Der Signaling End Point überträgt Signalisierungsnachrichten über einen Signaling Transfer Point zum Signaling Gatway, in dem die Signalisierungsnachrichen auf IP umgesetzt werden. Das SG schickt nun die Nachrichten in das IP-Netzwerk auf der Basis von RUDP bzw. IP zum Internet Signaling End Point. Die SCCP-Protokollebene ist optional, d.h. wenn die SCCP-Funktionalität beim jeweiligen Transport nicht benötigt wird, kann diese entfallen.

Generell muß der Transport von Signalisierungsnachrichten über IP verschiedene funktionale Anforderungen erfüllen. Er muß verschiedenste ZGS Nr. 7 Protokolltypen (z.B. ISUP, SCCP, MAP, ...) unterstützen und ein Basisprotokoll für Header-Formate und Sicherheitserweiterungen bereitstellen. In der Transportebene sind Mechanismen für die Flußkontrolle, sequentielle Auslieferung, Übertragungswiederholung sowie Informationen über Unerreichbarkeit der Gegenstelle gefordert. Eine Fähigkeit zum Multiplexen höherer Ebenen ist für eine bessere Skalierbarkeit hilfreich. Die existierenden Sicherheitsmechanismen im IP-Netzwerk müssen den Transport gewähren (z.B. durch Proxys oder Firewalls). Die Umgebung muß Knoten mit hohem Verkehrsaufkommen unterstützen sowie eine Möglichkeit besitzen, auf Verkehrsstau im Netz zu reagieren.

## 4.2 Backhaul

Das Backhaul [AuBM99b] (siehe auch Abbildung 7) besteht zwischen einem Media Gateway oder Signaling Gateway und einem Media Gateway Controller mit Rufverarbeitung („Call

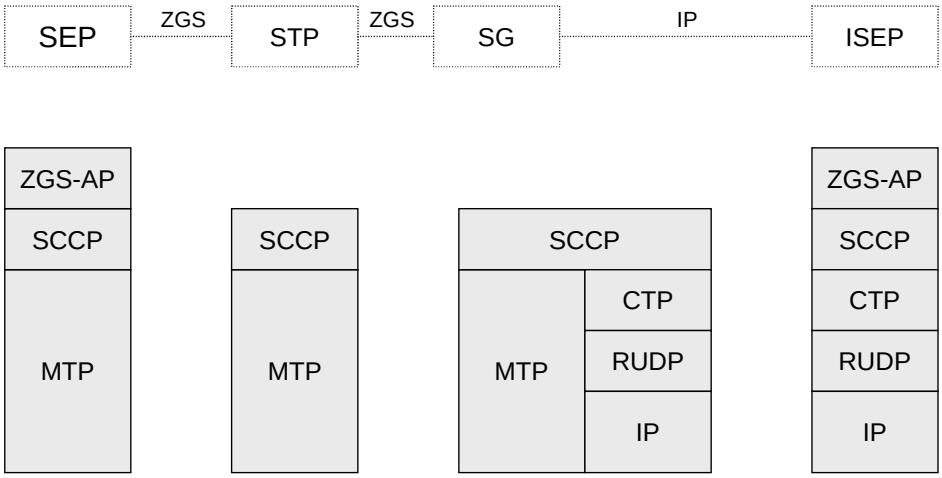


Abbildung 8: Protokollarchitektur des ZGS-Nr. 7

Processing“). Dieses bildet eine Schnittstelle zwischen dem öffentlichen Telefonnetz (PSTN) und dem Paket-orientierten Netz (IP/ATM). Es wird außerdem als „Backhaul“ bezeichnet, weil das Gateway die unteren Schichten des Protokolls terminiert (Ebene 1 und 2) und die anderen Ebenen zum MGC weiterleitet. Dies ermöglicht eine größere Skalierbarkeit, da die Protokollverarbeitung verteilt werden kann, d.h. das Gateway muß die Signalisierungsnachrichten nicht selbst verarbeiten.

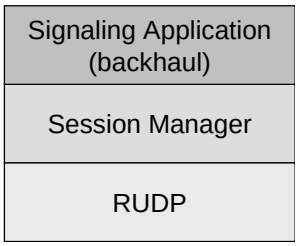


Abbildung 9: Schichteneinordnung

„Backhauling“ benötigt für den Transport von Signalisierungsnachrichten ein schnelles und zuverlässiges Protokoll. Vorgeschlagen wird eine UDP-Erweiterung (RUDP Reliable UDP) über IP. RUDP bietet eine gute Voraussetzung für die Kopplung sowie autonome Benachrichtigungen über Verbindungen (Connect/Disconnect). Bei redundanten Netzwerken bzw. redundanten MGC-Konfigurationen kann ein Session Manager Protokoll eine weitere Stufe der Zuverlässigkeit bieten. Der Session Manager bewerkstelligt dies transparent. Die Abbildung 9 beschreibt die Einordnung dieser Größen in die Ebenen. Das Backhaul-Protokoll ist jedoch nicht von dem Session Manager und RUDP abhängig. Letztere sind auch nur „Empfehlungen“ der Sigtran-Group.

## 4.3 Session Manager

Ein Session Manager [AuBM99a] gewährleistet transparente Netzwerkredundanz und eine redundante Konfiguration der Media Gateway Controller für Anwendungen, die Signalisierungsnachrichten über paket-orientierte Netzwerke transportieren bzw. für den Übergang vom ZGS auf IP. Diese Gewährleistungen sind für kommerzielle Anwendungen wesentlich, welche eine höchstmögliche Zuverlässigkeit fordern. Es ist eine logische Entsprechung von mehreren ZGS-Nr. 7-Linksets im Signalisierungsnetz.

Zwei grundlegende Funktionen werden von einem Session Manager bereitgestellt:

- das Management von „Sessions“ in einem redundanten Netzwerk, sowie
- das Management von „Sessions Sets“ in einer redundanten MGC-Konfiguration

Diese Funktionen können für eine oder mehrere Anwendungen bereitgestellt werden.

Eine „Session“ ist durch eine lokale sowie eine entfernte IP-Adresse mit Port definiert. Dies ist die „physikalische“ Verbindung zwischen einem MGC und einem Gateway (MG oder SG). Eine oder mehrere Sessions bilden eine „Session Group“, falls Netzwerkredundanz gefordert wird. Diese wird durch den Session Manager verwaltet. Wird eine redundante MGC-Konfiguration benötigt, wird ein „Session Set“ gebraucht. Dieses besteht aus mehreren Session Groups.

Es folgen Beispiele für die unterschiedliche Funktionalität: In der in Abbildung 10 dargestellten Konfiguration existieren ein ZGS-Nr. 7 und ein ISDN Kanal, welche beide mit derselben Session Group assoziiert sind. Netzwerkredundanz bedeutet hier, daß beide Sessions auf unterschiedlichen IP-Netzwerken realisiert sind.

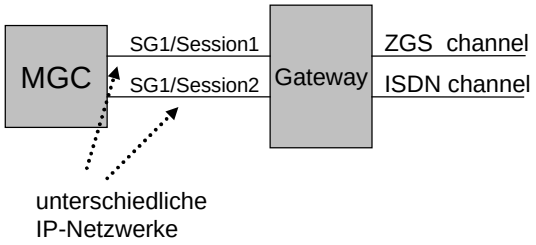


Abbildung 10: Session Manager Beispiel 1

In Abbildung 11 besteht das gleiche Szenario, nur mit dem Unterschied, das der ZGS-Nr. 7 und ISDN Kanal über unterschiedliche Session Groups verbunden sind. So kann z.B. der Signalisierungsverkehr getrennt werden. Natürlich können weitere Sessions für die Session Groups hinzugefügt werden, um Netzwerkredundanz zu unterstützen.

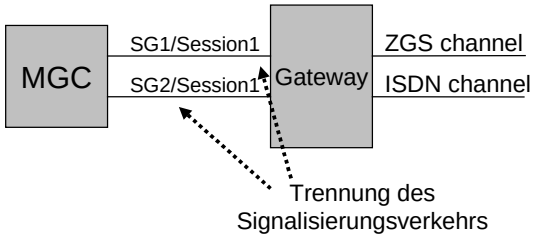


Abbildung 11: Session Manager Beispiel 2

Ein Beispiel für eine redundante MGC-Konfiguration ist in Abbildung 12 dargestellt. In diesem Fall verwaltet der Session Manager eine Active/Standby-Situation. Fällt z.B. ein MGC aus, so kann auf den zweiten umgeschaltet werden. SG1 und SG2 bilden hier zusammen ein Session Set.

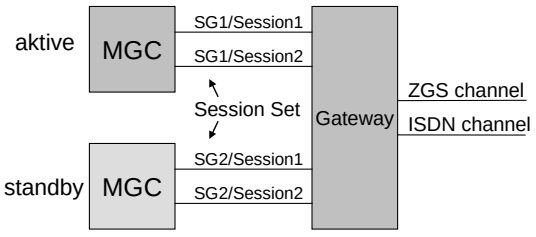


Abbildung 12: Session Manager Beispiel 3

Der Gebrauch von Session Groups und Session Sets ist erwünscht, um mehrere Verbindungen zwischen einem Gateway (Client) und einem MGC (Server) zu unterhalten, weil damit auf Ausfälle oder Verkehrssituationen reagiert werden kann. Dies erhöht die Verfügbarkeit des Gateways.

Wichtige Funktionen der Session Management Ebene sind:

- prioritätsbasiertes Management von Session Groups
- Mechanismen für Session-Ausfall und Umschalt-Situationen (Switchover)
- Mechanismen für redundante MGC-Konfiguration
- einer Anwendung die Kontrolle über den Zustand von möglichen Sessions erlauben
- Mechanismen zur Abfrage von Zuständen
- Allgemeine Monitorfunktionen

Session Manager benötigen zuverlässige, verbindungsorientierte Transportmechanismen in den unteren Ebenen. Vorgeschlagen wird RUDP.

## 4.4 RUDP (zuverlässiges UDP)

RUDP (Reliable UDP) ist ein einfaches paketorientiertes Transportprotokoll (es basiert auf RFC 1151 und 908). Es setzt auf das UDP/IP-Protokoll auf und bindet eine zuverlässige reihenfolgetreue Auslieferung für virtuelle Verbindungen ein [BoKr99]. Durch das flexible Design von RUDP ist dieses für eine Vielzahl von Transportarten geeignet, so auch für den Transport von Signalisierungsnachrichten gemäß dem Telekommunikationsprotokoll. Folgende Kriterien werden von RUDP erfüllt: Es bietet eine zuverlässige Auslieferung bis zu einem Maximum von Wiederholungssendungen und eine Auslieferung in richtiger Reihenfolge, gewährleistet durch Sequenznummern. Der nachrichtenbasierte Transport besitzt keine Staukontrollmechanismen, bietet aber gute Leistung mit wenig Overhead. Jede virtuelle Verbindung sollte charakteristische Konfigurationsmöglichkeiten besitzen. RUDP beinhaltet eine Fehlererkennung durch die selbe Prüfsummen wie in UDP oder TCP. Der „IPsec“-Standard erlaubt eine sichere Übermittlung (Security) der Daten in RUDP.

Um die Kommunikation zwischen den MGCs zu betreiben, schlägt die Arbeitsgruppe Sigtran das MDTP-Protokoll vor, auf welches nun näher eingegangen wird.

## 4.5 MDTP (Multi-Network Datagram Protocol)

MDTP [AuBM99b] ist ein experimentelles Protokoll zur Unterstützung von speziellen Anwendungen, z.B. der Signalisierung mit Rufsteuerung und Rufweglenkung. Es bietet einen fehlertoleranten gesicherten bzw. ungesicherten Datentransfer zwischen Kommunikationsprozessen über einem IP-Netzwerk. MDTP ist im Hinblick auf besondere Unterstützung für redundante Netzwerke und gutes Fehlermanagment entwickelt worden. Hoher Wert wurde auch auf Zeitkontrolle, d.h. auf die rechtzeitige Auslieferung von Paketen, und einer großen Konfigurationsflexibilität durch die Anwendung gelegt. MDTP ist als Anwendungsbibliothek implementiert worden und nicht als Betriebssystemfunktionalität. Der Grund für die Entwicklung von MDTP war, eine Umgebung für zuverlässige und echtzeitkritische kommerzielle Anwendungen zu schaffen (z.B. ein Rufsteuerungs-Protokoll für Internettelefonie).

Wichtige funktionale Leistungen von MDTP sind:

- Ein Prozeß muß in der Lage sein, gleichzeitig mit sehr vielen Endpunkten zu kommunizieren, welche Anwendungsendpunkte unterschiedlichster Art sein können (ZGS Nr. 7, IP, Mobil Network).
- Ein Prozeß benötigt eine sehr genaue Kontrolle über die zeitliche Auslieferung von Datagrammen, da für die Telefonie die rechtzeitige Auslieferung notwendig ist.
- Das redundante Netzwerk und die Fehlerkontrolle sollte möglichst transparent für die höheren Anwendung unterstützt werden.
- Das Protokoll sollte in der Lage, sein mit Datagrammen, welche nicht in richtiger Reihenfolge oder doppelt ausgeliefert wurden, umzugehen (was in redundanten Netzwerken häufig auftreten kann).

MDTP bietet ein 2-Wege-Verkehrsaufbau. Der gesicherte Transportmodus wird mit Hilfe von Piggyback Acknowledgment, diversen Timern, Gap Acknowledgment (selektiven Übertragungswiederholung) und Staukontrolle unterstützt. Nachrichten können mit Hilfe des Protokolls recht einfach fragmentiert werden. MDTP ist Multinetzwerkfähig, d.h. wenn mehr als ein Weg zwischen zwei Endpunkten existiert, kann MDTP, z.B. bei großen Verzögerungszeiten oder Unerreichbarkeit, zwischen diesen Wegen umschalten. Dies ist eine große Erleichterung für die gleichmäßige Verteilung der Verkehrslast der Anwendung oder der höheren Schichten. Um mit Datagrammen umzugehen, welche nicht in richtiger Reihenfolge bzw. doppelt ausgeliefert wurden, besitzt MDTP einen sehr effizienten Quittierungsmechanismus. MDTP ist im Gegensatz zu TCP für spezielle Anwendungen (fehlertolerante, zeitkritische, Redundanz unterstützende) optimiert. Es ist nicht entwickelt worden, um TCP generell abzulösen. MDTP unterstützt auch Multicast/Broadcast Datagramme, falls dies von den unteren Schichten angeboten wird.

## 5 Zusammenfassung

Die IETF Working Group Sigtran beschäftigt sich mit Bereitstellung einer Basis für die Kopplung von IP-Netzen mit öffentlichen Telefonnetzen. Schwerpunkt ist die Funktionalität und Kommunikation der „physikalischen“ Einheiten, der Gateways, welche Signalisierungsnachrichten umsetzen und austauschen. Die hohen Anforderungen der echtzeitkritischen Anwendungen bedürfen einer Vielzahl von schnellen Mechanismen und Protokollen. Eine Erfüllung der Fehlertoleranzanforderungen wird z.B. durch Unterstützung redundanter Verbindungen gewährleistet. Auf die Handhabung der eigentlichen Nutzkanalinformationen wird nicht näher eingegangen.

## Literatur


[AuBM99a] D. Auerbach, D. Berg und K. Morneault. Session Manager. Internet-Draft draft-ietf-sigtran-session-mgr-00.txt, Februar 1999. Arbeitsdokument der IETF Working Group SigTran.

[AuBM99b] D. Auerbach, D. Berg und K. Morneault. Signaling Backhaul Protocol. Internet-Draft draft-ietf-sigtran-signaling-backhaul-00.txt, Februar 1999. Arbeitsdokument der IETF Working Group SigTran.

[BGGH$^+$95] G. Bandow, H. Gottschalk, D. Gehrmann, W. Hlavac, H. Koch, W. Müller und D. Schwetje. *Zeichengabesysteme*. L.T.U. Vertriebsgesellschaft. 1995.

[BoKr99] T. Bova und T. Krivoruchka. Reliable UDP Protocol. Internet-Draft draft-ietf-sigtran-reliable-udp-00.txt, Februar 1999. Arbeitsdokument der IETF Working Group SigTran.

[OnRy99] L. Ong und I. Rytina. Architectural Framework for Signaling Transport, Februar 1999. Arbeitsdokument der IETF Working Group SigTran.

[Sieg99] G. Siegmund. *Technik der Netze*. Hüthig Verlag. 1999.