Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Pascal Kurtansky, Burkhard Stiller (Edt.)

PPS-Seminar: Grundlagen der Internet-Technologie 6

TIK-Report
Nr. 162, February 2003

Institut für Technische Informatik und Kommunikationsnetze
Computer Engineering and Networks Laboratory

Pascal Kurtansky, Burkhard Stiller (Edt.):
PPS-Seminar: Grundlagen der Internet-Technologie 5
February 2002
Version 1
TIK-Report Nr. 162

Computer Engineering and Networks Laboratory,
Swiss Federal Institute of Technology (ETH) Zurich

Institut für Technische Informatik und Kommunikationsnetze,
Eidgenössische Technische Hochschule Zürich

Gloriastrasse 35, ETH-Zentrum, CH-8092 Zürich, Switzerland

PPS-Seminar
Grundlagen der Internet-Technologie 6

Einleitung

Diese nun bereits sechste Auflage des PPS-Seminares sprach wiederum Studierende des Departements für Informationstechnologie und Elektrotechnik an, welche die Grundlagen und erste wichtige Begriffe des Internet erlernen möchten.
Das Seminar vermittelte dabei die wesentlichen Grundlagen für die Kommunikationstechnologie des Internet. Dabei wurden u.a. die folgenden Fragen aufgeworfen und Antworten hierzu gegeben: Was ist ein Netzwerk, was bezweckt die Adressierung, wie funktioniert E-Mail, welche Protokolle gibt es im WWW, wie werden drahtlose Web-Zugriffe möglich, was ist Mobile IP? Das Seminar vertiefte entsprechende Details der Internet-Technologien: Was ist die Internet-Architektur, welche Protokolle gibt es, welche Rolle spielt die nächste Generation der Internet-Protokolle, welche Entwicklungstendenzen zeigen sich? Insbesondere wurden einige Themen behandelt, die mit dem Auftritt des Internet als Daten- und Informationspräsentationsmedium zusammenhängen, u.a. das HTTP-Protokoll, sowie die Datenstrukturierungssprache XML.

Ablauf

Die Studierenden erarbeiteten wie in den vergangenen Semestern dieses Mal zu 10 vorgegebenen Themen (siehe unten) eigenverantwortliche, schriftliche Zusammenfassungen, die in diesem TIK-Report zusammengestellt sind. Diese Ausarbeitungen basieren auf teilweise bereitgestelltem Material sowie Literatur, die die Studierenden aus eigenem Antrieb ermittelt und erarbeitet haben. Erstmalig wurde den Studierenden dieses Semester ein zu verwendendes Word-Template abgegeben, um die Form der Ausarbeitungen optisch ansprechend zu gestalten und zu vereinheitlichen.

Neben dieser schriftlichen Arbeit, musste jeder Studierende einen Vortrag von genau 15 Minuten halten – mit einer maximalen Überzeit von zwei Minuten. Die Studierenden sollten lernen, in dieser Zeit den technischen Sachverhalt zusammenzufassen, das Essentielle herauszuschälen und anschliessend prägnant zu präsentieren. Ein nachfolgende kurze Diskussions- und Fragephase erlaubte das interaktive Behandeln von Unklarheiten, offenen Fragen sowie die Verknüpfung von den verschiedenen Themen.

Vorträge, Referenten und Titel

Vortrag	1	Eveline Minder	Grundlagen des Internet
Vortrag	2	Fabian Henzmann	IP, Adressierung und Routing im Internet
Vortrag	3	Peter Laudanski	IPng – Die nächste Generation des Internet Protokolls
Vortrag	4	Fabian Bläsi	MobileIP
Vortrag	5	Thomas Hug	TCP/UDP
Vortrag	6	Roman Suter	Das HTTP-Protokoll
Vortrag	7	René Kamer	Die Datenstrukturierungssprache XML
Vortrag	8	Emmanuel Corboz	Sichere Kommunikation – SSL, SHTTP
Vortrag	9	Patrick Moser	Elektronische Post im Internet
Vortrag	10	Patrick Fauquex	WAP und WML

PPS-Seminar
Grundlagen der Internet-Technologie, WS 02/03

Grundlagen des Internet

Eveline Minder
mindere@ee.ethz.ch
02. November 2002

1 Einleitung

Wann und wie hat das eigentlich alles angefangen? Über das Alter des Internets lässt sich streiten, Tatsache ist, dass seine heutige Bedeutung, trozt der anfänglichen relativ kleinen Erwartungen, umso grösser sind. Die ganz ersten Handlungen in Richtung Vernetzung von Computern geschahen im 1969: damals wurde erstmals ein Computer an einen Interface Message Processor (IPM),einen Spezialrechner, angeschlossen. Auch in diesem Jahr wurde zum ersten Mal ein Login-Befehl abgesetzt. Damit konnten aber nur Daten zwischen angeschlossen Rechner ausgetauscht werden. Der Anstoss zur Konstruktion der ganzen Netzwerktechnik gab Bob Taylor, der sich über die Tatsache ärgerte, dass er drei verschiedene Terminals brauchte, um mit drei verschiedenen Universitäten zu kommunizieren. Nach knapp sechs Jahren wurde dann das Projekt in die Tat umgesetzt.

Anfang der siebziger Jahre kam die Idee auf, dass die IMPs von Computern abgelüst werden könnten, die keine Spezialrechner waren. 1972 beschäftigte sich Metcalfe damit, zwei separate Netzwerke miteinander zu verbinden. Dabei erfand er die Übertragungstechnik, die er Ethernet nannte. Daraus folgte 1974 den Vorschlag für ein einheitliches Rechnerprotokoll – das TCP/IP. Hiermit konnte das erste Mal von einer wirklichen Vernetzung die Rede sein. Am 1.1.1983 wurde dieses Protokoll in den Rang eines offiziellen Standards erhoben: viele Netzwerker halten dieses Datum für den offiziellen Geburtstag des Internet.

2 Terminologie

Das Internet ist in aller Munde. Wörter wie "Surfen" oder "Homepage" sind jedem bekannt. Manchmal begegnen einem aber auch Wörter, bei denen man nur so gefühlsmässig weiss, was sie bedeuten. Hier ein paar Grundbegriffe mit ihrer Bedeutung.

2.1 Erklärung der wichtigsten Begriffe

Benutzer: ein menschliches Wesen, das mit Hilfe eines bestimmten Programms mit dem Web interagiert.

Browser: ein Programm, das verwendet wird , um auf Web-Server zuzugreifen und von dort heruntergeladene Dokumente darzustellen. Z.B Netscape Navigator und Microsoft Internet Explorer.

Client: Programme, die Web-Server kontaktieren und Daten anfordern. Z.B. Suchmaschinen oder Browser

Server: Der Begriff Sever bezieht sich im Web-Kontext auf einen Prozess auf einem Rechner, der die Funktionalität bereitstellt, auf Anfragen von Clients zu antworten.

Hypermedia: während Multimedia versucht, verschiedene Arten von Medien in einem Dokument zusammenzufassen (Text, Graphiken und Bilder), ist es das Ziel einer weiteren Entwicklung in der Geschichte der Dokumentmodelle, die Anornung der Informationen zu verallgemeinern. Z.B. in einem Buch ist die Reihenfolge des Inhalts festgelegt – mit dem Konzept des Hypertext und später Hypermedia will man sich von der starren sequentiellen Anordnung traditioneller Medien lösen. Man erhält Dokumente, wo die Informationsstücke miteinander verwoben sind (z.B. Homepages).

Internet: das ist die Gesamtheit aller vernetzten Computer, die das Paket der Internet-Protokolle als oberste Schicht ihrer Netzwerksysteme benutzen, d.h. die technische Verbindung aller Computer, die Informationen, welche auf eine bestimmte Art codiert sind, miteinander austauschen. Die Sammlung der Internet-Protokolle implementiert ein paketbasiertes Wide Area Netzwork, (WAN) das in der Lage ist, Netzwerke mit unterschiedlichen Protokollen und sehr verschiedenen Verbindungscharakteristika miteinander zu verbinden. D.h. in den einzelnen Netzwerken werden jeweils andere Arten von Codierungen gebraucht und das WAN, welches diese verbindet, muss sie verarbeiten können.

World Wide Web: Das ist ein verteiltes Hypermedia-System, das auf einigen der Dienste, die das Internet bereitstellt, aufsetzt. Die wichtigsten sind der Benennungsdienst, den das Domain Name System (DNS) bereitstellt, und der recht verlässliche, verbindungsorientierte Übertragungsdienst, das Transmission Control Protocol (TCP). D.h.es ist sozusagen eine Art, wie man das Internet gebrauchen kann.

URL: der Uniform Resource Locator benennt ein Dokument und macht somit möglich, dass bestimmte Informationen identifiziert werden können.

HTTP: der Mechanismus um die gefundene Information herbeizuschaffen.

2.2 Internetumgebung

Eine Verbindung zum Internet kann auf zwei prinzipiell verschiedene Arten erfogen: mit Einwählverbindungen und mit permanenten Verbindungen.

2.2.1 Einwählverbindungen

Wenn ein Computer nur als Client fungiert, ist es nicht nötig, dass er ständig mit dem Internet verbunden ist. Es reicht aus, die Verbindung nur dann herzustellen, wenn ein Server im Internet kontaktiert werden soll. Dies geschieht normalerweise via Modemverbindung, die immer dann adtiviert wird, wenn eine Internet-Verbindung benötigt wird.

2.2.2 Permanente Verbindung

Wenn ein Computer Serverfunktionalität hat, muss er eine ständige Verbindung zum Internet bekommen, da es normalerweise nicht absehbar ist, wann auf einen Sever zugegriffen wird. Einen ständigen Anschluss via Modem und Telefonleitung ist jedoch meistens zu teuer. Man verwendet hierfür Standleitungen.

3 Namen und Adressen

Um Internet-Daten zu transportieren, benötigt man nicht nur eine Verbindung, sondern auch eine Adresse. Folgend werden die Arten von Adressen vorgestellt und die Hierarchie bzw. die Ordnung innerhalb der Namen beschrieben.

3.1 IP-Adressen und DNS-Namen

Eine IP-Adresse ist eine 32-Bit-Zahl, mit der man einen Computer global eindeutig bezeichnt. Im Zusammenhang mit dem Web sind allerdings Computernamen (wie z.B. www.w3.org) wesentlich gebräuchlicher als IP-Adressen. Namen im Internet sind sogenannte DNS-Namen. Das Domain Name System (DNS) ist ein globaler Naming Service, der DNS-Namen auf IP-Adressen abbildet. D.h. das DNS ordnet jedem Namen seine IP-Adresse zu.

Der Grund für diese Zuordnung, sind zwei wichtige Vorteile des DNS. Der erste ist, dass man sich www.w3.org besser merken kann als 18.23.0.22 und der zweite, dass man somit eine höhere Abstraktionsstufe hat. D.h. wenn eventuell einmal die IP-Adresse geändert werden muss, kann dies geschehen ohne den DNS-Namen ändern zu müssen.

Die Domains der obersten Ebene der Hierarchieebene heissen Top-Level Domains (TDL), und jede TDL ist entweder eine Country-Code TDL (ccTDL) oder einer Generic TDL (gTDL).Domain-Namen werden von rechts nach links notiert, so dass der oberste Level ganz rechts steht. Z.B. bei dem Domain-Name www.ethz.ch beschreibt das *.ch* die oberste Domain, das *ethz* die Domain zweiter Stufe. Der Name ganz links bezeichnet einen Computer, der in diesem Fall den Namen *www* trägt, was nach allgemeiner Übereinkunft einen Rechner angibt, auf dem der Web-Server für eine Domain läuft.

4 Datenübertragung

Wie schon oben beschrieben, erlaubt die hierarchische Struktur der Domainbenennung eine rationelle Organisation des Netzes. Sie betrifft die Benennung von Dokumenten und sie erlaubt, wie z.B. auch bei den Telefonnummern, von jedem beliebigen Ort aus, jeden beliebigen Ort zu erreichen und die dazu nötige Nummer ohne grossen Aufwand zu finden.

Es gibt aber auch noch die zweite Art von Organisation, welche die Datenübertragung berifft und die eine geschichtete Struktur aufweist. Diese Schichtung ist eines der grundsätzlichen Prinzipien bei der Computerkommunikation, da sie nicht nur die Komplexität der beteiligten Verfahren auf ein leichter zu handhabendes Mass reduziert, sondern auch die Kapselung von Funktionalität und somit die Austauschbarkeit einzelner Schichten ermöglicht. D.h., die verschiedenen Dienste sind voneinander getrennt und können unabhängig voneinander bearbeitet werden. Man kann einzelne Teile verändern und verbessern, ohne alles andere auch anpassen zu müssen.

4.1 Die Schichtung von Information

Sie berifft die Art, wie die Dokumente verschickt werden. Sie beinhaltet das Trennen der Funktionen eines Netzwerks in Gruppen in sog. „Protokollschichten“. D.h. jede Protokollschicht führt seine spezifische Funktion aus und hat nur für diesen Teilaspekt der Datenübertragung die Verantwortung. Eine Protokollschicht kann mehrere Protokolle (oder auch Dienste genannt) haben. Die Gesamtheit der Protokollschichten ergibt das sog. Protokollpaket, welches die Datenübertragung tätigt.

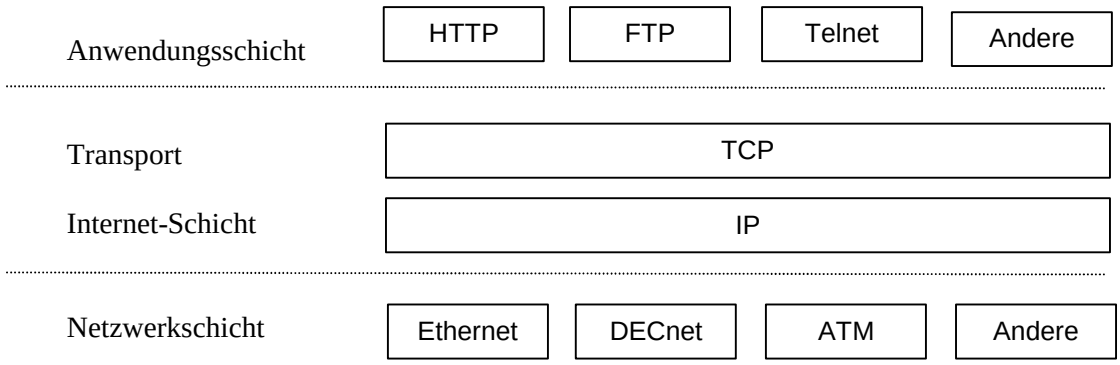


Fig. 1 Schichtung von Kommunikationsprotokollen

4.2 Erläuterung der Schichtung

4.2.1 Zugrundeliegende Netzwerke

Zuunterst liegt das Netzwerk. Diese Protokollschicht hat die Funktion, am Anfang und am Schluss der Datei, die übertragen werden soll, Informationen zu setzen, um den Anfang und das Ende zu markieren. Normalerweise besteht sie aus dem Ethernet oder Token Ring, sog. lokale Netzwerke (LAN). Es kann aber auch ein Wide Area Networking (WAN) sein.

4.2.2 IP und TCP

Die beiden Hauptprotokolle des Pakets der Internet-Protokolle sind das Internet Protocol (IP) und das Transmission Control Protocol (TCP).

Das IP ist ein verbindungsunabhängiges, optimiertes Protokoll zur Paketübermittlung welches das Zerlegen und Zusammensetzen der Pakete übernimmt. D.h. es sorgt dafür, dass die Informationen an das richtige Ort gelangen, Prozesse bekannt als Forwarding und Routing. Um das zu tun, muss das Protokoll die Topologie des Netzwerks verstehen, d.h. es muss wissen, wer mit wem verbunden ist. Obwohl IP die Möglichkeit bereitstellt, verschiedene Netzwerke grenzübergreifend miteinander zu verbinden, mangelt es ihm an der Fähigkeit, einzelne Prozesse auf Rechnern anzusprechen, und es implementiert, d.h. führt nur einen sehr einfachen Datagram Sevice aus. IP-Datagramme können verloren gehen, dupliziert werden und in anderer Reihenfolge als gesendet eintreffen. Da die meisten Anwendungen eine verlässliche Verbindung benötigen, wurde das TCP entworfen. Es ist ein verbindungsorientiertes Protokoll, d.h. es transferiert nicht einfach unabhängige Informationen, sondern es baut einen Zusammenhang auf, in dem die Informationen übertragen werden. Um dies zu tun, muss das Protokoll Extra-Arbeit verrichten. Es nimmt zuerst Kontakt auf mit dem Adressat und wartet auf eine Rückmeldung, bevor es weiter verhandelt, die Nachricht übermittelt und schlussendlich nochmals auf Bestätigung wartet, bevor abgehängt wird. Damit kann die Datenübertragung überwacht werden.

Das TCP und das IP sind also für ähnliches zuständig, wobei das TCP sozusagen auf dem IP aufbaut. Es hängt an die zu übertragende Datei nochmals eine Schicht von Informationen, welche z.B. Codes enthalten, um Fehler aufzuspüren und um die Datei, falls nötig, wieder zurück zu schicken.

4.2.3 Anwendungsprotokolle

Zusätzlich zu diesen Kommunikationsprotokollen IP und TCP gibt es auf einer Ebene weiter oben die sogenannten Anwendungsprotokolle. Die meisten Server implementieren gleich mehrere solche, d.h. führen sie aus. Beispiele hierfür sind HTTP, FTP und DNS. Sie sind zuständig für die Organisation der Informationen, d.h. sie steuern dann schlussendlich die Codierung der Datei selbst und deren Namen. Die ist gewissermassen der Kern des Ganzen Protokollpakets, nämlich der Inhalt, den man z.B. lesen will.

5 Vernetzung von Rechnern

Es existieren zwei verschiedene Rollen, welche ein System im Internet spielen kann: die eines Host und die eines Routers. Ein System, das Nachrichten erhält und sendet ist ein Host, eines das Nachrichten vin einem Netz auf das andere überträgt, ist ein Router. Router vernetzen verschiedene Ethernets und werden durch einen „Dialup Link“ verbunden. Routers sind Verzweigungsstellen, Hosts haben nur eine Verbindung zum Netz.

Routers besitzen normalerweise nur die unteren beiden Protokollschichten – das Ethernet und das Netzwerk. Solange ein Router kein Datenverkehr erzeugt oder aufnimmt, braucht er keine Transport- oder Applikationsprotokolle.

Innerhalb der Vernetzung der Rechner gibt es auch nochmals eine Hierarchie: der Provider steht hier im Zentrum. Die unter sich vernetzten Provider bilden sozusagen das Rückgrat des Internet. Einzelne Providernetzwerke können lokal begrenzt sein, oder sie können ganze Kontinente umfassen.

Am äusseren Rande des Internet befinden sich die „Sites“. Das ist eine Sammlung von Netzwerken, die von einem einzelnen Verwalter kontrolliert werden, z.B. Organisationen.

6 Zusammenfassung und Schlussfolgerung

Das WWW ist nur ein Teil des Internet und baut auf dessen Dienste auf. D.h. es braucht z.B. den DNS Dienst für die Adressierung der verschiedenen Dateien oder das TCP-Protokoll für deren Transport. Das Internet ist grundlegender und bezeichnet einfach die Verbindung von Computern. Das WWW ist sozusagen eine Anwendung des Internet.

In der allgemeinen Informationsübertragung im Internet steht die Schichtung der Kommunikationsprotokolle im Vordergrund. Von der Netzwerkschicht bis zur Anwendungsschicht hat jedes Protokoll seine genaue Aufgabe. Eines ist zuständig für die Abtrennung der Informationen (z.B. das Ethernet), eines für die Verbindung (IP), eines für die Informationstransport (TCP) und eines für die Organisation der Information, d.h. der eigentliche Inhalt einer Datei (z.B. HTTP).

Das Internet besteht aus der Vernetzung verschiedener Teilnetze, die miteinander kommunizieren und in denen es verschiedene Arten von Systemen (oder Rechner) gibt, z.B. sogenannte Hosts oder Routers.

Das Internet wurde ursprünglich für den Informationsaustausch in der Wissenschaft gedacht. Sehr schnell entwickelte es sich aber als Marktanteil in der Wirtschaft. Man kann das Internet als Revolution bezeichnen, allerdings führte sie nicht zu der Weltumspannenden, herrschafftsfreien Kommunikationsgemeinschaft, von der so mancher Freak träumte, der in den 70er Jahre Computer als neues Spielzeug entdeckte. Das Internet ist ein Geschäft: die Parallelen zwischen industrieller Revolution und Technologieschub durch das Netz sind auffällig. Träume, das Internet mache Schluss mit dem Standortvorteil der Zentren und hebe soziale Schranken auf, sind auch heute noch wohlfeil. Noch immer gibt es in den grössten Städten, in den meist bewohnten Vierteln am meisten Gebäude, die ans Glasfaserkabelnetz angeschlossen sind.

PPS-Seminar
Grundlagen der Internet-Technologie, WS 02/03

IP, Adressierung & Routing im Internet

Fabian Henzmann
fbi@student.ethz.ch
15. November 2002

1 Einführung

Das Internet ist zu einem Schlagwort der modernen Gesellschaft geworden. Grundsätzlich ist das Internet die Verknüpfung von Rechnern auf der ganzen Welt, welche in der Lage sind, miteinander zu kommunizieren. Doch auf welche Weise finden die Daten Ihren Weg in diesem riesigen Netzwerk von einem Rechner zu einem andern? Was geschieht, wenn gewisse Teile des Netzwerkes ausfallen sollten? Was passiert, wenn einzelne Datenpakete sich verirren oder ganz verloren gehen sollten? Diesen und noch vielen anderen Anforderungen muss das so genannte Internet Protokoll gerecht werden, damit optimale Verbindungsmöglichkeiten in einem Netzwerk gewährleistet werden können. Das Internet Protokoll wird demzufolge von allen Rechnern des Internets verstanden. Im Zusammenhang mit dem Internet Protokoll werden häufig ICMP und TCP genannt. Diese sollen jedoch nicht Bestandteil dieses Vortrages sein und werden so im Folgenden nur kurz erläutert.

1.1 Internet Control Message Protokoll (ICMP)

Das Internet Control Message Protokoll transportiert allfällige Fehlermeldungen und Diagnoseinformationen für das Internet Protokoll. Dieses Protokoll kann zum Beispiel bei einem Verlust des IP Paketes ein erneutes senden der Daten veranlassen.

1.2 Transport Control Protokoll (TCP)

Das Transport Control Protokoll sichert die Korrektheit der gesendeten Daten und kann bei allfälligen Problemen ein erneutes Senden der Daten veranlassen. Im Gegensatz zum Internet Protokoll ist das Transport Control Protokoll verbindungsorientiert. Weitere Informationen sind dem entsprechenden Vortrag zu entnehmen.

2 Das Internet Protokoll (IPv4)

Die Hauptaufgabe des Internet Protokolls ist, Datenpakete in verschiedene Netzwerke zu verteilen, so dass ein möglichst schneller und zuverlässiger Datenfluss garantiert werden kann. 1974 wurde in einem Artikel von V. Cerf und R. Kahn zum ersten Mal die Grundzüge der TCP/IP Protokolle und der Netzknoten-Architektur umschrieben:

- Unabhängigkeit von der verwendeten Netzwerk-Technologie sowie der Architektur der Hostrechner
- Universelle Verbindungsmöglichkeiten im gesamten Netzwerk
- Ende- zu Ende- Quittungen
- Standardisierte Anwendungsprotokolle

Um diesen Grundzügen gerecht zu werden, umfasst das Internet Protokoll im wesentlichen folgende Funktionen:

- Wegwahl
- Lebenszeitkontrolle
- Segmentieren (Fragmentieren) und Reassemblieren
- Adressierung

2.1 Merkmales des Internet Protokoll

- Verbindungsloses Protokoll
- Fragementiert die Pakete falls nötig
- Adressierung durch 32-Bit Internet Adresse
- Maximal 65535 Bytes Paketgrösse
- Ermittelt lediglich eine Kopfprüfsumme, keine Prüfsumme der transportierten Daten
- Nicht ständig benötigte Protokollfelder sind optional
- „Best Effort“ Zustellung

2.2 Der Aufbau des Internet Protokoll (IPv4)

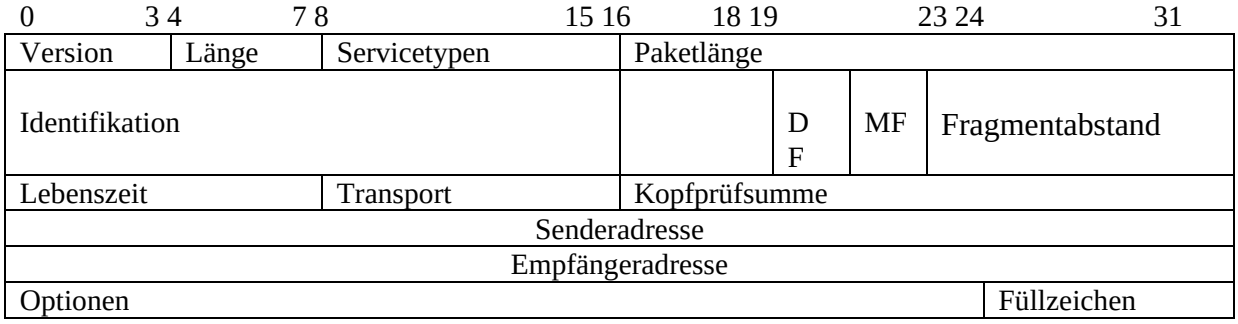


Fig. 1 IP Protokollkopf

2.2.1 Versionsnummer (4 Bits)

Die heute aktuelle Version 4 wird mit 4 Bits dargestellt. (0010)

2.2.2 Länge des Kopfes (4 Bits)

Da infolge von optionalen Optionsfeldern die Grösse des Paketkopfes variieren kann, muss diese ebenfalls angegeben werden um sicherzustellen, dass die darin enthaltenen Angaben stimmen.

2.2.3 Servicetypen (8 Bits)

Die Servicetypen sind Kriterien, wie die Nachricht behandelt wird. (z.B. Priorität). Jedoch wird davon in der Praxis sehr selten gebrauch gemacht.

2.2.4 Paketlänge (16 Bits)

Dieses Feld gibt die Gesamtlänge des IP Paketes an, inklusive des Protokollkopfes. Die Gesamtlänge ist auf 576 Bytes begrenzt. In der Regel sind jedoch die Rechner in der Lage, wesentlich grössere Pakete zu reassemblieren (zusammenzusetzen). Es wird doch nur festgelegt, dass jeder Host in der Lage sein muss, Pakete von mindestens 576 Bytes zu reassemblieren.

2.2.5 Identifikation (16 Bits)

Die Identifikation wird zum korrekten Reassemblieren der einzelnen Datenpakete verwendet. Die Identifikation ist eindeutig und wird z.B. durch einen Zähler am absendenden Host vergeben.

2.2.6 Flags (2 Bits)

Durch diese 2 Bits wird die Vorgehensweise im Falle einer Fragmentierung gesteuert. Das IP Paket darf unter keinen Umständen fragmentiert werden, wenn dass Bit DF „don't fragment" gesetzt ist. (Ist das Datenpaket zu gross zum Weitertransportieren und das Flag auf DF gesetzt, wird es weggeworfen). Das Bit MF „more fragments" zeigt an, ob dem IP Paket noch weitere Teilpakete folgen oder nicht.

2.2.7 Fragmentabstand

Im Falle eines MF Flags, beschreibt der Fragmentabstand die Lage der in diesem Paket enthaltenen Teilnachricht relativ zum Beginn der Gesamtnachricht. Dadurch wird dem empfangenden Host ermöglicht, die Nachricht korrekt zusammenzusetzen.

2.2.8 Lebenszeit: Time To Live (8 Bits)

Dieses Feld dient dazu, verirrte Pakete nach einer gewissen Zeit zu löschen, damit nicht unnötiger Traffic entsteht. Unix Rechner setzen diese Lebenszeit auf einen Wert zwischen 15 und 30. Jeder Netzknoten dekrementiert (verringert) diesen Wert um 1. Somit ist die Lebenszeit TTL gleichbedeutend der maximalen Anzahl von Netzknoten, die vom Paket durchlaufen werden können. Nimmt die Lebenszeit den Wert 0 an, wird das Paket von verarbeitenden Rechner weggeworfen und es wird über ICMP (Internet Control Message Protokoll) eine Nachricht an den Absender des Paketes verschickt.

2.2.9 Transport (8 Bits)

Dieses Feld gibt an, welchem Tansportprotokoll das Paket zugestellt werden muss. Zur Zeit existieren ca. 50 offizielle Protokolle. (Wert 1= ICMP, Wert 6 = TCP, Wert 17 = UDP)

2.2.10 Kopfprüfsumme (16 Bits)

In diesem Feld ist die Prüfsumme der Felder im Protokollkopf enthalten. Somit kann sichergestellt werden, dass die von Host empfangenen Pakete die korrekten Kopfdaten enthalten haben. Die eigentlichen Nutzdaten werden aus Gründen der Effizienz nicht geprüft. Die Internet-Prüfsumme ist bei allen TCP/IP Protocolen wie folgt:
Das 1er Komplement der 16-bit Summe aller 16-Bit-Worte der zu überprüfenden Daten.

2.2.11 Senderadressen und Empfängeradressen

Diese beiden Felder enthalten die 32 Bit lange Internet Adresse des Senders, bzw. des Empfängers.

3 Adressierung

Damit jeder in einem Netzwerk eingebundene Rechner eindeutig identifiziert werden kann, muss ein geeignetes Adressierungssystem vorhanden sein. Die Adressen des (IPv4) sind in einen Präfix, eine Netzadresse und eine Endsystemadresse aufgeteilt. Die 32 Bits, welche der Adressierung zur Verfügung stehen werden in 4 Felder à 8 Bits aufgeteilt. Jedes Feld beschreibt einen 8 Bit Wert zwischen 0 und 255 (dezimal). Die einzelnen Felder werden untereinander mit einem Punkt abgetrennt. Ein Beispiel einer IP Adresse ist z.B. 124.97.30.214

3.1 Adressklassen

Um den unterschiedlichen Grössen von Netzwerken gerecht zu werden, definierte man verschieden Klassen von Netzwerken. Dazu variiert man das Verhältnis der Bits für die Netzwerkkennung und Endsystemkennung.

Klasse	**Max Netzwerke**	**Hosts**	**Bemerkungen**
A-Klasse	126	>16'000'000	Wenige Netze, sehr viele Hosts
B-Klasse	>16'000	65536	Viele Netze mit vielen Hosts
C-Klasse	>2'000’000	254	Sehr viele Netze mit wenig Hosts

Fig. 2 Adressklassen

Klasse A-Adressen sind demzufolge für sehr grosse Netzwerke geeignet. Es können jedoch nur 126 verschiedene Netze unterschieden werden. In den Anfangszeiten des Internets (Arpanet) existierten nur Klasse-A Netzwerke, da man annahm, dass es nur wenige, dafür sehr grosse Netzwerke geben würde. Seit aber in vielen Organisationen lokale Netzwerke eingeführt wurden, stimmt diese Annahme nicht mehr. Aus diesen Gründen wurde beschlossen, weitere Klassentypen einzuführen. So entstanden die B-Klasse und die C-Klasse. Die B-Klasse ist für mittelgrosse Netze und die C-Klasse ist für kleine Netze mit wenigen Hosts. Die Wahl der Klasse richtet sich nach der Anzahl der vorhandenen Hosts und der Netzwerke. Weltweit werden die Netzwerknummern vom Network Information Center (NIC) verteilet. Damit nicht jede Organisation mit über 254 Endsystemen eine B-Klasse Adresse benötig, welche nahezu erschöpft sind, werden stattdessen einzelne zusammenhängende C-Klassen Blöcke vergeben. Diese zusammenhängenden Blöcke werden zu einem Adresseintrag zusammengefasst. Diese Methode nennt man CIDR (Classless Inter-Domain Routing). Die Knappheit an IP Adressen kann so nur aufgeschoben werden und wohl erst mit IPnG (Internet Protokoll New Generation) behoben werden.

3.2 Subnetzwerke

Die Subnetzadressierung stellte eine in der Praxis sehr verbreitete Technik dar, um grosse Netzwerke (A-Klasse Netzwerke und B-Klasse Netzwerke) zu strukturieren. Subnetze sind kleine Unternetzwerke in einem grossen Netzwerk. Es ist auch möglich, Subnetze in Subnetzen zu bilden. Von aussen ist aber das gesamte Netzwerk als eine Einheit sichtbar. Eine solche Aufteilung wird mit Hilfe der so genannten Subnetzmasken realisiert. Die so genannte Subnetzmaske gibt dabei an, welche Bits einer IP-Adresse zur Adressierung eines Subnetzes verwendet werden. Für die Bestimmung der Subnetzmaske geht man wie folgt vor:

- Für jedes gesetzte Bit (1) der Subnetzmaske gilt, dass das entsprechende Bit der Indernet Adresse als Teil des Subnetzwerkes interpretiert wird.
- Jedes gelöschte Bit (0) wird als Teil der Host Adresse interpretiert.

In der Praxis werden dabei meistens die Werte 255 (1111’1111) oder 0 verwendet

.

	1. Byte	2. Byte	3. Byte	4. Byte
Klasse A Netzwerk	Netzwerknummer	Hostnummer		
Netzwerkmaske	255	255	0	0
Subnetzwerk	Netzwerknummer	Subnetzwerknummer	Hostnummer	

Fig. 3 Subnetzwerk eines Klasse A Netzwerkes

4 Routing

In einem Netzwerk ist es möglich, ein Datenpaket über viele Wege zu einem bestimmten Ziel zu versenden, da eine möglichst grosse Redundanz angestrebt wird. Mit Routing bezeichnet man das Verfahren, welches beim Suchen und Auswählen des idealen Weges zur Zieladresse angewendet wird. Die Einträge des Routers können statisch erfolgen, dass heisst meist durch manuelle Eingabe oder dynamisch über den Austausch von Routing-Informationen zwischen den Routern über so genannte Routing-Protokolle. Bei den Routing-Protokollen lassen sich zwei grundsätzliche Verfahren unterscheiden:

4.1 Distanz Vektor Routing

Bei diesem Prinzip des Routing-Protokolls besitzt jeder Router Informationen über seine Entfernung zu jedem benachbarten Router und verteilt seine Routing-Informationen an die benachbarten Router.

4.1.1 Routing Information Protocol (RIP)

RIP ist ein Vertreter der Distanz Vektor Protokollen, welches sehr weit verbreitet ist. Jeder Router sendet in einem zyklischen Abstand von 30 Sekunden seine aktuelle Information über die angeschlossenen Links an die benachbarten Router und teilt diesen mit, wie gross die Distanz des Routers zu andern Netzen ist. Diese inkrementieren einfach die Anzahl der Hops um eins. Die Nachteile von RIP liegen darin, dass das Ausbreiten von Zustandsinformationen über das Netz von der Häufigkeit des Sendens von Routing-Informationen abhängt. Sendet zum Beispiel eine Router alle 30 Sekunden seine Routing-Informationen weiter, so dauert es 7 Minuten, bis der Zustand eines Netzes an ein über 15 Hops entfernten Knoten verbreitet hat. Auch kann mit RIP nur die Hopdistanzen als Routing-Metrik benutzt werden.

4.2 Link-State-Routing

Das Link-State-Routing ist aufwendiger als das Distanz Vektor Routing: Jeder Router besitzt Informationen über jeden andern Link der Domäne und ist so in der Lage, die komplette Netztopologie zu berechnen.

4.2.1 Open Shortest Path First (OSPF)

Im Gegensatz zu RIP ist das OSPF ein Link-State-Routing-Protokoll. Die Router tauschen eine Beschreibung ihrer direkt angeschlossenen Links aus. Dieser Informationsaustausch erfolgt über das so genannte Hello Protokoll und erfolgt in kurzen Abständen < 30 Sekunden. Mit Hilfe dieser Informationen ist jeder Router einen Netzes in der Lage, die gesamte Topographie des Netzwerkers zu berechnen.

4.3 Border Gateway Protokoll (BGP)

Das RIP und das BGP Protokoll werden innerhalb von Domänen benutzt. Um Netze über Domänengrenzen zu verbinden wird das so genannte Border Gateway Protocol von Edge-Routern verwendet. Neben der Hauptaufgabe des BGP, dem Finden des günstigen Weges zwischen zwei Domänen, muss das BGP noch administrative Randbedingungen berücksichtigen. So kann es sein, dass eine gewisse Domäne nicht für Durchgangsverkehr benutzt werden sollte . Zum Durchführen des BGP Protokolls werden die Edge-Router über das stabile Transportprotokollverbindungen TCP miteinander verbunden und tauschen so ihre Routing Informationen aus. BGP gehört zu der Familie der Distanz-Vektor Protokollen, es enthält zusätzlich zu den Distanzen von einem Router zu anderen auch die Information über den kompletten Weg zu einem bestimmten Ziel.

5 Referenzen

www.reto-burger.ch/teko/TCP_IP_Protokolle_bur_v1.html
www.informatik.uni-erlangen.de
www.intsun14.int.uni-saarland.de
www.digital-labs.com

PPS-Seminar
Grundlagen der Internet-Technologie, WS 02/03

IPv6 / IPng
Das nächste IP

Peter Laudanski
laupeter@ee.ethz.ch
22. November 2002

1 IPv6 / IPng – Das nächste IP

Im letzten Vortrag wurde Ihnen die Funktionsweise eines der wichtigsten Bausteine des Internets präsentiert – des IP, das zur Zeit in der Version 4 eingesetzt wird. In diesem Dokument werden Sie erfahren, warum überhaupt ein neues Internet Protokoll spezifiziert wurde und welche Neuerungen es mit sich bringen wird.

1.1 Geschichte und Ziele

Wir alle haben es miterlebt. Spätestens seit den Neunzigerjahren liess sich das sog. Internet-Boom nicht übersehen. Dies v.a. dank der am CERN-Institut entwickelten Basis für das WWW, das die Navigation leicht verständlich gemacht hat. Aus dem ursprünglichen, fast nur der Wissenschaft und dem Militär zugänglichen Werkzeug wurde ein immer breiter angewandtes Medium, das nun auch von der Wirtschaft und den Privatanwendern rege genutzt wird.

Mit dieser Entwicklung kamen auch neue Bedürfnisse, denen das bereits in Jahre gekommene IPv4 nur teilweise oder ungenügend gerecht wird.

Als Antwort darauf wurden schon seit dem Anfang der Neunziger neue Entwicklungen in Gang gesetzt. 1994 wurden die Spezifikationen für das IPv6 festgelegt, von der Internet Engineering Task Force (IETF) empfohlen und von der Internet Engineering Steering Group (IESG) zum Standard vorgeschlagen.

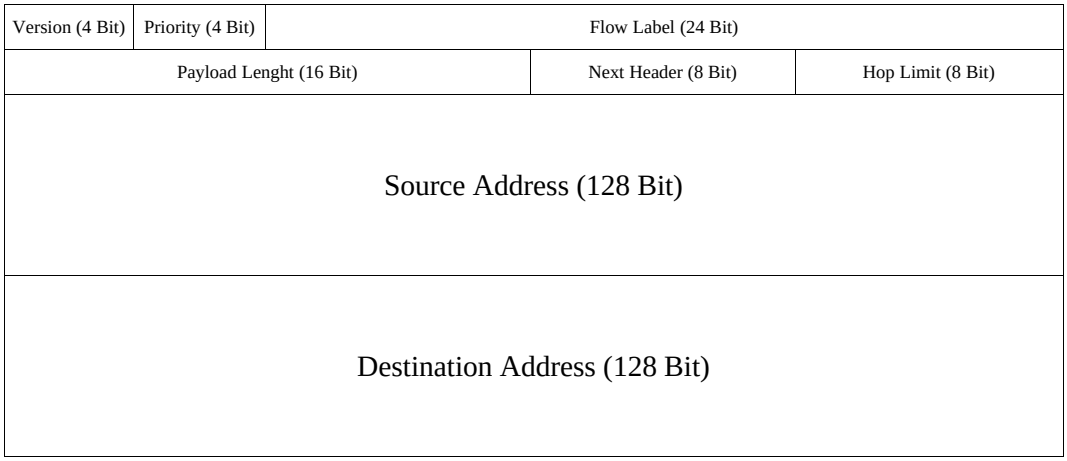
Von den Entwicklern wurde besondere Weitsichtigkeit verlangt und folgende Ziele wurden angestrebt:

- Abwärtskompatibilität zu IPv4, die einen schrittweisen und autonomen Umstieg unter einem vertretbaren Aufwand ermöglicht.
- Addressing und Routing, die dem schnellen Wachstum auch längerfristig gerecht bleiben. Dies beinhaltet sowohl einen genügend grossen Adressraum (IPv4 könnte schon vor dem Ende unserer Dekade nicht mehr genügend Adressen anbieten), der flexibel und übersichtlich administriert werden kann, wie auch Robustheit.
- Effizienter Einsatz in der Vielfalt der Netzwerktypen und Leistungsansprüchen.
- Solide Basis für die wachsenden Ansprüche an die Sicherheit des Datenaustausches.
- Vertretbare Komplexität der Implementierbarkeit.
- Und schliesslich die Kosteneffizienz.

Es wird sofort ersichtlich, dass die Ansprüche sehr vielfältig sind. Ihre möglichst gute Erfüllung soll proprietären Lösungen zuvorkommen und somit ein auf offenen Standards basiertes Internet weiterhin ermöglichen.

1.2 Das Resultat: IPv6 / IPng

1.2.1 Aufbau des allgemeinen Paketformats



Die Datenpakete des IPv6 sind im Minimum aus weniger Feldern aufgebaut als die im IPv4 und somit auch "schlanker". Sie können aber durch die Erweiterungs-Header mit zusätzlichen Informationen / Funktionen skaliert werden.

Im Version-Feld

wird die Versionsnummer des Protokolls übermittelt.

Das Priority-Feld

bildet den ersten Mechanismus der Quality of Service. Hier wird die Priorität des Datenpakets gegenüber anderen Paketen derselben Quelle festgelegt.

Das Flow Label-Feld

bildet den zweiten Mechanismus der Quality of Service. Hier werden Pakete gekennzeichnet, die von den Routern besondere Behandlung brauchen. Zu den wichtigsten Anwendungsbeispielen gehört die Möglichkeit mehrerer (unabhängiger) Flüsse zwischen Quell- und Zieladresse und die Echtzeit-Übertragung.

Im Payload Lenght-Feld

wird die Länge des Datenpakets mitgeteilt. Standardmässig ist diese auf 2^{16} Bytes beschränkt, kann aber mit Hilfe der Jumbo-Payload-Option höher gesetzt werden.

Das Next Header-Feld

identifiziert den Typ des nachfolgenden Headers.

Im Hop Limit-Feld

wird die maximale Anzahl der Forwarding-Schritte festgelegt, bis das Paket zerstört wird.

Das Source Adress-Feld

enthält die Adresse des Absenders.

Das Destination Adress-Feld

enthält die Adresse des Empfängers.

Die 128 Bit Kodierung der Adressen ermöglicht mehr Hierarchie-Stufen und eine einfachere Auto-Konfiguration. Dank der Ausrichtung auf 64 Bit Grenzen wird die Leistung in Routern gespart. Zudem macht sie bis zu $6.65 * 10^{23}$ Adressen pro Quadratmeter Erdoberfläche möglich. Die allgemeine Kodierung hat die Form x:x:x:x:x:x:x:x, wobei jedes x einer Zahl von 0 bis FFFF entspricht. Auf die zulässigen Abkürzungen der Syntax wird hier nicht eingegangen.

1.2.2 Arten der Adressen

Unicast-Adressen

Das Paket wandert vom Sender zu genau einem explizit adressierten Interface. Es wird die 1:1-Kommunikation genannt.

Der Aufbau einer Unicast-Adresse ist folgendermassen konzipiert:

- Globaler / öffentlicher Teil: Präfix zur Bestimmung des Typs und der Struktur der Adresse, Top Level Aggregator (TLA) auf dem Level der öffentlichen Transit-ISPs und Next Level Aggregator (NLA) auf dem Level der untergeordneten ISPs oder mit TISPs verbundenen Firmen
- Lokationsspezifischer Teil / Site Level Aggregator zur Beschreibung der Subnetz-Struktur und die Interface-ID zur Beschreibung des Interface des IP-Systems
- Endsystem-Identifikator

Dank dem Konzept der Standort- und Link-lokalen Adressen ist für viele Aufgaben auch der Einsatz weniger komplexer Adressen möglich. Standort-lokale Adressen sind für grössere Netze (noch) ohne Internet-Anbindung von Bedeutung. Die Link-lokalen Adressen leisten v.a. während den ersten Schritten der automatischen Konfiguration und in Netzen ohne Router ihre Dienste.

Anycast-Adressen

Dieser Adress-Typ lässt sich äusserlich nicht von Unicast unterscheiden und dient der Kennzeichnung von einer Menge Interfaces, die i.A. zu verschiedenen Zwischensystemen gehören. Das Paket erreicht genau ein Interface dieser Menge, in der Regel den gemäss der Routing-Metrik nächstgelegenen.

Multicast-Adressen

Das Paket wandert vom Sender zu einer Multicast-Adresse und kann von allen Mitgliedern der Gruppe empfangen werden. Dieses Konzept kommt bei Gruppenkommunikation zum Einsatz.

Der Aufbau einer Multicast-Adresse ist folgendermassen konzipiert:

- Das Flag-Feld zeigt an, ob es sich um eine transierte Gruppe ohne permanente Adresse oder um eine permanente, der Internet Assigned Numbering Authority (IANA) bekannte Gruppe handelt.
- Das Scope-Feld kodiert den Gültigkeitsbereich / die Reichweite der Gruppe. Die möglichen Skalierungen sind: Knoten, Link, Standort / Organisation und global.

Die Idee der Multicast Adressen verspricht effizientere Bandbreitenausnutzung bei Übertragung an mehr als einen Empfänger.

1.2.3 Die Erweiterungen

Zwischen dem beschriebenen Minimal-Header und den Daten lassen sich bei Bedarf zusätzliche Header platzieren. Hop-by-Hop-Optionen, der erste Destination-Options-Header und der Routing-Header werden noch von den Routern ausgewertet. Die restlichen richten sich an die Endsysteme.

Jede Erweiterung darf nur einmal pro Paket vorkommen. Bei mehreren Erweiterungen ist ihre Reihenfolge und der Verweis auf die Art der folgenden Erweiterung verbindlich.

Hop-by-Hop-Optionen

Hier werden Informationen / Optionen eingefügt, die in jedem Knoten ausgewertet werden müssen.

Destination-Options-Header

Hier befinden sich zusätzliche Angeben über den Zielknoten.

Routing-Header

Hier lässt sich der Weg (z.B. die Wahl der zu passierenden Provider) des Datenpakets beeinflussen.

Fragment-Header

Hier befinden sich die Informationen zur Aufteilung und Zusammenführung der Pakete, falls sie länger sind als die Pfad-MTU (Maximum Transition Unit) eines Teils des Netzes erlaubt. Diese Operationen werden nicht mehr von den Routern, sondern nur noch von den Quell- und Zielmaschinen durchgeführt.

Im Normalfall kommt jedoch die Pfad-MTU-Erkennung zum Einsatz, um Fragmentierung und Reassemblierung zu vermeiden.

Sicherheitsoptionen

Der Authentification-Header bietet den ersten Teil der integrierten Sicherheitsfunktionen. Seine Aufgabe ist die Feststellung, ob die Daten vom Sender tatsächlich gesendet wurden und ob sie beim Empfänger unmanipuliert angekommen sind.

Encapsulation-Security-Payload-Header bietet den zweiten Teil der integrierten Sicherheitsoptionen und beschäftigt sich mit der Verschlüsselung der Nutzlast und der Ende-zu-Ende-Erweiterungsheader im sog. Transport-Modus.

Im Tunnel-Modus wird das gesamte Paket chiffriert und ein neuer unverschlüsselter IP-Header erzeugt.

Im Security-Parameter-Index (SPI) sind die zwischen dem Sender und dem Empfänger vereinbarten Sicherheitsparameter (Schlüssel und Algorithmen für Authentifizierung / Verschlüsselung, Lebenszeit der Schlüssel, Sicherheitsstufe,...) enthalten.

1.2.4 ICMP und RSVP

Auch auf der Maschinenebene selbst hat sich einiges getan.

ICMP steht für Internet Control Message Protocol. Zu seinen Aufgaben gehören die Fehlernachrichten, die Informationsnachrichten (Echos und Gruppenverwaltung) und die Neighbour Discovery (Erkennung angeschlossener Knoten, Router, ihrer Parameter, Unterstützung der automatischen Konfiguration,...).

RSVP steht für Resource Reservation Setup Protocol und dient der Reservierung zahlreicher Ressourcen auf dem Weg und in den Clients.

1.2.5 Technisch-wirtschaftliche Aspekte

Dank der eingebetteter Kompatibilität zu den IPv4 Strukturen steht der Interoperation nichts im Wege, was wichtige Konsequenzen hat. Bis auf die Koordination mit den DNS-Servern können die Updates firmen- / organisationsintern völlig autonom erfolgen. Bis auf gewisse Router können die Maschinen eines Netzwerks ebenfalls unabhängig voneinander umgestellt werden. Bei weniger wichtigen Geräten lässt sich sogar auf ein Update verzichten.

In den meisten Fällen ist sehr wenig bis fast keine Vorbereitung nötig. Da die Maschinen auch nach dem Update ihre bisherigen Adressen beibehalten können, sind aufwendige Pläne für neue Adressenbelegung nicht nötig. Nach dem erfolgten Umstieg können die "Plug & Play" Funktionen des neuen IP ausgespielt werden.

1.3 Schlusswort

Zuerst muss ich etwas vorwegnehmen: Bei genauer Lektüre der Quellen wird man auf eine Reihe von Ausnahmefällen aufmerksam gemacht, die trotz aller Neuerungen nur mit zusätzlichen Eingriffen gelöst werden können.

Unter dem Strich lässt sich jedoch klar sagen, dass IPv6 / IPng die im ersten Teil erwähnten Anforderungen erfüllt und viel "dazugelernt" hat. Die Ingenieure hinter dem Projekt haben eine erstaunliche Arbeit geleistet.

Noch ist der Einsatz in der Praxis kaum sichtbar, da uns IPv4 derzeit noch keine drastischen Engpässe bietet und einige Neuerungen des neuen IP sich für das derzeitige IP implementieren liessen. Aber "unter der Haube" haben die Vorbereitungen auf Hard- und Softwareebene schon kurz nach der Spezifikation begonnen und es ist nur noch eine Frage von wenigen Jahren, bis sich die Umstellung bemerkbar machen wird.

Referenzen

1. Felix von Leitner – "Das nächste Netz", c't 16/2001
2. Robert M. Hinden - "IP Next Generation Overview", 1995
   http://playground.sun.com/pub/ipng/html/INET-IPng-Paper.html

Weiterführende Links

IETF: www.ietf.org
IESG: www.iesg.org
IPv6: www.ipv6.org
IPv6: playground.sun.com/pub/ipng/html/

PPS-Seminar
Grundlagen der Internet-Technologie, WS 02/03

Mobile IP

Fabian Blaesi
blaesif@ee.ethz.ch
22. November 2002

1 Einführung

Das Internet hat sich schnell in der ganzen Welt verbreitet und ermöglicht somit den Datentransfer praktisch überall. Profitieren von diesen Informationsquellen kann man zu Hause, im Büro oder in der Schule. Das Verlangen nach Mobilität macht auch hier keinen Halt, deshalb bemühte man sich um ein Internet Protokoll, das auch unterwegs anwendbar ist. Das Resultat war das Mobile IP, welches jeder Zeit und praktisch überall einen zuverlässigen Internetzugang bietet. Nun ist es möglich das Büro überall hin mitzunehmen, was eine unglaubliche Flexibilität garantiert. Diese Vorteile werden immer mehr genutzt, dies sieht man an der rasanten Zunahme von mobile IP-fähigen Geräten, wie gewisse Mobiltelefone, Laptops oder PDAs.

1.1 Motivation

Daten die von Endsystem zu Endsystem durch das weltweite Internet transportiert werden, werden, basierend auf der IP-Quell- und Zieladresse im Paketkopf durch die Hilfe der Router vermittelt. Jeder Router legt anhand der am besten passenden Netzpräfix fest, an welchen Ausgang das Paket weitergeleitet wird. Der letzte Router vor dem Zielsystem bestimmt das physikalische Subnetz des empfangenden Rechners. Nach der klassischen Adressierung liegt ein Rechner mit der IP-Adresse 129.12.31.145 im physikalischen Subnetz 129.12.31. Wird ein Endsystem durch eine geänderte Funkanbindung mit Hilfe eines Wireless LAN oder durch einfaches Umstecken der Verbindung von einem physikalischen Subnetz in ein anderes gebracht, können ohne Zusatzmassnahmen keine Pakete mehr empfangen werden, da sich der Rechner in einem falschen Subnetz(mit anderem Präfix) befindet. Es gibt 3 Lösungen für dieses Problem:

1.1.1 Wechseln der IP-Adresse

Die IP-Adresse des mobilen Endgeräts passt sich bei jedem Wechsel des Subnetz dem jeweiligen aktuellen Aufenthaltsort an, das heisst es übernimmt den Präfix des Subnetz, in dem das Gerät sich momentan befindet. Dies hätte jedoch zur Folge, dass alle DNS-Einträge dem Endsystem angepasst werden müssten. Eine solche Änderung der DNS-Einträge dauert allerdings zu lange. Ausserdem würden alle TCP-Verbindungen abbrechen. Ein weiterer Punkt gegen diesen Lösungsvorschlag ist, dass die meisten Endsysteme aus Sicherheitsgründen nicht berechtigt sind die Einträge zu verändern.

1.1.2 Spezifische Wege zum Endsystem

Ein spezieller Eintrag in jedem Router zu jedem mobilen Endsystem wäre ein weiterer Lösungsansatz, dieser ist jedoch nicht umsetzbar, da diese schon ohne zusätzliche Einträge an die Grenze ihrer Kapazität stossen.

1.1.3 Mobile IP

Der dritte Vorschlag zur Behebung dieser Probleme ist die Erschaffung eines neuen Internetprotokolls, dem mobile IP.

1.2 Anforderungen an Mobile IP

Ein mobiler Knoten muss in der Lage sein, mit anderen Knoten zu kommunizieren, auch wenn er seine Lokation im Internet ändert. Dabei soll er seine IP-Adresse behalten können. Das ganze soll so aufgebaut sein, dass kein Kommunikationspartner etwas von der Mobilität eines Endgeräts mitbekommt. Bei einem Wechsel des Aufenthaltsortes eines mobilen Endgerätes sollen die höheren Schichten ohne Unterbruch weiterarbeiten. Ein weiterer wichtiger Punkt ist die vollständige Kompatibilität mit dem bestehenden IP. Da unter der Mobilität oft die Sicherheit leidet, muss Sicherge-
stellt werden, dass Registrierungsnachrichten, d.h. die Nachrichten zur Integration des mobilen Endsystems ins Netz, authentifiziert werden. Jedoch sollten möglichst wenig zusätzliche Daten ausgetauscht werden müssen, da die meisten mobilen Endgeräte nur über kabellose schmalbandige Anbindungen erreichbar sind.

2 Terminologie

Die folgenden fünf Begriffe werden im Zusammenhang mit Mobile IP oft verwendet und deshalb an dieser Stelle kurz erklärt.

- **Mobile Node (MN)**
  Als Mobile Node wird das Endsystem bezeichnet, welches den Netzanschluss wechseln kann, ohne seine IP-Adresse zu wechseln.

- **Home Agent (HA)**
  Der Home Agent befindet sich im Heimnetz des Mobile Node, typischerweise auf dem Router, der den Aufenthaltsort des Mobile Node verwaltet. Als Heimnetz wird das Subnetz verstanden, zu dem der Mobile Node laut seiner IP-Adresse gehört.

- **Foreign Agent (FA)**
  Der Foreign Agent befindet sich im momentanen Fremdnetz, also im in dem Netz, indem sich der Mobile Node momentan aufhält. Dieser befindet sich ebenfalls auf dem Router, welcher die vom Home Agent empfangenen Pakete an den Mobile Node weiterleitet.

- **Care of Address (COA)**
  Die Care of Address stellt den momentanen Aufenthaltsort des Mobile Node dar. Alle an den Mobile Node gesendeten Pakete gelangen zuerst zur Care of Address. Die Weiterleitung der Pakete zum Mobile Node geschieht durch einen Tunnel. Die Care of Address gibt den Tunnelendpunkt an.

- **Correspondent Node (CN)**
  Der Correspondent Node ist der Kommunikationspartner des Mobile Node. Er kann fix oder ebenfalls mobil sein.

3 Datentransfer

Werden Pakete von einem Correspondent Node an einen Mobile Node gesandt, geschieht dies wie üblich über die IP-Adresse des mobilen Knoten. Die Pakete erreichen auf dem herkömmlichen Weg den Router im Heimnetz des MN (1.). Dort befindet sich der Home Agent des MN, dieser weiss, wo sich der MN momentan befindet. Falls der MN im Moment am Heimnetz angeschlossen ist, werden die Datenpakete wie bei einem festen Knoten zugestellt. Die Mobile IP Funktionalität kommt also gar nicht zum Tragen. Befindet sich der MN jedoch in einem Fremdnetz, wird das Paket vom Home Agent abgefangen und gekapselt. Das heisst, das ursprüngliche Paket wird vollständig in ein neues IP-Paket gepackt, es wird dem ursprünglichen Paket also ein neuer Header gegeben. Dem auf diese Art und Weise neu entstandenem Paket wird als Zieladresse die Care of Address eingetragen. Somit ist das Paket bereit für die Reise durchs Internet, was mit Tunneling bezeichnet wird (2.). Hierbei ist der Weg des Pakets genau festgelet. Über die COA, welche den Tunnelendpunkt darstellt, gelangt das Paket schliesslich zum Foreign Agent, mit dessen Hilfe das Paket wieder entkapselt wird und an den mobilen Knoten gesandt wird (3.). Weil bei den beiden Header viele Felder doppelt vorkommen, gibt es auch eine minimal Kapselung, welche die doppelten Felder eliminiert, um so die Datenmenge zu minimieren. Der Rückweg ist wesentlich einfacher, insofern der Correspondent Node
nicht auch ein MN ist. Da der MN die Adresse des CN aus dem Absender kennt, kann er diesen ohne Umwege über die Foreign- oder Home Agents direkt zu seinem Kommunikationspartner schicken (4.). Der MN gibt dabei seine IP-Adresse an, dadurch erfährt der CN nicht, dass sich der MN in einem Fremdnetz befindet. Er schickt seine Pakete weiterhin an die Adresse des MN. So merkt auch der MN nichts von seiner Mobilität, da er die Pakete so erhält, wie sie der CN auf die Reise geschickt hat. Wäre der Kommunikationspartner ebenfalls ein Mobile Node, würde der Rückweg auf die selbe Weise geschehen wie der Hinweg.

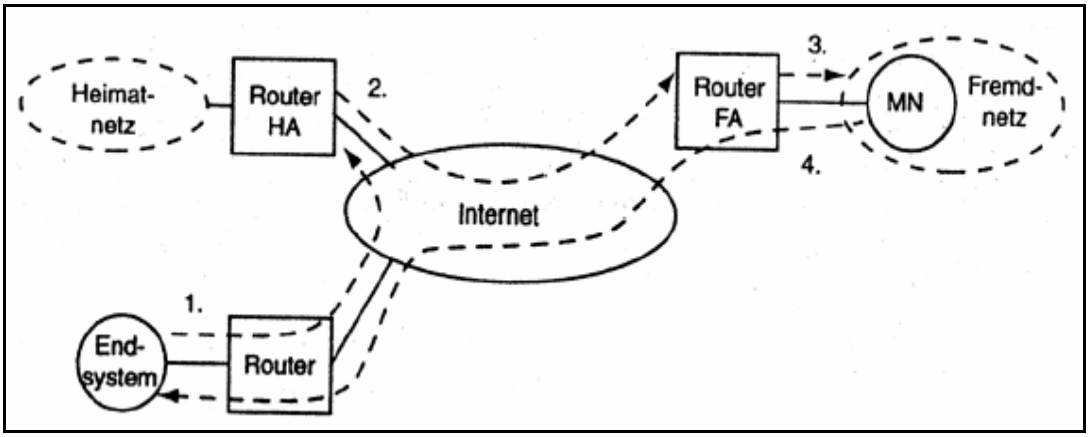


Fig. 1 Ablauf des Datentransfers zwischen einem fixen Endsystem (Correspondent Node) und einem mobilen Knoten

3.1 Agent Advertisement

Damit der Datentransfer von und zu einem Mobile Node möglich wird, muss dieser ins Internet integriert werden. Aus Sicherheitsgründen gibt es gewisse Vorkehrungen bei der Integration. Der mobile Knoten muss ständig informiert sein, in welchem Subnetz er sich befindet. Dies geschieht indem der Agent periodisch Nachrichten, sogenannte Agent Advertisements, in ihre jeweiligen Netze senden. Anhand des Headers kann der MN herausfinden in welchem Netz er sich befindet. Für mobile Knoten, die oft und schnell das Netz wechseln, können mittels Agent Solicitations sofort eine Nachricht beim Agent erzwingen.

3.2 Registrierung

Bei der Registrierung informiert der mobile Knoten den Home Agent über seinen aktuellen Aufenthaltsort. Dies kann durch Abmelden des mobilen Knoten beim Heimagenten, durch eine Anmeldung beim Heimagenten oder durch eine Registrierung durch einen Fremdagenten geschehen.

4 Optimierung

4.1 Umgehen des Triangular Routing

Das senden von Paketen via Home Agent zum Mobile Node, sogenanntes Triangular Routing, stellt einen Umweg dar. Dies führt zu grösseren Verzögerungen und zu einer erhöhten Netzlast. Es besteht die Möglichkeit, dass ein Sender den aktuellen Aufenthaltsort des Mobile Node lernt nachdem ihm dieser vom Home Agent mitgeteilt wurde. Er kann die zu sendenden Pakete anschliessend direkt zur alten Care of Address tunneln.

4.2 Reverse Tunneling

Die einfache Methode des Sendens von IP-Paketen vom mobilen Knoten zu einem Empfänger hat einen grossen Nachteil: Die Pakete erhalten eine topologisch falsche Absenderadresse, das heisst die Pakete werden aus einem physikalischem Subnetz gesendet, dessen Präfix nicht zur Sendeadresse passt. Da die meisten Router einen Firewall besitzen, werden solche Pakete einfach gelöscht. Dies hat zur Folge, dass das Mobile IP in solchen Netzen gar nicht funktioniert. Dies ist der Grund weshalb das Reverse Tunneling entwickelt wurde. Hierbei wird auch für den Weg vom Mobile Node zu einem Empfänger zwischen Foreign Agent und Home Agent ein Tunneling mit Kapselung eingesetzt.

4.3 IPv6

Mobile IP wurde ursprünglich für IPv4 entwickelt, die neue Version 6 erleichtert aber einiges. Viele neue Sicherheitsmassnahmen sind in IPv6 standardmässig integriert. Hinzu kommt, dass bei IPv6 keine Foreign Agents mehr benötigt werden, da alle Router das Router Adverisement beherrschen. Ein mobiler Knoten kann einen Sender direkt über seine Care of Address informieren.

5 Probleme

Einige Probleme sind noch nicht vollständig gelöst. Ein wichtiger Punkt ist die Sicherheit, die naturgemäss bei Mobilität ein Schwachpunkt ist. Die beschriebene Authentisierung der Registrierung beim Foreign Agent (siehe Kapitel 3.2) ist in der Realität schwierig, da der FA unter der Verwaltung einer fremden Organisation steht. Zudem ist vieles noch nicht standardisiert, so zum Beispiel die Verwaltung von Schlüsseln.

Referenzen

1. Abgegebene Unterlagen

2. http://www.tm.uka.de/lehre/SS01/vorlesungen/V_MK_Unterlagen/pdf/mk08-MobileIP.pdf

PPS-Seminar
Grundlagen der Internet-Technologie, WS 02/03

TCP und UDP

Thomas Hug
hugth@ee.ethz.ch
22. November 2002

1 Internet- Transportprotokolle

In der Transportschicht des Internets existieren zwei gängige Protokolltypen, die dem IP-Protokoll direkt übergeordnet sind. Es sind dies das verbindungslose User Datagram Protocol (UDP) und das verbindungsorientierte Transmission Control Protocol (TCP). Da UDP im Grunde das IP-Protokoll mit einem anderen Header ist, konzentrieren wir uns hier vor allem auf das TCP.

1.1 Das Transmission Control Protokol (TCP)

Da das Internet aus vielen verschiedenen Netzwerken mit total unterschiedlichen Topologien besteht, wurde das TCP entwickelt, das für einen zuverlässigen Bytestrom zwischen Sender und Empfänger zuständig ist. Es ist für die Zuverlässigkeit des Datentransfers verantwortlich, die IP nicht bieten kann und trotzdem von den Benutzern erwartet wird. Es ist das verbindungsorientierte Transportprotokoll des Internets.

1.1.1 Eigenschaften des TPC

- **Verbindungsorientiert**, das heisst, es wird zunächst eine Sitzung eröffnet, bei der im Netz ein Weg zum Empfänger gesucht und erstellt wird, über den anschliessend beliebig lang Daten ausgetauscht werden können. Am Schluss wird die Verbindung wieder abgebaut.
- **Byte-Strom- orientiert:** Die Daten werden beim Übertragen weder abgeändert noch interpretiert.
- **Zuverlässigkeit:** TCP nummeriert jedes Byte mit einer Sequenznummer. Solche Bytes können auch zusammen verschickt werden, man spricht dann von einem Segment. Der Empfänger bestätigt nun jedes Segment anhand der Sequenznummern der einzelnen Bytes. Bleibt eine Bestätigung aus, so sendet TCP das Segment erneut.
- **Flusskontrolle:** Zur Zwischenspeicherung der Segmente besitzt jede TCP- Instanz einen Puffer. Die Empfangende Instanz teilt nun der Sendenden immer mit wie viel Bytes sie noch akzeptiert, um ein überlaufen und somit Fehler zu vermeiden.
- **Vollduplex und Punkt zu Punkt Verbindung:** Zwei über TCP kommunizierende Instanzen können gleichzeitig senden und empfangen. Jede Verbindung hat jedoch genau zwei Endpunkte. TCP unterstützt weder Multicasting noch Broadcasting.

1.1.2 Der TCP- Protokoll- Header

Die sendende und die empfangende TCP- Einheit tauschen Daten in Form von Segmenten aus. Ein solches Segment besteht aus einem 20- Byte- Header, einem optionalen Teil, und den eigentlichen Datenbytes. Die Grösse der Segmente hängen von zwei Faktoren ab: Sie müssen einerseits in das 65.535 Byte grosse IP- Nutzdatenfeld passen, anderseits hat jedes Netz eine maximale Transfereinheit (MTU, Maximum Transfer Unit). Die Segmente können aber auch von den Routern in Teile zerlegt werden, falls sie zu gross sind.

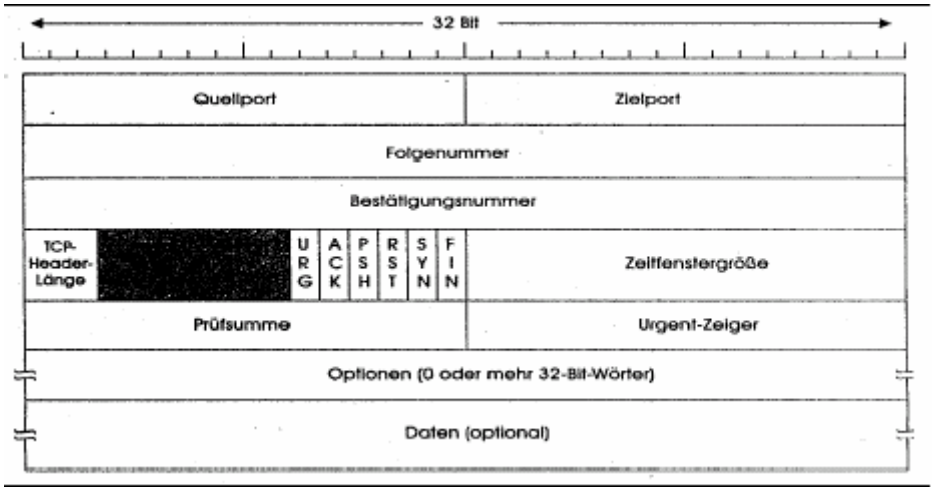


Abb. 1: TCP- Header

Jedes Segment besteht aus einem Header, der 20 Byte gross ist und alle Informationen für eine sichere Übertragung besitzt. Dieser Header hat ein festes Format (siehe Abb. 1). Im Folgenden werden wir jedes Steuerfenster des Headers einzeln betrachten um alle Funktionen einer TCP-Übertragung zu verstehen.

- **Source und Destination Port (Quell- und Zielport):** Hier werden die beiden Endpunkte einer Verbindung identifiziert. Jeder Endpunkt hat eine genaue Adresse die aus einer IP- Nummer und einer Portnummer besteht (dazu später mehr).
- **Sequence number (Folgenummer):** Jedes zu übertragende Datenbyte ist nummeriert. Dieses Feld enthält nun die Sequenznummer des ersten Datenbytes des TCP- Segments. Somit kann kontrolliert werden ob alle Segmente beim Gegenüber ankommen, und der Gegenüber seinerseits, kann die Segmente wieder der Reihe nach ordnen (es kann nämlich sein, dass ein früher gesendetes Päckchen nach einem späteren eintrifft).
- **Acknowledgement number (Bestätigungsnummer):** Dieses Feld besitzt die Sequenznummer des nächsten von der Partnerinstanz erwarteten Daten-Bytes, also die Sequenznummer des letzten Segments plus eins.
- **TCP Header Lenght:** Da das Optionenfeld nicht unbedingt Daten enthalten muss, gibt dieses Feld an, wie viele 32 Bit Wörter im TCP- Header enthalten sind.
- Das nächste (schwarze) Feld ist das Reservefeld, ist 6 Bit lang und wird nicht genutzt. Diese 6 Bits sind mit 0 zu initialisieren.

Die nachfolgenden 6 Bits sind sogenannte Flags und haben je nachdem ob sie auf 1 gesetzt sind folgende Bedeutung:

- **URG (Urgent Pointer)** ist auf 1, wenn der urgent Pointer (Dringend Zeiger) benutzt wird
- **ACK** ist auf 1, wenn die Acknowledgement number gültig ist. Falls das Segment keine Bestätigung enthält, ist ACK = 0 und das Feld Acknowledgement number wird ignoriert.
- **PSH (Push Daten)** ist auf 1, um den Empfänger aufzufordern, die Daten der Anwendung sofort zur Verfügung zu stellen. Ansonsten würde der Empfänger die Daten erst zwischenspeichern bis sein Puffer voll ist (Effizienzgründe).
- **RST (Reset)** ist auf 1, um die Verbindung abzubrechen (wenn z.B. ein Host abgestürzt ist). Im Allgemeinen entstehen Probleme beim Empfang eines Segmentes mit eingeschaltetem RST.
- **SYN** wird benutzt um eine Verbindung aufzubauen (mehr davon später).
- **Fin** wird benutzt um eine Verbindung abzubauen. Es besagt, dass der Sender keine weiteren Daten mehr zu übertragen hat.

- **Window:** Dieses Fenster dient der Flusskontrolle. Hier wird die Anzahl Bytes angegeben, die der Absender im weiteren Verlauf senden kann. Wenn das Window auf 0 ist, so besagt das, dass der Empfänger momentan keine Daten mehr aufnehmen kann und eine Verschnaufpause will. Dieses Feld ist eine Lösung auf das Problem von Überlastungen im Netz.
- **Checksum (Prüfsumme):** Dieses Feld bietet extreme Zuverlässigkeit. Es prüft den Header und die Daten indem die Summe aus allen 16 Bit Wörtern gebildet wird.
- **Options:** Dieses Feld bietet die Möglichkeit, zusätzliche Funktionen bereitzustellen, die nicht im Header definiert sind.

Der Datenteil kann durchaus leer sein, vor allem beim Verbindungsauf- und -abbau.

1.1.3 Eine TCP- Verbindung

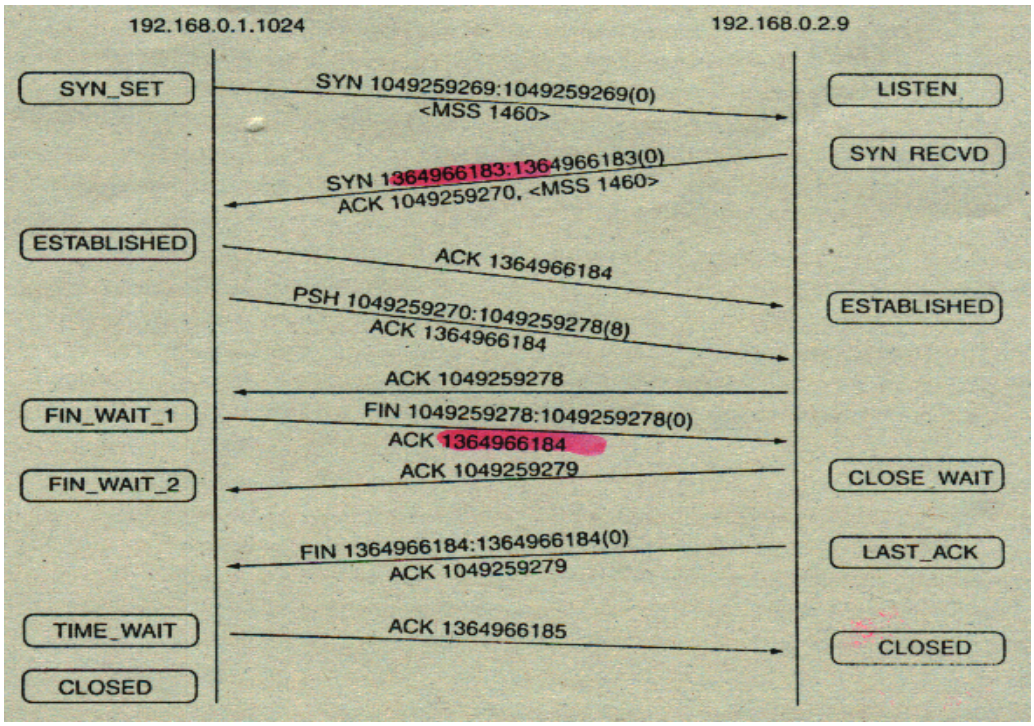


Abb. 3: Die TCP- Verbindung

Der Verbindungsaufbau

Bei TCP handelt es sich um ein verbindungsorientiertes Protokoll, das heisst, vor dem Datenaustausch muss eine Verbindung aufgebaut werden.

Um eine Verbindung aufzubauen muss das eine Ende der Verbindung (z.B. der Client) die IP- Adresse des anderen Endes (z.B. des Servers) kennen, und aber auch die Portnummer unter welchem der vom Client gewünschte Dienst angeboten wird. Das ist wie die Strasse und die Hausnummer einer Person, die ich suche. Die Ports haben Nummern von 0 bis 65'535. Es gibt bestimmte Ports, die so genannten „well known port numbers“, die einem ganz bestimmten Dienst zugeordnet sind wie z.B. http oder pop3 usw. (diese Portnummern können auf einer Liste nachgeschaut werden z.B. *4* http://acronymsonline.com/tcp-udp.htm*)*. Auch der Client selber wird identifiziert durch eine IP- Nummer und läuft über einen Port. Somit kann ein Client auch mehrere Verbindungen aufbauen und für jede Verbindung einen Port brauchen. Die Kombination aus IP-Adresse und Portnummer wird auch Socket genannt. Für eine Verbindung im Internet ist also an beiden Enden ein Socket nötig.

Der Verbindungsaufbau geschieht nun durch ein dreifaches Handshake. Man kann sich das vorstellen, wie wenn zwei Menschen telefonieren: Die eine Person ist zu Hause und ist bereit einen Anruf zu empfangen, die andere Person läutet an und dann beginnt das Gespräch mit: „Hallo wie geht's?“ „Danke gut und dir“ „Danke mir auch….“ Und erst dann beginnt das eigentliche Gespräch. Der Server befindet sich im LISTEN- Zustand, er wartet passiv auf eine ankommende Verbindung. Der Client sendet sein erstes Segment mit einem SYN- Flag und initialer Sequenznummer und geht in den Status SYN_SENT. Der Server, sendet ein Paket zurück, auch mit gesetztem SYN- Flag und einer ACK, das um eins höher ist als die Sequenznummer des vorigen Pakets (Rückwärtssynchronisation). Das dritte Paket kommt nun wieder vom Client, der analog sein ACK schickt, das SYN aber auf 0 hat. Nach diesem dreifachen Handshake befinden sich beide Enden (hier: Client und Server) im Status ESTABLISHED. Die Verbindung ist erstellt.

Der Datenaustausch

Beim nächsten Packet werden nun effektiv Daten ausgetauscht. Der Client sendet ein Segment und startet gleichzeitig einen Timer. Kommt das Segment am Ziel (hier also beim Server) an, sendet der Server ein Segment (mit Daten, falls vorhanden) mit der Bestätigungsnummer zurück. Läuft der Timer des Senders ab, bevor die Bestätigung zurückkommt, so wird das Segment vom Sender erneut gesendet.

Sender und Empfänger unterhalten je einen Sende- und Empfangspuffer, in welchem normalerweise die zu sendenden bzw. die angekommenen Daten zwischengespeichert werden, bis eine Mindestmenge an Daten im Puffer vorhanden sind, um sie dann zu senden oder weiterzuleiten. Das erhöht die Effizienz. Bei einigen Dienstleistungen ist es aber nicht sinnvoll die Daten zwischenzuspeichern (wie z.B. bei Telnet- Anwendungen). Diese können direkt weitergeleitet werden mit Hilfe des PSH- Flag.

Der Verbindungsabbau

Falls das eine Ende (hier der Client) nichts mehr zu senden hat, so sendet es ein Segment mit dem FIN- Flag, führt ein so genanntes *active close* aus und geht in den Zustand FIN_WAIT_1. Die Partnerinstanz (hier der Server) weiss, dass es keine Daten mehr erwarten muss, sendet eine Bestätigung (ACK), bringt den Client in den Zustand FIN_WAIT_2 und geht selber in den Zustand CLOSE_WAIT über. Nun kann der Server immer noch Daten senden und der Client könnte noch bestätigen, falls er aber fertig ist, so sendet er ein FIN-Flag und geht in den Zustand LAST_ACK. Der Client seinerseits bestätigt dies, geht in den Zustand TIME_WAIT und nach einer bestimmten Zeit in den Zustand CLOSED. Die Verbindung ist abgebaut.

1.2 Das User Datagram Protocol (UDP)

Der zweite Protokolltyp der Transportschicht einer Internetübertragung ist das User Datagram Protocol (UDP). Es ist das verbindungslose Transportprotokoll, das heisst die Datenpakete werden ins Netz geschickt und müssen ihren Weg dann selber finden, ohne eine Verbindung zum Ziel aufzubauen. Es wird auch nicht kontrolliert, ob die Pakete ihr Ziel erreicht haben oder nicht. Benutzt wird das UDP vor allem bei Anwendungen, die auf Grundlage einer Anfrage und einer Antwort laufen. Mit Hilfe von UDP besteht im Gegensatz zu TCP auch die Möglichkeit, das Broadcasting und Multicasting des Internet- Protokolls zu nutzen. Das Protokoll ist aber im Vergleich zu TCP sehr einfach aufgebaut. Da dieselben Funktionen eigentlich schon im Internetprotokoll (IP) vorhanden sind, hat UDP nicht mehr viel zu tun. Das einzige was bei IP noch fehlt sind die Portnummern. Im Allgemeinen kann man die Unterschiede zwischen TCP und UDP vergleichen mit einem Telefonanruf und einer Postkarte. UDP wäre die Postkarte: kurze Nachrichten und die Verbindung nicht abgesichert.

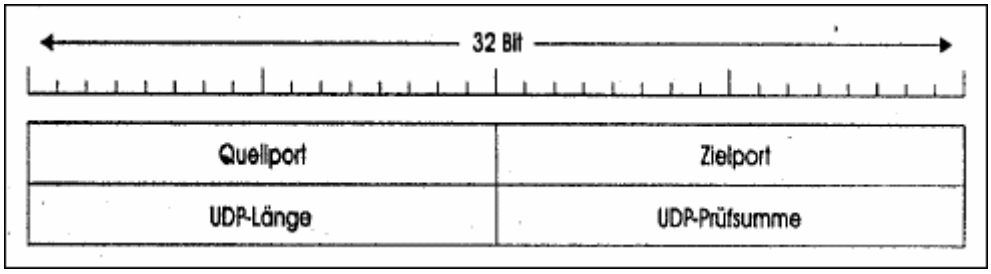


Abb. 3: UDP- Header

Der UDP- Header ist 8 Byte gross und besitzt vier Prüffelder. Sourceport und Destinationport dienen demselben Zweck wie jene zwei Ports im TCP- Header. Das Feld UDP-Length gibt die 8 Byte des Headers plus die Daten an. Auch die Checksum ist ähnlich wie bei TCP und beinhaltet den UDP- Header und die UDP- Daten.

1.3 Das Drahtlose TCP/ UDP

Theoretisch sollten die beiden Transportprotokolle netzunabhängig sein. Es sollte also einwandfrei funktionieren, egal ob es nun über eine Glasfaserleitung oder ein Funknetz läuft. Praktisch sieht das aber ein wenig anders aus. Viele TCP- Implementierungen sind auf verkabelte Netzwerken optimiert, versagen aber auf drahtlosen Netzen. Das Hauptproblem liegt bei der Überlastungsüberwachung. Da werden für das TCP viele Algorithmen eingebaut damit das Netz möglichst nicht überlastet. Da aber die drahtlosen Verbindungen viel unzuverlässiger sind können nicht dieselben Algorithmen angewendet werden. Das Netz ist aber äusserst inhomogen, die ersten paar Kilometer können z.B. verdrahtet sein und die anderen 100 Meter drahtlos.

Es gibt nun die Lösung des indirekten TCP, wobei die TCP- Verbindung aufgeteilt wird. Die erste Verbindung läuft vom Sender zur Basisstation, die zweite von der Basisstation zum Empfänger, wobei die Basisstation lediglich Pakete von einer auf die andere Verbindung kopiert. So können homogene Verbindungen geschaffen werden.

Doch dies führt wieder andere Probleme mit sich, denn nun bestätigt einfach die Basisstation die Segmente, und der Sender weiss nicht ob das Paket wirklich den Empfänger erreicht hat. Dazu muss nun noch einen Snooping Agent (Schnüffelagent) eingesetzt werden, der die TCP-Segmente auf dem ganzen Weg beobachtet. Doch dieses Prinzip hier noch zu erklären würde in dieser Arbeit zu weit führen.

1.4 Schlusswort

Wir haben nun zwei Transportprotokolle kennen gelernt und wissen im Groben, wie sie funktionieren. Das UDP für die einfache, schnelle Verbindung, das TCP für die zuverlässige, sichere Verbindung mit vielen Zusatzfunktionen. Das Ganze ist aber noch viel komplizierter, wie wir bei der drahtlosen Verbindung gesehen haben und man könnte noch viele Dinge diskutieren und entdecken, die es braucht damit überhaupt eine Verbindung im Internet zustande kommt.

Quellenverzeichnis

1. Abgegebenes Material
2. http://www.i-m.de/home/datennetze/tr_tcp1.htm
3. http://www.kl.unibe.ch/sec2/gymbield/unterricht/Faecher/Informatik/Architektur/tcp.htm
4. http://acronymsonline.com/tcp-udp.htm

PPS-Seminar
Grundlagen der Internet-Technologie, WS 02/03

Das
Hypertext Transfer Protocol

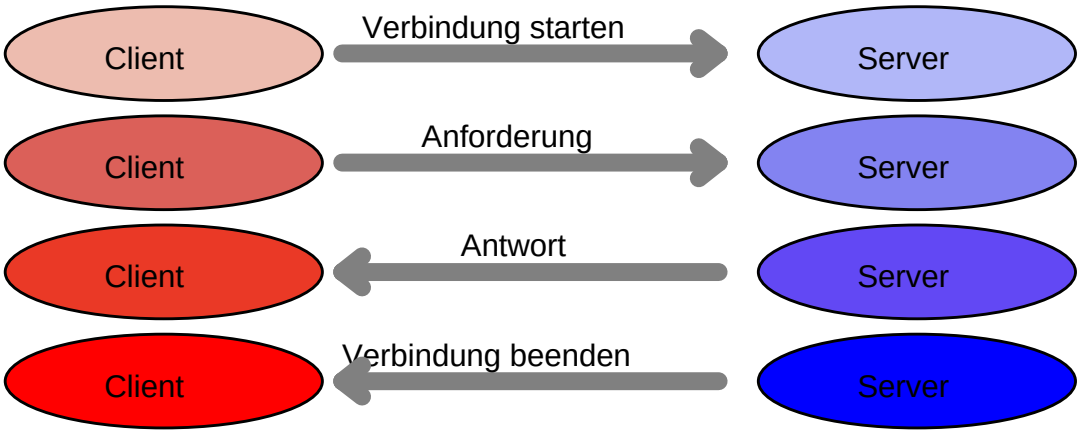
Suter Roman
suterrom@ee.ethz.ch
06. Dezember 2002

1 Einstieg ins HTTP

Der Datenaustausch nimmt heutzutage immer mehr zu und es muss auch immer noch schneller sein. Zu tausenden sind Computer und Server auf der ganzen Welt verteilt miteinander vernetzt und kommunizieren miteinander. Sie sprechen, fragen und geben antwort. Man könnte es so simple sagen. Dass aber eine so simple Kommunikation überhaupt stattfinden kann braucht es Regeln und Verfahren welche alle Rechner kennen und anwenden können.
Beim World Wide Web, dem wohl bekanntesten Dienstleistung im Internet, ist die Grundlage durch das HTTP-Protokoll gegeben. Durch dieses Protokoll kann man von der einen Seite Informationen anfordern, von der anderen Seite diese Forderungen erwidern oder weiterleiten. Ein klassisches Beispiel dazu wäre natürlich die Anfrage nach einer Webseite, welche der betreffende Server dann ausgibt oder es das bekannte 'Not found' auf den Bildschirm projiziert. Diese beiden Sachen sind dann aber auch schon die einzigen, welche der Benutzer auch wirklich mitkriegt. Für die ganze Client/Server Architektur des HTTP-Protokolls zu kennen, braucht es schon ein gezielteres nachprüfen. Dies empfiehlt sich natürlich vor allem für Administratoren von Servern, dessen Rechner meist nur darauf ausgerichtet sind solche Protokolle zu verarbeiten. Für den normalen WWW Benützer ist es im Gegensatz dazu nicht wichtig zu wissen, wie das WWW intern funktioniert.

1.1 Request/Response Protokoll

Etwas zur näheren Definition eines Protokolls: es ist so dass ein Protokoll gewisse Regeln festsetzt die sowie der Client wie auch der Server kennt und bei der gegenseitigen Kommunikation anwendet. Es entsteht also eine Reuest/Response Interaktion zwischen diesen beiden Subjekten.
Darstellung 1 zeigt eine solche typische Verbindung zwischen zwei Rechnern:



Darstellung 1

Der Client als erster ist ein Programm das Verbindungen (TCP) aufbaut um seine Request's dann auch zu senden. Beispiele für einen Client sind natürlich der gute alte Browser oder sonst welche Programme die das Netz nach Informationen absucht.
Der Response Partner des Clients ist der Server. Er hat die gesuchten Informationen für den Client auf irgendeine Weise interpretieren können und gibt deshalb eine Antwort. Dass er es überhaupt interpretieren konnte, für dies ist natürlich wieder das HTTP Protokoll verantwortlich.

1.1.1 Request

Zuerst kommt ein Request. Denn bei einer Interaktion zweier Parteien muss immer erst einer einen Antrag zur Kommunikation machen. Durch das TCP, dem allgemeinen Verbindungsprotokoll, wird eine Verbindung vom Client zum Server aufgebaut. Die Request Nachricht wird dann nach erfolgtem Verbindungsaufbau vom Client an den Server gesendet und gibt den Request des Client an.
Eine Request Nachricht ist wie folgt zusammengesetzt:

Request-line:	*Method SP Request-URI SP HTTP-Version CRLF*
	Leerzeile!
Header:	**(general-header | request-header | entity -header)*
	Leerzeile!
	CRLF
Inhalt:	*[Message-Body]*

Darstellung 2

Im Falle der Request Nachricht handelt es sich in der start-line der Nachricht im Gegensatz zu einer Response Nachricht natürlich um eine Request-line. Darin enthalten sind die wichtigsten Informationen über den Antrag: der Name einer Methode, ein Trenner (z.B. ein SPace) , einer Request-URI, wieder ein Trenner und schließlich die HTTP-Versionsausgabe.
Für die Methoden gibt es Standartmethoden und 'Erweiterungs- Methoden. Ein Server muss aber nur mindestens *GET* und *HEAD* verstehen. Alle anderen Methoden sind optional. Die weiteren gängigsten Standartmethoden sind *POST, PUT, DELETE* und *TRACE.*
Alle diese Begriffe geben an, welche Methode der Server auf die in der *Request-URI* angegebene Ressource anwenden soll.

GET — Diese Methode wird vom Client eingesetzt, um eine Information von einem Server abzurufen. Normalerweise handelt es sich dabei um ein Dokument oder eine andere statische Information (u.a. Medialdateien). Welche Datei es sein soll wird in der Request-URI angegeben.

PUT — PUT ist sozusagen das Gegenstück von *GET.* Hier wird im Gegensatz dazu eine Information dem Server übergeben. Der Request fordert dadurch den Server an die angehängten Informationen (im Entity, Body) an der angegeben Stelle (Request-URI) zu speichern.

OPTIONS — Hier werden Informationen über die möglichen Kommunikationsoptionen angefordert. Der Client wird dadurch über die allgemeinen Optionen innerhalb der Interaktionskette über einen Server informiert.

TRACE Durch diese Methode kann festgestellt werden wie eine Nachricht durch Zwischenstationen wie zum Beispiel Proxies verändert worden ist. Das kann zu Diagnosezwecken benützt werden, da es für einen Benützer durchaus interessant sein kann, wie eine an einen speziellen Server gesendete Nachricht tatsächlich von diesem empfangen wird.

DELETE Fordert den Server dazu auf, eine bestimmte, in der Request-URI angegebene Datei zu löschen.

POST Diese Methode ist ganz ähnlich wie die Methode PUT, hat aber einen wesentlichen Unterschied. Hier werden nicht neue Informationen auf einem Server gespeichert, sondern die Request-URI gibt einen bereits existierenden Pfad an, wo eine bereits existierende Information überschrieben wird.

1.1.2 Response

Die Antwort durch ein HTTP-Protokoll von einem Server fällt sehr ähnlich aus (**vergl. Darst. 2 und 3**), wie bei einer Anforderung. Im wesentlichen unterscheidet sich die Antwort durch die Status-line, die nun anstelle der Request-line tritt.

Status-line:	*HTTP-Version SP status-code SP reason-phrase CRLF*
	Leerzeile!
Header:	**(general-header | response-header | entity-header)*
	Leerzeile!
	CRLF
Inhalt:	*[message-body]*

Darstellung 3

Die Status-line enthält wie auch schon die Request-line wieder die Protokoll Version. Anschliessend folgt dann einer der wichtigsten Teile für den Benutzer selber. Die Response enthält einen sogenannten Status-code, der eine Aussage darüber macht, was der Request bewirkt hat. Dabei handelt es sich um den dreistelligen, ganzzahleigen Ergebniscode. Im Moment kann die Anfangsziffer von 1-5 gehen, die verschiedenen Status werden dadurch auch gerade kategorisiert:

1xx Zeigt eigentlich lediglich an, dass der Request vom Server erfolgreich empfangen wurde.
Beispiel: 100 (continue) → Information, Server bearbeitet Request

2xx Der Server meldet, dass er einen Request empfangen und bearbeitet hat.
Beispiel: 200 (ok) → bearbeitet,
202 (accepted) → Server nimmt gesendete Informationen an

3xx Hier braucht der Server weitere Informationen um den request richtig zu verstehen und zu bearbeiten. Weitere Massnahmen vom Client sind deshalb erforderlich.
Beispiel: 300 (multiple choice) → Client muss genauer auswählen
302 (found)→ Server erwartet weitere Befehle für dieses Dokument

4xx	Dies tritt auf, wenn ein HTTP Request nicht bearbeitet werden kann, weil der Client einen Fehler gemacht hat (Syntaxfehler, Autorisierungsfehler).
Beispiel:	401 (unauthorized) → keine Berechtigung für dieses Dokument 404 (not found) → der Klassiker, meist durch ungültige URL 415 (unsupportet media typ) →nicht übertragbarer Medientyp
5xx	Mit der 5 als Anfangsziffer handelt es sich um einen Server Fehler. Meist war der Request in Ordnung, trotzdem kann ihn der Server nicht interpretieren. Der Server gibt wenigstens an, ob es sich bei der Fehlersituation um ein vorübergehendes oder um ein dauerhaftes Phänomen handelt.
Beispiel:	500 (internal server error) → Server kann nicht reagieren 505 (http version not supported) → veraltete HTTP-Version

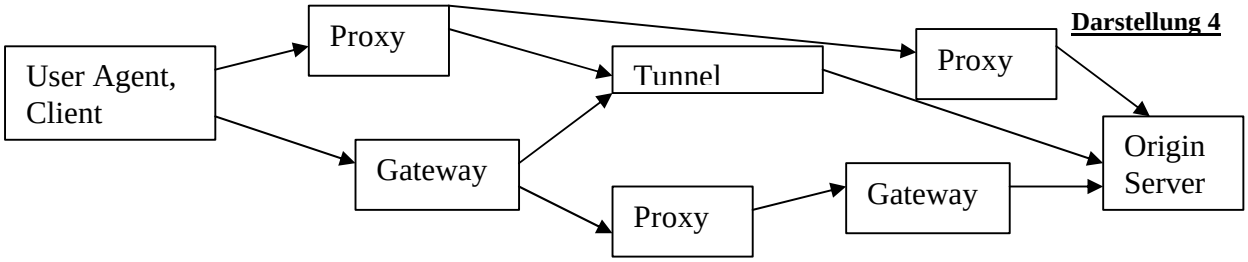
1.1.3 Header

Um noch genauere Angaben über die Interpretation der Forderung oder der Antwort zu machen, hat die Version HTTP/1.1 eine Menge *HEADER* zur Verfügung.

- General Header:	Sie betreffen lediglich die übertragene Nachricht und finden sowohl bei Request als auch bei Response Nachrichten allgemeine Verwendung. Sie betreffen nicht das übertragene Entity.
- Entity Header:	Entity Felder definieren Informationen über den Entity Body oder, falls ein solcher nicht vorhanden ist, über die im Request angegebene Ressource. Sie können also beispielsweise die Länge oder die Codierung angeben, nicht aber den Inhalt des Entity selber.
- Request Header:	Enthält Informationen über den Request oder über den sendenden Client selber. Allerdings enthält dieser Header keine Informationen über den Körper der Nachricht.
- Response Header:	Diese Header können bei Response Nachrichten dabei sein. Sie enthalten Informationen die nicht in der Status-line angegeben werden können. Auch sie geben aber keine Informationen darüber ab, was im Entity steht.

1.2 indirekter Datenverkehr

Datenverkehr zwischen einem Client und dem Server, wo die Daten gelagert sind, haben in der grossen Verküpfung von Rechnern im WWW meist einen längeren Weg zurückzulegen. Deshalb ist es naheliegend, dass eine Verbindung zwischen einem Client und einem Origin Server meist über mehrere Stationen stattfindet (**Darst. 4**). Es gibt drei Möglichkeiten wie solche gelinkte Verbindungen bestehen können.



1.2.1 Der Proxyserver

Der Proxy übernimmt sowohl die Funktion eines Client als auch die Funktion eines Servers. Das heisst genauer, dass er vom Client ankommende Requests entgegen nimmt, gleichzeitig aber auch selber Requests an einen Server verschickt. Er unterstützt daher den Origin Server, wo die Daten eigentlich gelagert sind. So kann der Proxy zum Beispiel in seinem Cache häufig besuchte Websites zwischenspeichern und sie direkt an einen Client weitersenden → der Origin Server wird entlastet.

1.2.2 Der Gateway

Ein Gateway ist auch ein Programm das anderen Servern als Zwischenprogramm dient. Ein Client, der einen Request an ein Gateway sendet, weiss jedoch meist nicht, dass er nicht mit dem Origin Server kommuniziert.

1.2.3 Der Tunnel

Ein Tunnel ist dann wiederum ein Programm, welches bei der HTTP-Kommunikation als blinde Zwischenstation dient. Aus diesem Grunde wird die Nachricht aber weder interpretiert noch bearbeitet. Der Tunnel befördert die Nachricht nur an den gewünschten Server weiter.

1.3 Geschichte: von HTTP/0.9 - 1.1

Bei den ersten Entwürfen dieses Protokolls ging es vor allem darum, eine einfache aber trotzdem schnelle Datenverteilung zu erlangen. Auf den anfänglich knapp an Ressourcen laufenden Rechnern sollte also nicht viel Raum für das Protokoll bereitgestellt werden müssen und, das war sehr wichtig, es sollte schnell reagieren, denn die Daten waren wie heute auch schon auf einer ansehnlichen Menge von Servern verteilt.
Natürlich haben sich bis heute die Ziele in Sachen Schnelligkeit und Einfachheit nicht geändert, aber mit der heutigen Nachfrage nach Leistung und vor allem Konstanz sind die Ansprüche stets gestiegen.

1.3.1 HTTP/0.9

Das erste Protokoll war der Grundstein des Erfolgs des heutigen Protokolls. Es glänzte durch seine Einfachheit: Hier wurde noch ausschliesslich die Methode *GET* unterstützt. Der Client sendete das Schlüsselwort und den Dateinamen an den Server. Nach dieser Anfrage antwortete der Server mit den Daten und unterbrach zum Beenden nach Datenübertragung die Verbindung einfach.
Die grösste Einschränkung dabei war, dass nur reine Texte übertragen werden konnten. In der heutigen Zeit der Medienhochkonjunktur nicht mehr auszudenken.

Weiter war es nicht möglich dem Server Daten zu übermitteln. Der Server war nur in der Lage nach einer Anfrage durch ein HTTP-Protokoll mit dem entsprechenden File zu antworten. Diese beiden Einschränkungen galt es zu überbrücken um das WWW weiterzubringen.

1.3.2 HTTP/1.0

Die Version 1.0 war schon eine wesentliche Verbesserung. Es war nun möglich, verschiedene Medientypen zu senden und diese zu empfangen. Weiter konnte der Server nun erweiterte Antworten geben. Statt nur die Daten selber zurückzugeben konnte er nun auch noch Informationen die Daten anhängen. Diese zusätzlichen Informationen waren in den neuen Header-files dieser Version gespeichert. Die Antwort des Servers erlangte dadurch ein verbessertes Verständnis. Ein Beispiel davon ist die Fehlermeldung, welche nun neu durch das Header *POST* gesendet werden konnte. Sie ist zwar nicht beliebt, aber sie enthält für den Benutzer wichtige Informationen.
Die Version 1.0 basierte aber weiterhin auf einer einzelnen Request/Response-Interaktion pro Verbindung und erforderte, dass der Server nach dem Senden des Requests die Verbindung unterbrach. Dies war ein Problem welches zur nächsten Version führen müsste, und es war mitunter ein Grund dafür, dass diese Version nie richtig zum Einsatz gelangte.

1.3.3 HTTP/1.1

Dies ist die aktuelle Version des Protokolls. Es brachte die Möglichkeit zur Verbindungserhaltung (P-HTTP), daneben aber auch gleich eine akzeptable Variante der bisherigen Verbindungstechnik, welche durch mehrere Requests die Verbindungsabbrüche kompensiert. Weitere Neuerungen waren die Unterstützung des Header-Felds *Host*, neue Reqest-Methoden und die Möglichkeit, Daten nun auch teilweise zu übertragen. Auch die Content Negotiation hat sein erstmaliges Auftreten in dieser Version.
Das HTTP-Protokoll ist damit wirklich vielseitig geworden, dabei aber auch umfangreich. Mittlerweile sind die anfangs gesteckten Ziele, Einfachheit und Schnelligkeit, wieder weiter weg gerückt. Diesen Trend zu brechen soll nun die Herausforderung für die nächste Version des Protokolls sein.

1.4 Schlusswort

Das ganze World Wide Web läuft nur mithilfe dieses kurzen Protokolls. Trotz seiner vermeintlich sehr einfachen Weise ist es in der Lage ganz komplex wirkende Resultate dem Benützer zu übergeben. Durch ganz einfache Request/Response Interaktionen kann das Protokoll heute dem Benützer so viel bieten, dass eine ganze Web-Gesellschaft entstanden ist. Und diese Gesellschaft wächst und verändert sich noch immer - mit ihr das Werkzeug: HTTP.

Quellenverzeichnis:

1. http://www.informatik.uni-trier.de/~sack/ProSeminarSS2001/HTTP/ppframe.htm

PPS-Seminar
Grundlagen der Internet-Technologie, WS 02/03

XML
Datenstrukturierungssprache

René Kamer
Rkamer@ee.ethz.ch
13. Dezember 2002

1 XML – Eine Einführung

In diesem Kapitel soll klargemacht werden, was XML überhaupt ist. Es wird erklärt, wieso XML überhaupt entwickelt wurde und woher die Regeln kommen, die der „Web-Sprache“ XML zugrunde liegen.

1.1 Was ist XML?

XML steht für ”Extensible Markup Language“, was auch schon die wichtigste Neuerung gegenüber HTML zeigt. Denn XML ist, gegenüber HTML, eine erweiterbare Internet-Hochsprache, d.h. der Anwender kann selbst die bestehenden Strukturen und Formate von XML erweitern. Doch dazu später mehr.

XML ist, ebenso wie auch HTML, von der Web-Hochsprache SGML abgeleitet, welche aber viel komplexer und mächtiger ist als XML. Man sagt auch SGML ist die Muttersprache aller Websprachen, denn mit ihr kann man wirklich alles darstellen, von der Musiknote über Grafiken usw.

Ausser der Fähigkeit, erweitert zu werden, unterscheidet sich XML auch noch in anderen Belangen von HTML. Bei HTML sind Tags Anweisungen für den Browser, wie er eine anzuzeigende Seite darzustellen hat. Anders als bei HTML werden in XML die Tags auch zur Datenstrukturierung eingesetzt, was mehrere neue Möglichkeiten bietet.

Prinzipiell ist also die Information, die ein HTML-Dokument liefert die gleiche, wie die, welche ein XML liefert, mit dem Unterschied, dass ein XML File eine logische Struktur bietet.

1.2 Warum wurde XML entwickelt?

HTML hat sich jetzt schon jahrelang behauptet als häufigstes Format für Web-Inhalte. Warum sollte man also plötzlich auf die Idee kommen, eine neue Sprache einzuführen, was ja mit erheblichem Aufwand verbunden ist und auch eine umfassende Vorausplanung erfordert. Der Grund ist der, dass HTML zu einfach gestaltet ist. Ausser dem Inhalt kann man einem HTML nur noch vorgeben, wie es im Web dargestellt werden soll, und das auch nur in dem Beschränkten Rahmen der HTML-Syntax. Man stelle sich nun vor, man sucht auf einer HTML-Page die Farbe ”Schwarz“. Dies wird jedoch keinesfalls nur ein ”Schwarz“ zurückliefern, sondern vielleicht auch noch Schwarzenstrasse, Schwarzkopfindianer und andere unbrauchbare Suchresultate. Es ist also keine Strukturierte Information vorhanden. Bei XML kann man sich nun die Eigenschaft der Tag-Struktur zu nutze machen und effektiv nach dem Tag ”Farbe“ suchen und innerhalb dieses Tags nach ”Schwarz“. Dies wird ein einheitlicheres Suchresultat ergeben.

Ebenfalls ein wichtiger Grund für die Entwicklung von XML war, dass im HTML-Bereich immer mehr Wildwuchs betrieben wurde. Jeder Hersteller hatte seine eigenen Standards, es wurde und wird immer unübersichtlicher, wie ein HTML genau auszusehen hat. Dies führte dazu, dass eigentlich praktisch keine HTML’s den vom W3C definierten Standards entsprach und daher Browser, welche die Information darstellen wollten, sehr tolerante Parser (Interpret für Informationen in einem Web-File) bereitstellen mussten, um die Seite noch einigermassen korrekt darstellen zu können. Ein XML dagegen muss schon in der Erstellungsweise ”well formed“ sein. D.h. dass die Richtlinien strenger gehandhabt werden als bei HTML files, XML sind in sich abgeschlossen, können folglich ohne sogenannte Interpreter-Files dargestellt werden und haben ihre logische Struktur. Doch dazu mehr im nächsten Abschnitt 2.

2 Aufbau und Darstellung von XML-Dokumenten

Wie vorhin erwähnt gibt es für XML wesentlich strengere Vorschriften als für HTML in Bezug auf den Programmcode. Die W3C[1] hat diesbezüglich die XML-Spezifikationen herausgegeben, welche die Regeln, um ein XML zu schreiben, genauestens festlegt (die neueste Spezifikation ist vom 15. Oktober 2002 und kann auf Englisch unter [1] oder auf Deutsch unter [2] abgerufen werden). Ein XML Dokument muss in sich abgeschlossen sein, d.h. es muss zu jedem Tag auch ein End-Tag geben, wogegen in HTML wie auch in SGML die Markup Minimization eingesetzt wird (man braucht nicht die vollständige Information der Tags anzugeben, also nur minimiert, der fehlende Teil wird vom Parser dazu interpretiert). Ausserdem bietet XML auch die Möglichkeit, XML spezifische Teilprogramme einzusetzen, um z.B. eine mathematische Formel darzustellen.

2.1 DTD's

Ein DTD enthält die XML-Tags, welche, wie anfangs erwähnt, auch vom Benutzer mitverändert werden können. Es gibt auch in HTML eine DTD, nur ist diese im Browser gespeichert und kann nicht verändert werden. Da XML Tags also folglich immer anders sein können, müssen diese mit dem XML-Dokument mitgeliefert werden. Hier ein kleines Beispiel einer XML DTD:

```
<?xml version="1.0"?>
<!DOCTYPE result[
     <!ELEMENT result (item+) >
     <!ELEMENT item (name, (phone | email)*)+ >
     <!ELEMENT name (#PCDATA) >
     <!ELEMENT phone (#PCDATA) >
     <!ELEMENT email (#PCDATA) >
]>
```

Diese DTD kann entweder als einzelnes File mit dem XML-File mitgeschickt werden oder auch gleich in's XML File eingebunden werden. Wie man sieht wird auch die XML Version angegeben, da sich die Versionen immer noch stark unterscheiden und daher auch anders aufgebaut und zu interpretieren sind (die aktuelle Version ist XML 1.1). Was oben im DTD steht kann der Browser nun als Baumstruktur zusammenstellen.

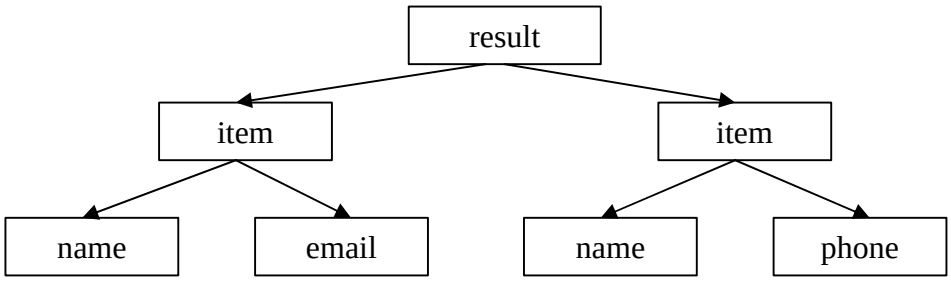


Fig. 1 Baumstruktur eines DTD

2.2 Wie wird XML dargestellt

[1] World Wide Web Consortium

Der Code eines XML ist nicht selbstdarstellend wie ein HTML, das ja die Informationen zur Darstellung in den Tags bereithält, sondern hier dienen die Tags der Strukturierung wie oben dargestellt. Wenn man also ein XML-Dokument mit DTD an einen Browser schicken würde, würde dieser einfach den Inhalt des XML's mit den Tags ausgeschrieben darstellen. Damit das nicht passiert, muss mit dem XML und dem DTD noch ein weiteres File mitgeschickt werden, das die Darstellung des Dokuments regelt. Hier gibt es zwei Möglichkeiten: Entweder man schickt ein CSS- (Cascading Style Sheets) oder ein XSL (Extensible Style Language) File mit. Beide regeln die Darstellung des XML's, wobei die CSS-Variante schon bei HTML eingesetzt wurde und für XML nur wenige Modifikationen benötigt. Ein kleines Beispiel:

```
item {display: block; margin-bottom: 5mm }
name {display: list-item; font-weight: bold }
phone {display: list-item; font-style: italic }
       email {display: list-item; text-decoration: underline }
```

Die XSL Variante ist die offizielle, vom W3C spezifizierte Style Sheet Sprache, um XML auch in seinem vollen Umfange nutzen zu können. Sie ist sehr komplex, weshalb hier nicht näher darauf eingegangen wird.

3 Die Konvertierung von XML

Da XML ja nun eine vielseitig verwendbare Sprache mit strukturierter Information bereitstellt ist es vielleicht sinnvoll, ein XML in ein anderes Format zu konvertieren und vice versa, sowie auch von XML zu XML (ist auch eine Anwendung, da die DTD's ja verschieden sein können). XML hat aber nun so seine Eigenheiten, welche nicht unbedingt mit den anderen Formaten verträglich sein müssen. Z.B. ist ein XML selbstbeschreibend und geschlossen, hat eine durchsuchbare Dokumentstruktur und ein sehr leistungsfähiges Linking modell (dazu später mehr). Also erfordert eine Konvertierung klare Regeln und Konventionen, damit nicht ungewollt Information verfälscht werden oder verloren gehen.

3.1 Die Konvertierung verschiedener Formate nach XML

Für alle Formate, die zu XML konvertiert werden sollen, muss eine klare Zuordnung gemacht werden zwischen dem Quell- und dem XML-Format. Das heisst, man muss die DTD so modellieren, dass die Struktur des Quellformats exakt in der XML-Version wiedergespiegelt wird. Hier gibt es zwei verschiedene Szenarien: Man kann für das Publishing konvertieren oder um das XML format In-House (Firmenintern) zu benutzen.

Wenn nur für Publishingzwecke konvertiert wird, ist hauptsächlich darauf zu achten, dass die Darstellung im Web korrekt ist. Daher konzentriert man sich weniger stark auf die Modellierung der geeigneten DTD, als auf das korrekte Implementieren der Style Sheets.

Wenn man allerdings für In-House Zwecke konvertiert, muss man sehr korrekt darauf achten, die DTD korrekt zu schreiben, da auf keinen Fall Information verloren gehen darf. Hat man jedoch diese DTD einmal zusammen, lässt sich die Konvertierung faktisch automatisieren, falls man nicht viele verschiedene In-House Formate nach XML konvertieren muss.

Die Konvertierung eines Formats nach XML wird in Fig. 2 dargestellt.

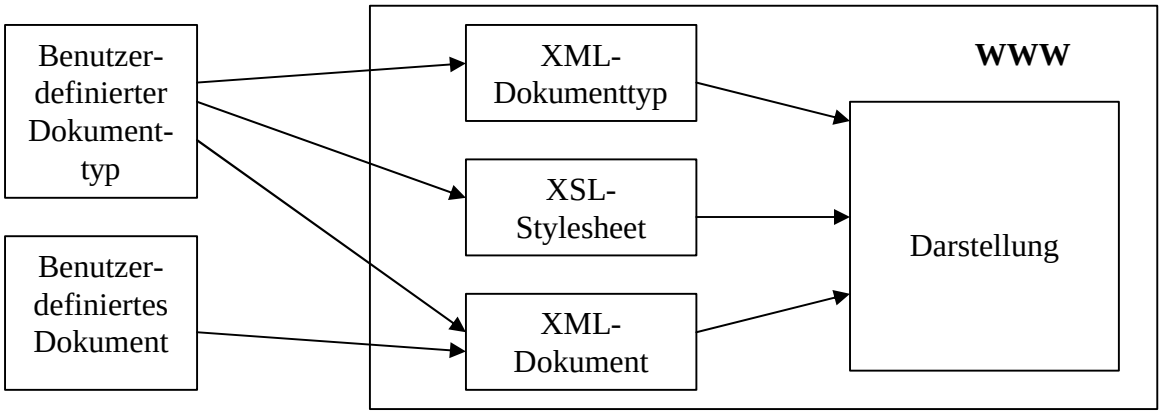


Fig. 2 Konvertierung eines Fremdformats nach XML

3.2 Die Konvertierung von XML nach XML

Obwohl schon XML zum Einsatz kommen mag, muss in vielen fällen trotzdem nochmals in ein weiteres XML-Format (eigtl. nur andere DTD) konvertiert werden, um z.B. XML's mit anderen XML-basierten Anwendungen auszutauschen. Häufig bringt dies einen (beabsichtigten oder unbeabsichtigten) Informationsverlust mit sich. Die Umwandlung ist indes ziemlich einfach: Man muss die zwei DTD's einander zuordnen (möglicherweise auch unter Verwendung von Style Sheets), was sich, vorausgesetzt klare Zuordnung, vollständig automatisieren lässt. Eine Handlung von Seiten des Anwenders sollte nur noch in vereinzelten Fällen nötig sein.

3.3 Die Konvertierung von XML in andere Formate

Eine Umwandlung in einen anderen Formattyp ist im Grunde genommen die Extraktion der Daten aus dem XML-File. Dazu muss ein entsprechender Konverter entwickelt werden, um die DTD für das Fremdformat korrekt zu interpretieren. Falls die Zielarchitektur ähnlich komplex ist wie DTD gestaltet sich dies relativ einfach. Wenn aber das Zielformat eine niedrigere Abstraktionsstruktur aufweist, können sich grosse Probleme stellen. Es gehen höchstwahrscheinlich Informationen verloren, wenn diese Zielarchitektur nicht manuell aufgebaut wird, sodass die Informationsgehalte wenigstens einigermassen übereinstimmen.

4 Links in XML

Das Linkkonzept in XML ist sehr verschieden von den bis jetzt von HTML bekannten Mechanismen. Die Link-Einbettung in XML hat gar eine eigene kleine Sprache: XML Linking Language (Xlink). Sie bietet XML das Umfeld, um Links einzubetten.

Während in HTML Links immer Bestandteil der Ressource sind, die sie verknüpfen, muss das in XML nicht unbedingt so sein. Die Unterschiede werden in diesem letzten Kapitel erklärt.

4.1 Inline Links

Diese XML Link-Art ist noch am ehesten mit HTML vergleichbar, da der Link noch Teil der Ressource ist, welche der Link mit einer anderen verbindet. Die Inline Links sind also stark an die Ressource gebunden.

4.2 Out-of-line Links

Out-of-line Links sind nun ein völlig neues Konzept. Der Link ist nicht mehr an die Quell-Ressource gebunden, sondern wird ausserhalb des Dokuments festgelegt. Dies erfordert jetzt aber Kenntnis, wo sich der Link befindet. Wie soll also ein solcher out-of-line Link von einem Dokument referenziert werden? Man macht einen Zeiger auf diesen, was man auch als verlinken eines Links bezeichnen könnte. Wenn das Linking-modell also dadurch so verkompliziert wird, muss es gute Gründe geben, out-of-line Links zu benutzen. Diese sind hauptsächlich in der Link-Topologie gegeben welche zwei völlig neue Konzepte zulässt:

- Bidirektionalität eines out-of-line Links: Während es in HTML nur möglich war, vom Link auf die nächste Seite zu kommen, aber nicht wieder über den gleichen Link zurück auf die Ursprungsseite, ist dies in XML anders. Der Link kann (muss aber nicht) beidseitig funktionieren, sodass zwischen zwei Files immer auf dem gleichen Link bleibend hin- und her gesprungen werden kann.
- Anzahl der Knoten: Während man in HTML nur zwei Ressourcen über einen Link verknüpfte, ist es in XML nun möglich, eine praktisch unbegrenzte Anzahl Ressourcen miteinander zu verlinken.
  Ein Beispiel wäre eine Mailliste, wo man die Mailing-Liste selbst mit all ihren Teilnehmern

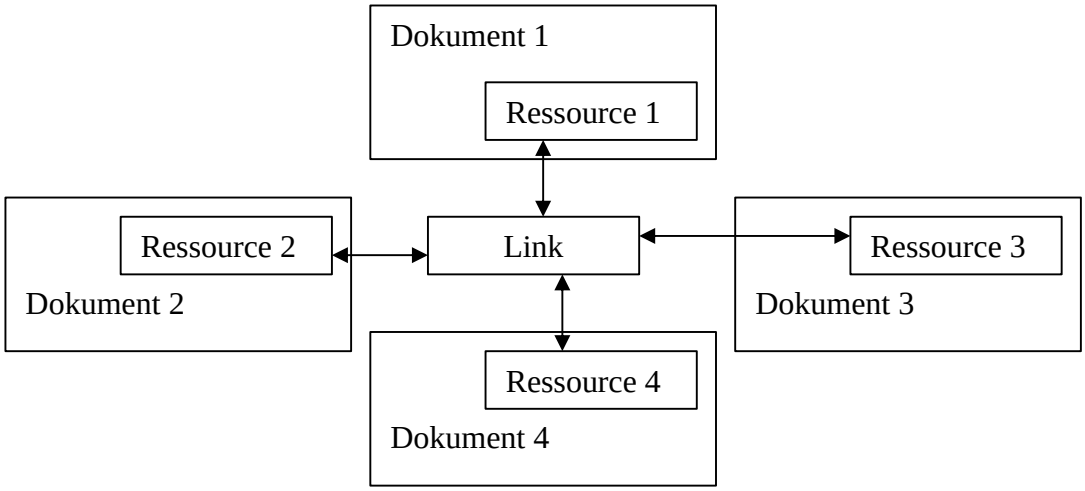


Fig. 3 Bidirektionale Links mit mehreren Knoten

verknüpft. In Abhängigkeit von der genauen Definition des Link-Typs können die Links zu den Teilnehmern Links zu ihren Homepages oder zu ihren E-mail-Adressen sein. Man kann die Liste in alle Richtungen durchlaufen, also die Mailingliste selbst betrachten, oder den Weg zu anderen Teilnehmern zurückverfolgen.

Diese zwei Möglichkeiten bieten schon ein grosses Feld von Anwendungen, das Konzept beider ”Tools“ wird in Abb. 3 nochmals anschaulich gemacht.

5 Fazit

Soviel gut durchdachte Arbeit wird schon heute belohnt, da XML mittlerweile doch schon recht verbreitet ist und auch fast alle Browser XML interpretieren können. Es wird jedoch niemals HTML ablösen, da HTML einfach für viele Anwendungen absolut ausreichend ist und daher weiter verbreitet bleiben wird, was gute aber auch schlechte Seiten hat. Wenn man sich vorstellt, dass XML vorherrschend wäre, dann wäre das Internet ein Glaspalast. Persönliche Daten würden so schnell aufgespürt, dass man sich vor Spam und sonstigen Anzüglichkeiten nicht mehr gefeit wäre. Andererseits wäre genau dies praktisch, falls man zum Beispiel eine bestimmte Verknüpfung von Gegebenheiten suchen wollte für einen Vortrag. Es wäre das Eldorado für Informationen (nicht dass es das nicht schon wäre). Aber wie gesagt, die Anwendung von HTML ist meist viel verhältnismässiger...

6 Quellenangaben

1. www.w3c.org
2. www.edition-w3c.de
3. www.xml-magazin.de

PPS-Seminar
Grundlagen der Internet-Technologie, WS 02/03

SSL, SHTTP – sichere Kommunikation

Emmanuel Corboz
corboze@ee.ethz.ch
13. Dezember 2002

1 Einleitung

Im XXI. Jahrhundert ist das Internet ein nicht mehr wegdenkbares Hilfsmittel für unsere Gesellschaft. Dies geht vom einfachen surfen eines privaten Users bis zu finanziellen Transaktionen wie E-Commerc oder E-Banking oder Datenaustausch von Hochschulen, Börsen, Sicherheitsdiensten,…

Der Datenaustausch wird dank moderner Technologie (Cable, ADLS, DSL, Glasfaserleitung,…) immer schneller und effizienter, gleichzeitig werden auch aber die Anforderungen immer größer. Es wird vor allem immer einen immer besserern Quality of Service verlangt. Die Sicherheit im Internet ist ein Punkt der im Quality of Service beinhaltet ist.

Die Sicherheit im Internet wird immer schwieriger zu bewerkstelligen da das Internet von Tag zu Tag wächst und vor allem in immer mehr Applikationen Verwendung findet. Das Internet ist heutzutage schon nicht mehr „kontrollierbar“. Somit wird es immer schwieriger einen sicheren Datenaustausch zu gewährleisten.

Um die Sicherheit im Internet für die Massen an Applikationen die heute gebraucht werden zu gewährleisteten, werden Vier wichtige Voraussetzungen verlangt:

- Integrität
  Die Daten müssen unangetastet von einem Punkt zum anderen gelangen.

- Vertraulichkeit
  Die Daten dürfen nur von bestimmten Personen gesichtet werden.

- Authentizität
  Die versendeten Daten müssen vom Empfänger einstimmig und zweifelsfrei über seinen Inhalt und Sender bestimmt werden können.

- Nichtabstreitbarkeit
  Der Copyright oder die Herkunft der Daten müssen einer Drittperson nachweisbar sein.

Es wird nicht immer die absolute Sicherheit verlangt. Dem privaten User der im WWW surft reicht wenn seine Informationen bis zu einem gewissen Grad geschützt sind. Es ist aber klar dass bei finanziellen Transaktionen die Verbindung und die Daten effizient geschützt sein müssen.

Die Sicherheit im Internet wurde seit den Anfängen nicht zur Seite gelassen hat aber seit der Erfindung der WWW und der wirtschaftlichen Wichtigkeit der Internet stetig zugenommen.

Beim heutigen Stand der Technik gibt es Zwei Unsicherheitsfaktoren bei denen in Fragen kommenden angebotenen Diensten die durch Angreifer (Hacker) ausgebeutet werden können:

- Die Daten im Internet werden meist unverschlüsselt versendet. Das heißt dass die Daten an gewissen Stellen wie POP3 – Ports mit gewissen Softwares (Sniffer, Brute Force Attack,…) einfach „abgehört“ werden können.

- Da die Datenübertragung größtenteils unverschlüsselt erfolgt kann eine „man in the middle attack“ erfolgen. Diese Sorte von Angriff ist sehr häufig im Internet. Der Angreifer klinkt sich mittels geeigneter Software in die Verbindung seines potentiellen Opfers ein und „hört“ mit solange die Verbindung steht..

Um den Versand und Empfang von Daten im Internet effizient zu schützten gibt es Heutzutage Zwei Möglichkeiten:

- Verwendung einer sicheren Verbindung:
  Bei dieser Art Sicherheit wird eine sichere Transportarchitektur verwendet. Das heisst dass man der Anwendungsschicht nichts ändert aber auf der Transportebene Infrastruktur implementiert die eine sichere Verbindung gewährleistet. Diese Art Sicherheit kann dann von den Anwendungen benutzt werden die ein solches Verfahren unterstützten.

- Verwendungen einer sicheren Anwendungsebene:
  Hier ist der Fall das unsere Transportarchitektur nicht sicher ist. Somit müssen wir also auf der Anwendungsebene direkt in die verschiedenen Infrastrukturen der jeweiligen Anwendungen eine Sicherheit implementieren. Dies ist meist ein Protokoll in Form einer Software die unsere Daten verschlüsseln

Die sichere Kommunikation verlangt also mehre Hilfsmittel und verlangt gewisse Voraussetzungen (wie zum Beispiel ein orientierte Verbindung) um einen sicheren Datenaustausch zu gewährleisten.

2 Kryptographische Verfahren

Kryptographische Verfahren wurden schon bei den Römern gebraucht um „Daten“ zu Verschlüsseln und sicher von einem Punkt zum anderen zu transportieren. Spätestens seit der Erfindung und dem täglichen Gebrauch des Internet in unsere Gesellschaft gibt es auch für den Datenaustausch kryptographische Verfahren.

2.1 Klassifizierung

Man unterscheidet Zwei verschiedene Sorten von kryptographischen Verfahren; ein schwaches Verfahren das wenig Rechnenleistung und wenig Zeit braucht gegen ein starkes Verfahren das Aufgrund der vorhandenen Rechnenleistung, oder enormer Zeitbeanspruchung oder aus technischen Gründen nicht gebrochen werden kann.

Wenn der Verschlüsselungsalgorithmus geheim gehalten wird liegt meist ein schwaches Verfahren vor da meistens die Schlüssellänge nicht sehr groß ist.

Bei einem starken Verfahren wird der Verschlüsselungsalgorithmus als bekannt angesehen aber die Größe Schlüssellänge wird als Maß der Sicherheit angesehen.

Starke Verfahren unterscheidet man in Zwei Klassen: die symmetrischen Verfahren und die asymmetrischen Verfahren.

2.2 Symmetrische Verfahren (Secret oder Private Key Verfahren)

Bei symmetrischen Verfahren haben der Sender und der Empfänger den gleichen Schlüssel. Somit müssen Schlüssel und Algorithmus geheim gehalten werden. Dieses Verfahren wird auch Konventionell als Secret oder Private Key Verfahren genannt.

Symmetrische Verfahren werden in Zwei Klassen unterteilt: Block- und Stromchiffre Verschlüsselung.

2.2.1 Blockchiffren

Das Blockchiffre-Verfahren arbeitet wie es der Name sagt mit Textblöcken.

Das Dokument wird in Blöcke (festgelegte Größe durch den Algorithmus typischerweise 64 Bit) aufgeteilt. Diese Blöcke werden mit Hilfe der Kernfunktion verschlüsselt. Dies sieht folgendermaßen: nach der Eingabe eines Klartextes wird eine interne Rundfunktion aufgerufen und mehrmals durchlaufen (kommt auf den Algorithmus an) danach kommt eine Endbehandlung.

Die Rundenfunktion auch Kernfunktion genannt, zerwürfelt die Bits und ersetzte Teilblöcke durch andere Bitfolgen. Der Empfänger der Daten kann die Daten entschlüsseln in dem er mit Hilfe des Key und des Algorithmus das Ganze umgekehrt Laufen lässt.

Ein Beispiel für dieses schon in den 70er Jahren entwickeltes Verfahren ist der DES (56 Bit Schlüssel). Heute stehen größere Schlüssel wie der Tripple DES auch 3-DES (112 Bit) oder IDEA (128 Bit) zur Verfügung.

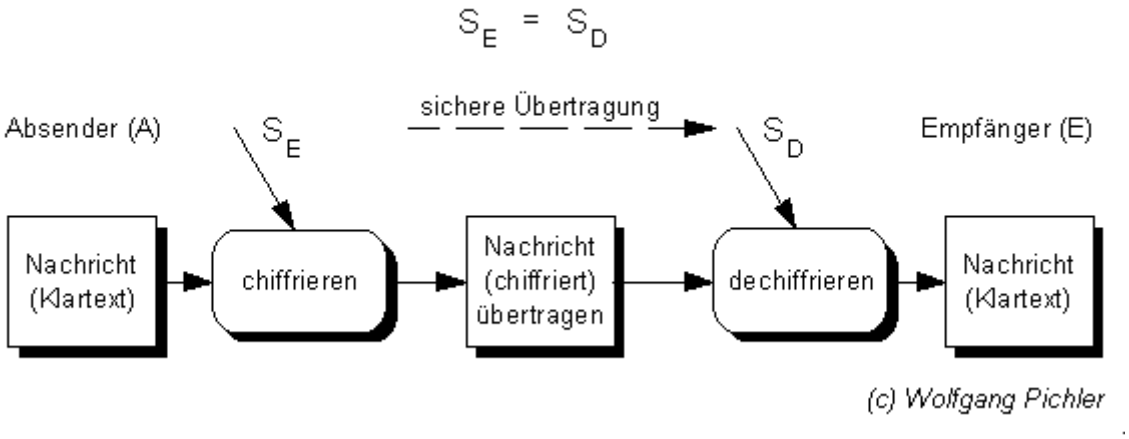
2.2.2 Stromchiffren

Das Stromchiffren Verfahren arbeitet im Gegensatz zu Blockchiffren nicht mit Blöcken sonder mit einzelnen Bits oder Byte. Der Grundlegende Unterschied zwischen diesen beiden Verfahren liegt bei der Transformationsfunktion die nicht konstant ist. Das heißt dass gleiche Textstücke unterschiedlich chiffriert werden. Die Positionen der einzelnen Stücke spielen innerhalb des Textes auch eine Rolle.

Eine geheime generierte Pseudozufallszahlenfolge (die abhängig vom Schlüssel und manchmal auch vom Klartextblock ist) und den einzelnen Bits des zu verschlüsselnden Klartextblockes werden durch einfache Funktionen verknüpft (meisten XOR).

Die Entschlüsselung wird mit Hilfe der Key und der umkehr Funktion (hier XOR) bewerkstelligt.

Dieses Verfahren wird in mehren Bereichen angewendet wo vor allem eine gute Verschlüsselung nötig ist aber nicht sehr viel Rechnenleistung vorhanden ist. Zum Beispiel das A5 Verfahren das ein 54 Bit Verschlüsselung generiert wird benutzt um Sprachdaten der GSM Netzte zu verschlüsseln. Es gibt auch größere Schlüssel wie die RC4 Verschlüsselung die meistens 128 Bit und größer ist.



2.3 Asymmetrische Verfahren (Public Key Verfahren)

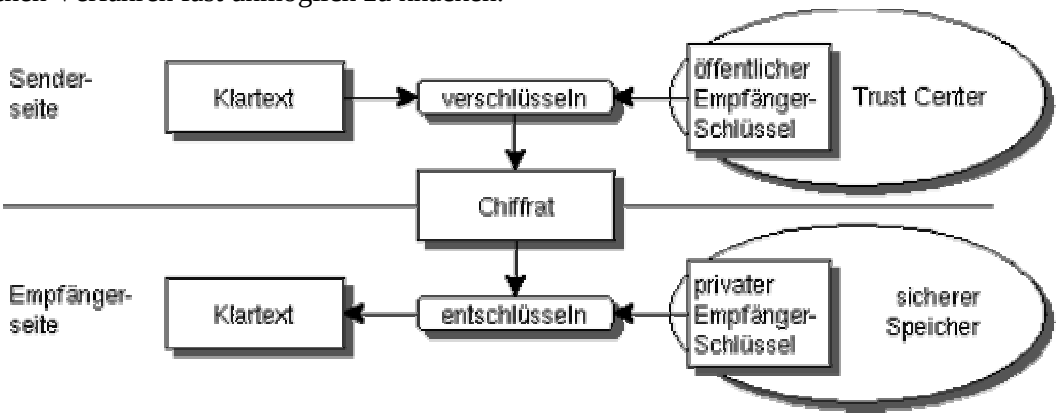
Die Ersten brauchbaren asymmetrischen Verfahren stehen erst seit 1982 zur Verfügung. Die asymmetrischen Verfahren bestehen im Wesentlichen darin dass es einen spezifischen Verschlüsselungs- und der Entschlüsselungs- Key gibt die nicht die gleichen sind. Diese Zwei Keys stehen in einem hochgradig komplizierten mathematischen Zusammenhang aus dem den einen Key nicht vom anderen abgeleiten werden kann. Das heißt es nützt einem Angreifer nichts das er den Algorithmus oder einen Key kennt um eine Attacke zu vollführen. Dieses Verfahren ist sehr Rechenintensiv, dies von allen Seiten her.

Bei diesem Verfahren gibt es mehre Möglichkeiten die verschiedenen Keys ungesichert zu senden oder zu veröffentlichen.

Mit Hilfe dieses Verfahrens kann man eine Key zum Beispiel ohne großes Risiko veröffentlichen um zum Beispiel die Herkunft von Daten zu beweisen da es ja nur einen anderen (Private-) Key gibt der zu diesem Key passt.

Es können so auch Daten sicher versendet werden in dem der Empfänger dem Sender seinen Public Key gibt, der Sender die Daten mit Hilfe dieses Key verschlüsselt und dann sendet. Da nur der Empfänger den dazu passenden Key hat, ist dies eine sehr sichere und effiziente Methode Daten über ein unsicheres Netz oder eine nicht gesicherte Verbindung zu versenden.

Auch dieses Verfahren ist nicht absolut sicher. Da aber heute die technischen Möglichkeiten immer größer werden, gibt es für dieses Verfahren Schlüsseln und Algorithmen die fast unmöglich zu knacken sind. Zum Beispiel gibt es heutzutage Schlüssel wie zum Beispiel der RSA Schlüssel die eine Länge von 1024 Bit und mehr besitzen. Solche Algorithmen sind in einem asymmetrischen Verfahren fast unmöglich zu knacken.



2.4 Hashfunktionen

Viele Verschlüsselungs-Protokolle haben als festen Bestandteil eine Hashfunktion implementiert. Die Hashfunktion kann mit einer Prüfsumme verglichen werden oder auch einem Finger-Abdruck.

Bei einer Hashfunktion wird zum Beispiel ein Text oder Dokument als Input angegeben. Die Hashfunktion generiert nun eine feste Länge die normalerweise 128 Bit groß ist (kann auch 160 Bit sein). Dies Länge ist der Output und kann nicht mehr zurückverwandelt werden. Versucht nun jemand das Verschlüsselte Dokument zu knacken wird dieser Hashwert verändert. Kontrolliert man nun die Prüfsumme des Original Hashwert und die des modifizierten Hashwert, merkt man ob es sich entweder um die gleichen Dokumente handelt oder nicht oder ob das Dokument während seiner Versendung „angegriffen“ wurde. Natürlich müssen gewissen Bedingungen erfüllt damit die Hashfunktion auch funktioniert.

Die Hashfunktionen laufen auch mittels eines Algorithmus wie zum Beispiel MD 5 oder SHA – 1.

2.5 Digitale Signaturen

In Punkt 2.3 wurde erklärt das mittels einem Public- oder Private- Key der Verfasser der Daten ermitteln lässt. In Punkt 2.4 wurde erklärt wie mittels einer Hashfunktion ein Datenpaket auf seine Integrität kontrolliert werden kann.

Nun kann der Sender sein Datenpaket senden und diesem eine abgesetzten Signatur implementieren. Das heißt der Sender ermittelt der Hashwert, verschlüsselt ihn und fügt ihn dem Datenpaket zu. Der Empfänger entschlüsselt nun Daten und Hashwert. Er ermittelt mit der gleichen Hashfunktion den dazugehörenden Wert und vergleicht mit dem Wert den er bekommen hat. Sind die Werte gleich so wurden die Daten nicht „angegriffen“ ansonsten liegt ein Angriff an.

2.6 Zertifikate

Zertifikate werden gebraucht um zwischen einem Public- Key und einer Identität einer Verbindung herzustellen. Dies heißt dass diese Verbindung von einer gewissen Zertifizierungs-Stelle bestätigt wird. Diese Stelle bestätigt die Zugehörigkeit eines Public- Key zu einer bestimmten Identität. Das es aber keine solche „offiziellen“ Stellen gibt muss jeder „User“ selber entscheiden wem er ein solches Zertifikat ausstellt und wem nicht. Die Vertrauenswürdigkeit eines solches Zertifikates lann sehr umstritten sein.

3 SSL – Secure Socket Layer

Secure Socket Layer abgekürzt SSL ist eine Technik um HTTP-Kommunikationskanäle zu sichern. Diese Technik wurde von der Firma Netscape für Ihren gleichnamigen Browser entwickelt. Die IETF (Internet Engeneering Task Force) führ den Standardisierungsprozess unter dem Namen Transport Layer Security, abgekürzt TLS.

Die Technik basiert auf einem Public-Key Verfahren. Dabei wird zwischen der Anwendungsschicht, HTTP und der Transportschicht, TCP/IP eine neue Schicht, das SSL eingefügt.

Das Implementieren von SSL ist gestaltet sich als problemlos und kann auch noch moduliert werden mit zum Beispiel zusätzlichen Sicherheitsfunktionen.

3.1 Ziele von SSL

Das Hauptziel von SSL ist es eine sichere Kommunikation über eine unsichere Anwendung zu gewährleisten. Das SSL Protokoll hat Drei Grundlegende Eigenschaften:

- Verbindungssicherheit
  Um eine Verbindung zwischen Client und Server aufzubauen ist zuerst ein Handshake nötig. Mit Hilfe dieses „Händeschütteln“ und einem Verschlüsselungsverfahren wird ein geheimer Schlüssel definiert. Die Verbindung kann dann über eine symmetrische oder asymmetrische Datenverschlüsselung erfolgen. Es werden zunehmend asymmetrische Verfahren benutzt.

- Optionale Authentifizierung
  Die Identität seines Kommunikationspartners bei einer Verbindung kann mit Hilfe eines asymmetrischen Verfahrens, mit Hilfe eines Public-Key ermittelt und authentifiziert werden.

- Zuverlässigkeit der Verbindung

Das Ziel einer sicheren Verbindung ist das diese auch zuverlässig ist. Dies kann mit Hilfe eines MAC (Message Authentifiaction Code) bewerkstelligt werden. Bei dieser Überprüfung mit Hilfe einer Hashfunktion werden die Daten auf Ihre Integrität überprüft.

Das SSL Protokoll hat Vier Hauptziele die hier nach „Wichtigkeit“ aufgeführt sind:

- Kryptographische Sicherheit
  SSL soll eine sichere Verbindung zwischen Zwei Kommunikationspartner aufbauen die nicht durch einen Dritten eingesehen werden kann.

- Interoperabilität
  Unabhängige Programmiere sollen Anwendungen entwickeln können die SSL einschließen und in der Lage sind Verschlüsselungsparameter auszutauschen ohen jeweils den Code der anderen zu kennen.

- Erweiterbarkeit
  SSL, versucht einen Rahmen herzustellen, innerhalb dessen sich neue Verfahren sowohl zur Erstellung von Public-Keys als auch zur Verschlüsselung größere Datenmengen den Erfordernissen entsprechend miteinander verbinden lassen. Dabei werden auch zwei „Sekundärziele“ erreicht:

  - Es muss keine neues Protokoll entwickelt werden (damit werden auch keine neuen Sicherheitslücken gezeugt).
  - Es muss keine neue Sicherheitsbibliothek implementiert werden.

- Relative Wirksamkeit
  Kryptographische Verfahren, insbesondere asymmetrische Verfahren die mit Public-Keys arbeiten sind meistens sehr rechnenintensiv vor allem bei größeren Schlüsseln. SSL besitzt darum ein Caching System um neu Aufzubauende Verbindungen zu reduzieren. Somit wird werden die Verbindungen rund die Netzwerkauslastung reduziert.

Wir haben hier ein effizienten und effektiven Schutz vor einem Belauschungsangriff also einem „man in the middle attack“ aber keinen Schutz vor einem direkten Angriff auf das System durch eine Mittelsperson.

3.2 Verbindungsaufbau

Das SSL Protokoll kann allgemein drei verschiedene Arten von Verbindungen zwischen einem Client und einem Server aufbauen die sich aber nur in einem wesentlichen Punkt voneinander unterscheiden, der Authentifizierung. Diese muss dann ein akzeptables Zertifikat besitzen das auch von einer annehmbaren Authentifizierungsstelle ausgestellt wurde.

- Anonyme – Verbindung
  Bei dieser Art von Verbindung ist weder der Client noch der Server authentifiziert.

- Server – Authentifizierte – Verbindung

Diese Art von Verbindung geht nur in eine Richtung. Nur der Server muss sich authentifizieren aber nicht der Client. Das heißt der Client weiß das der Server ein akzeptables Zertifikat besitzt und kenne auch seine Identität.

- Client – Server – Authentifiziert – Verbindung
  Diese Verbindung verlangt von Beiden Seiten, Client und Server eine Authentifizierung mit einem akzeptablen Zertifikat beiderseits.

3.3 Funktionsweise

SSL funktioniert in 2 Phasen. Zuerst initialisiert der Client einen Verbindungsaufbau, ein so genanntes „Handshake".Bei diesem „Handshake" wird der Status der Beiden Seiten ermittelt. Dabei können auch noch zusätzliche Authentifizierungsmöglichkeiten eingeschaltet werden. Nach der ersten Initialisierung wird nun aus der ClientKeyExchange Nachricht einen Sitzungs-Schlüssel (SSL basiert auf einem Sitzungskonzept) errechnet der nur für diese Sitzung gültig ist. Nachdem Client und Server die Schlüssel ausgetauscht haben und beiderseits eine Kontrolle (bei Client mit Hilfe der CertificateVerfiy uns beim Server mit Hilfe der ClientKeyExchange) gelaufen ist, schließt die ChangeCipherSpec Nachricht die Aushandlungen ab und überprüft deren Erfolg. Bei Erfolg wird die Verbindung aufgebaut.

Das Verschlüsselungsverfahren ist wesentlicher leistungsfähiger während dem Schlüsselaustausch als nachher.

Falls nach einer gewissen Zeit einer der Seiten die Sicherheit als nicht mehr vertretbar annimmt so wird die Verbindung unterbrochen und der ganze Prozess fängt neu an.

3.4 Kombinieren von SSL und HTTP => HTTPS

Man kann einer normalen, unsicheren TCP-Verbindung das SSL Protokoll in der Transportschicht implementieren. Das heißt dass der Client schlussendlich eine sichere SSL Verbindung zum Server aufbaut. Dies wird bewerkstelligt das der URL-Präfix nicht mehr „http" sondern „https" heißt. Da aber diese Art von Verbindung nicht sehr gebräuchlich ist, müsste der User einzeln jedes Mal das SSL Protokoll implementieren.

4 Secure HTTP => SHTTP

Die Alternative zu HTTPS ist Secure HTTP, S-HTTP, SHTTP. Die Beiden Protokolle haben sehr ähnliche Verschlüsselungsverfahren aber auch SHTTP ist nicht sehr verbreitet.

SHTTP basiert auf ein auf HTTP basierendes Nachrichtenformat. Dieses Nachrichtenformat wird aber durch verschiedene Sicherheitsmerkmale erweitert.

Der Hauptgedanke bei SHTTP ist das die HTTP Nachricht gekapselt wird und mit einem Header bestückt wird der alle Informationen enthält.

Da SHTTP nicht exakt auf der HTTP-Syntax beruht ist es egal welche http Version benutzt wird.

5 SSH – Secure Shell

SSH wurde 1996 von Tatu Ylone entwickelt. Dieses aus einer Software-Suite und einem Protokoll bestehende Kryptographieprogramm ist Heute ein viel benutztes Programm um die unterschiedlichsten Kommunikationskanäle kryptograpisch Abzusichern.

Die Verschlüsselung basiert auf einem asymmetrischen Verfahren, kombiniert mit einem symmetrischen Private-Key Verfahren.

1998 wurde die Version SSH2 herausgeben. Die Neue Version ist schneller, flexibler der Public-Key Authentifizierung. SSH2 wurde mit neuen Features versehen wie mehrfach Abfragung von Pivate- oder Public-Keys, sftp (Secure FTP Deamon),…

Die Nutzung von SSH1 und SSH2 ist vor allem Lizenzbedingt. SSH1 kann noch von Firmen zu Firmenzwecken gebraucht werden während SSH“ nur noch freie Nutzung für Bildungseinrichtungen erlaubt (zum Beispiel: ETHZ).

Ein Gruppe von Entwickler hart sich nun daran gemacht ein freier Version von SSH zu entwickeln. Die momentane Version von Open-SSH unterstützt noch nicht aber bald SSH2.

Mit SSH kann man Tools wie Telnet oder r-host komplett ersetzten. SSH hat ein grosses Potential um in anderen Einsatzgebieten gebraucht zu werden. Eines wäre das Port-Forwarding. Auch in vielen anderen TCP/IP basierten Anwendungen könnte SSH zum Einsatz gebracht werden.

5.1 Verbindungsaufbau / Funktionsweise

- Client sendet dem Server einer Initialisierung einer SSH Sitzung.
- Der Server generiert ein Schlüsselpaar und startet einen Demaon.
- Nun tauschen Client und Server ihre Protokollversionen aus.
- Der Server sendet das erzeugte Schlüsselpaar und der Client vergleicht es mit seiner Liste (sind die Schlüssel nicht vorhanden kann der Client sie auch integrieren, dies kann aber gefährlich sein da die Serveridentität nicht bescheinigt ist).
- Ist der Client mit dem Schlüsselpaar zufrieden so erzeugt er einen Sitzungsschlüssel den er mit dem anderen Schlüsselpaar zurücksendet. Wobei der Client mitteilt was für ein kryptographisches Verfahren er anwenden möchte.
- Der Server kann nun mit dem von ihm erzeugten Schlüsselpaar den Sitzungsschlüssel entschlüsseln.
- Die Verbindung verläuft jetzt verschlüsselt.
- Nun muss sich der Client mittels Passwort- oder Public-Key-Verfahren Authentifizieren.
- Je nach Einstellung der SSH Version können Sicherheitsparameter nach gewissen Kriterien verändert werden.

6 Zusammenfassung / Schlussfolgerungen

Das Verlangen an einer sicheren Kommunikation wächst und wird von den Internet-Usern immer mehr gefragt. Dies wird aber mit zunehmender Größe des Internets immer schwieriger. Sichere Kommunikation ist Heute auch Notwendig da immer mehr LAN's, WANS's und andere Netze entstehen. In den letzten Jahren wurden sehr viele Anwendungen vor allem aus wirtschaftlichen Gründen ins Internet verlegt. Dabei werden seit Jahren immer bessere Verschlüsselungsverfahren,… entwickelt. Der Feind schläft aber auch nicht und er lernt dazu. Ein Beweis sind die Milliarden Hohen Reparaturkosten die jedes von diesen Angreifern erzeugt werden. Man könnte sagen sie wachsen mit der Herausforderung. Schon Heute haben Hacker es geschafft die vermeidlich sicheren SSH oder SSL Protokolle bis zu einem gewissen Grad zu knacken.

Um der hohen Anforderung an sichere Kommunikation folge leisten zu können müssen vier Voraussetzungen da sein: Integrität, Vertraulichkeit, Authentizität, Nichtabstreitbarkeit. Wenn man von diesen Voraussetzungen ausgeht um neue kryptographische Methoden zu entwickeln, sollte auch das Maß an sichere Kommunikation erfüllt werden.

Referenzen

1. Abgegebene Unterlagen durch den Assistenten.
2. http://www.klammeraffe.org/~brandy/artikel/ssh.html/
3. http://www.ssl.de/
4. http://www.ietf.org (/rfc/rfc2660.txt?number=2660)
5. http://www.demonium.de/th/home/studium/work/ssl/node3.html/
6. http://developer.netscape.com/docs/manuals/security/sslin/contents.htm/
7. http://www.netscape.com/

PPS-Seminar
Grundlagen der Internet-Technologie, WS 02/03

E-Mail
Post der Zukunft

Patrick Moser
moser_patrick@hotmail.com
18 Dezember 2002

1 Senden und empfangen von E-Mails

Zuerst ein bisschen Geschichte, man konnte schon gegen ende der 60er Jahre nachrichten von User zu User senden aber leider nur auf der selben maschine. Im Jahre 1971 erfand ein Ingenieur namens Ray Tomlinson das @ um eine Nachricht auf andere Server zu schicken. Nach dieser Erfindung stand dem erfolg der E-Mail nichts mehr im Wege. Wie üblich wurde das ganze am Anfang nur in Wissenschaftlichen kreisen gebraucht, aber mittlerweile brauchen es fast alle.

1.1 SMTP / ESMTP / X400

Das sind die Protokolle die man braucht um E-mails zu senden, die meist genutzten sind SMTP und ESMTP, wehrend dessen X400 fast in Vergessenheit geraten ist da sich die X400er maschinen nicht durchsetzen konnten obwohl das Protokoll funktionell überlegen ist.

1.1.1 SMTP – Simpel Mail Transfer Protocoll

SMTP ist das Basisprotokoll das wir fürs senden von E-Mails brauchen, es ist ein ziemlich einfaches protokoll und funktioniert über Port 25.

Die verschiedene SMTP Comands sind:

- HELO = ist das erste command das man bei einer connection mit einem SMTP Server eingibt, es ist so was wie das hallo am Telefon, man bestetigt damit das der Server bereit ist und man wirklich connected ist.
- MAIL = nach diesem command schreibt man noch die Adresse des Absenders hin. absender@alpha.com
- RCPT = Nach diesem command schreibt man die Adresse des Empfänger hin. empfänger@beta.com
- DATA = Nach diesem command interpretiert der Server alles als daten, und hier schreibt man auch die Memo-Headers rein (Date, Subject, To, CC, From)
- SEND = Sendet die E-Mail zu einem Terminal und nicht in eine Mailbox.
- SOML = Sendet die E-Mail zu einem Terminal oder mailt sie in eine Mailbox.
- SAML = Sendet die E-Mail zu einem Terminal und mailt sie in eine Mailbox.
- EXPN = nach expand schreibt man noch eine Adresse hin und wen es eine mailing list ist zeigt der Server alle Adressen der liste an.
- RSET = Löscht alle eingegebene Daten und man kann von neuem anfangen als währe nie was passiert.
- HELP = Nicht alle Server unterstützen dieses command aber wen ja bekommt man eine liste der akzeptierten commands und weiter auch wie man sie braucht.
- NOOP = Erzeugt einfach eine OK meldung des Servers und nichts mehr, kann man brauchen um zu schauen ob man immer noch connected ist zb.
- TURN = Da SMTP meistens zwischen zwei Servern läuft kann man mit dem command TURN die rolle der selben invertieren, also derjenige der vorhin die mails sendete empfängt sie jetzt und der andere sendet sie.
- QUIT = mit diesem command beendet man die Verbindung mit dem Server.

Für mehr infos über die einzelnen commands und das SMTP Protokoll verweise ich auf die URL: http://www.freesoft.org/CIE/RFC/821

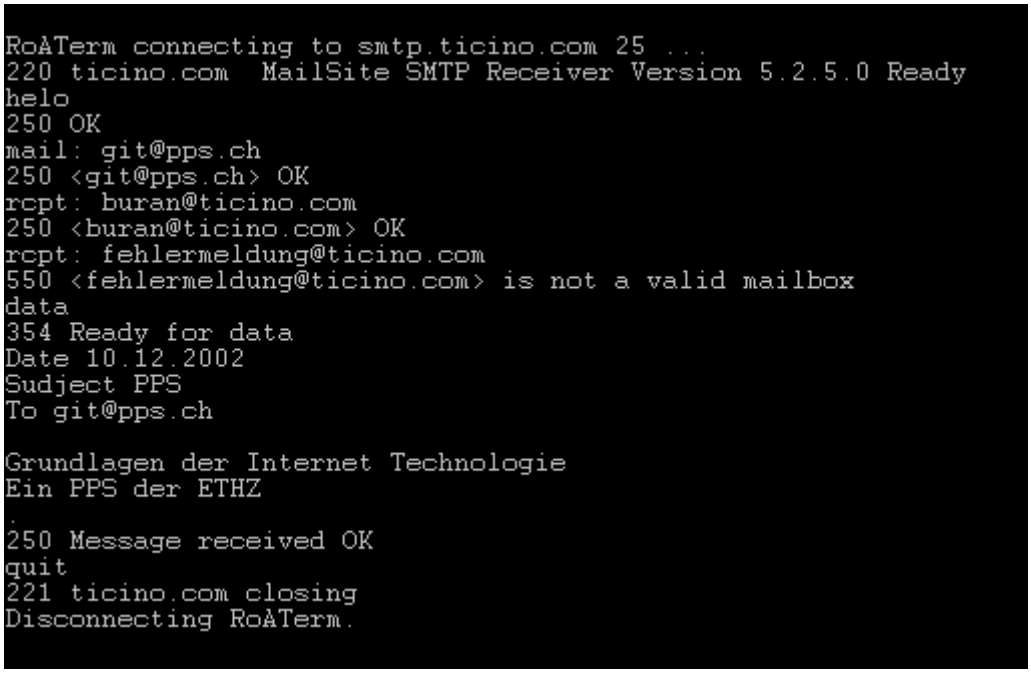


Fig 1. Hier sieht ihr eine typische SMTP Sitzung.

1.1.2 ESMTP – Extended Simpel Mail Transfer Protocoll

Extended SMTP ist eine Ergänzung zu SMTP und ermöglicht die einfache Implementierung dieses Protokolls. Die beliebtesten Implementierungen sind:

- 8-Bit Zeichensatz = Hiermit kann man auch nicht ASCII Zeichen versenden, und damit auch Bilder und verschiedene Attachments.
- Ankündigung der E-Mail grösse = Da die meisten Mailboxses eine begrenzte Kapazität haben kann der Server eine E-Mail verweigern.
- Ermöglicht das teilen von E-Mails = Da E-Mails immer grösser und grösser werden ist es sinnvoll eine E-Mail zu zerlegen und in einzelnen Paketen zu senden, und wen ein Paket nicht ankommt muss man nicht die ganze mail neu senden sonder nur das beschädigte Paket.

1.2 POP / IMAP / P7

Das sind die 3 Protokolle um E-Mails, zu bekommen, POP3 nud IMAP sind die am meist verbreiteten, und da P7 auch auf X400er maschinen läuft ist es auch in Vergessenheit geraten.

Bevor ich aber von POP und IMAP rede möchte ich noch kurz die 3 verschiedene arten des E-Mails lesen erläutern.

- Offline = Hierbei verbindet sich der Client periodisch mit dem Server, transferiert die Daten von dem Server auf die eigene HD Wo man sie danach verwaltet. Die Mails werden aber auf dem Server gelöscht so das ein anderer Client diese Mails nicht mehr lesen kann.
- Online = Mit dieser Methode greift man auf den Server zu und verwaltet die E-Mails dort, sodass auch andere Clients die Mails sehen können.
- Getrennt = Auch hier wie im Offline ladet man die Mails auf die eigene HD aber man Löscht sie nicht. Man bearbeitet die Mails auf der eigenen HD und beim nechsten mahl werden die Mails auf dem Server geupdated. Somit bleiben die Mails auch noch für andere Clients sichtbar.

1.2.1 POP – Post Office Protocoll

Da unsere Computer ja nicht immer mit dem Internet connected sein können sind die Mailboxses auf Server, und um die Mails zu lesen muss man sie irgendwie von dort abholen, hierzu gebt es das POP Protokoll. Wie SMTP ist es ein ziemlich einfaches Protokoll da es auch mit einigen wenigen commands auskommt.

Authorization level:

- USER = Hier muss man die gewünschte Mailbox angeben
- PASS = Hier die Password
- QUIT = und mit diesem command beendet man die Verbindung

Minimal Level:

- STAT = Zeigt eine Statistik der Mailbox an
- LIST = Zeigt eine liste der einzelnen E-Mails
- RETR = Sendet die gewünschte E-Mail zu dem Client
- DELE = Löscht die gewünschte E-Mail
- NOOP = Wie im SMTP erzeug es nur ein OK des Servers.
- RSET = Unmarkiert alle E-Mails die als Deleted markiert sind.

Optional level:

- TOP x = Zeigt die ersten x Zeilen einer gewünschten E-Mail oder von allen.
- UIDL = Zeigt die Speicheradresse einer E-Mail an, Diese Adressen Sind Serverabhängig und jede E-Mail hat seine eigene so das man sie nicht verwechseln kann.

Für mehr infos über die einzelnen commands und das POP Protokoll verweise ich auf die URL: http://www.faqs.org/rfcs/np.html#POP

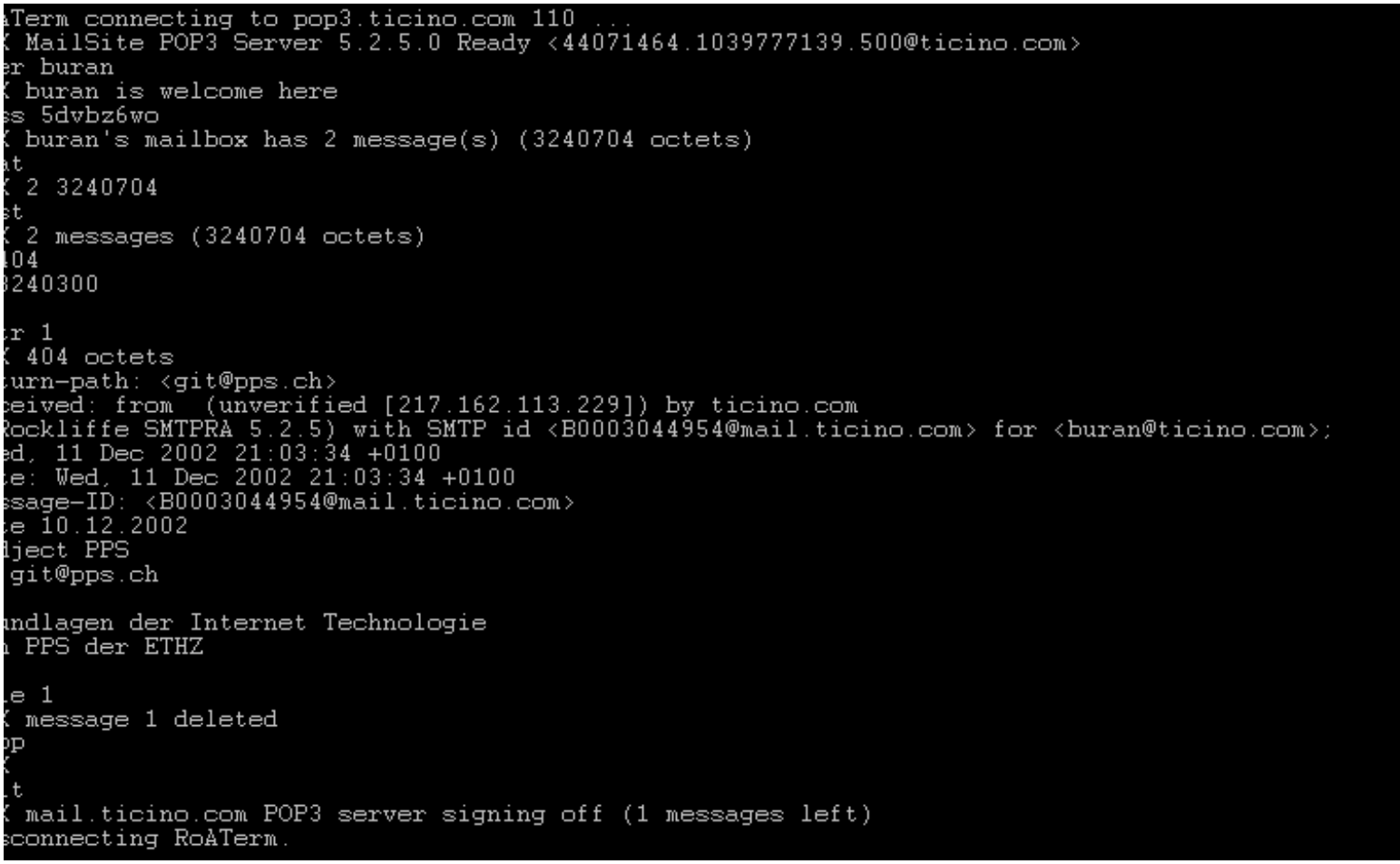


Fig 1. Hier sieht ihr eine typische POP3 Sitzung.

1.2.2 IMAP – Internet Message Access Protocoll

Da POP wie ihr gesehen habt nur für Offline betrieb brauchbar ist brauchte man ein neues Protokoll, und das ist IMAP. IMAP beinhaltet aber unterstützt dabei noch mehrere andere Sachen. Die Verschiedene Vorteile kann man wie folg beschreiben.

- Basilare Ordner Bearbeitung = IMAP kann ohne Probleme Ordner kreieren und verwalten
- Unterstützt gemeinsame Ordner = IMAP erlaubt das ein Ordners von mehreren Benutzer geteilt wirt, und damit wissen andere Clients was verändert wurde und wie es verändert wurde.
- Abholen von MIME Teilstücke = DA E-Mails immer grösser werden erlaubt IMAP das abholen von MIME Teilstücken, so das man nciht die ganzen E-Mails runterladen muss wen man die Post nur kurz durchblättern will.

Da IMAP ein sehr komplexes Protokoll ist werde ich hier die Funktionsweise nicht erläutern. Für mehr Informationen über IMAP und wie es funktioniert gebe ich wieder eine URL an: http://www.faqs.org/rfcs/np.html#IMAP

2 Sicherheit

Die Sicherheit ist sicherlich ein grosses Thema heutzutage, leider können die obengenante Protokolle unsere Daten nicht schützen. Es gibt aber schon andere Protokolle wie S/MIME und openPGP. Die Funktionsweise dieser Protokolle wurde schon in dem Beitrag zur Sicherheit von Emmanuel Corboz ausfürlich dokumentiert, Trotzdem gebe ich noch ein par URL's dazu.

- S/MIME = http://www.faqs.org/rfcs/rfc2633.html + http://www.faqs.org/rfcs/rfc2632.html
- openPGP = http://www.faqs.org/rfcs/rfc3156.html

3 Spam

Als Spam definiert man viele ungewollte E-Mails die einem den Account füllen, meistens ist es Werbung für die unterschiedlichsten Sachen. Ich selber definiere die Kettenbriefe und Haox-Botschaften die im Internet zirkulieren auch als Spam aber das ist auch nur meine Meinung. Spam ist sicherlich was nerviges und manchmal auch Offensiv, aber es ist eigentlich wie die Werbung im Fernsehen oder die Plakaten auf der Strasse, die Werbung in Zeitschriften und im eigenen Briefkasten.

Werbung nimmt man normaler weise als Beliebtheitsmesser eines Mediums, desto mehr Werbung desto mehr ist das Medium beliebt, daher muss man Spam ja als anzeige nehmen das E-Mail immer noch ein sehr beliebtes Medium ist und sicherlich bleiben wirt.

Gegen dem Spam kann man etwa so wenig machen wie gegen die Werbung im Fernsehen, das einfachste ist die Werbungen einfach zu löschen ohne sie anzuschauen, weil an den Absenderserver zu schreiben nützt meistens auch da sie nichts dafür können.

Ich hoffe das diese kurze Zusammenfassung über E-Mail jemanden von nutzen sein wirt, und wen noch fragen aufbleiben meine E-Mail ist: moser_patrick@hotmail.com , aber auf der URL http://www.faqs.org/rfcs/np.html findet man alle RFC's und nicht nur über E-Mail.

Referenzen:

1. http://www.howstufworks.com
2. http://www.faqs.org/rfcs/np.html

ETH Zürich
PPS-Seminar WS 02/03
Grundlagen der Internet-Technologie

WAP und WML

Patrick Fauquex
fauquexp@ee.ethz.ch
20. Dezember. 2002

1. Das WAP

1.1 Einleitung

In den letzten Jahren gewannen die Handys, Notebooks und die PDAs enorm an Bedeutung und ein Ende dieses Booms ist immer noch nicht in Sicht. Solche mobilen Alternativen zum gewöhnlichen Computer sind heute aus unserem Leben, sei es nun in der Freizeit oder im Beruf, kaum mehr wegzudenken.
Parallelen zu diesem Boom findet man auch in der Entwicklung des Internets wieder. Auch ein Internetanschluss gehört in der heutigen Zeit zur Grundausstattung jedes Haushalts.
Es war daher unumgänglich, neue Wege zu entwickeln, um das Internet mit den mobilen Alternativen zu kombinieren und somit die Datenübertragung vom Internet auf das Handy (u.a.) zu ermöglichen. Eine Möglichkeit bietet das WAP (Wireless Application Protocol), das trotz mehreren Problemen bei der Entwicklung schliesslich im Jahr 1998 eingeführt wurde.

1.2 Das WAP Forum

Im Jahre 1997 legten die damals vier grössten Mobilfunkhersteller (Nokia, Phone.com, Motorola und Ericsson) die Grundlagen für die Entwicklung des WAP und gründeten das WAP Forum.
Dieses Forum ist eine nicht gewinnorientierte Verbindung verschiedener Telekommunikationskonzerne, der heute bereits mehr als 90% aller Technologieunternehmen angehören. Das Ziel bei der Entwicklung des WAP war, einerseits die wenigen Ressourcen des mobilen Gerätes geschickt zu verwenden und anderseits die Einschränkungen der Geräte durch die Funktionalität des Netzwerkes zu kompensieren. Kurz gesagt, man wollte auf dem mobilen Endgerät so wenig wie möglich verändern und die dafür erforderliche Intelligenz aufs Netzwerk verlagern.
Zudem ist das WAP Forum für die einheitliche Umsetzung vom WAP in den verschiedenen Handys verantwortlich und definiert, wie die einzelnen Geräte miteinander kommunizieren sollen. Damit kein Hersteller die Oberhand gewinnen kann, werden auch die technischen Fortschritte berücksichtigt und alle 6 Monate eine Liste der aktuellen Spezifikationen präsentiert.

Als schliesslich 1998 das WAP 1.0 eingeführt wurde, war ein weiterer Meilenstein in der drahtlosen Netzwerkverbindung gelegt worden. Dieses erste Protokoll sollte vor allem der Weiterentwicklung dienen, und so war erst mit dem WAP 1.1 und dem WAP 1.2 Ende 1999 der Standard genügend stabil und leistungsfähig, so dass fast überall WAP-Dienste eingeführt wurden.
Mit der Einführung des WAP 2.0 2001 wurden wieder völlig neue Richtlinien gesetzt. Den Hersteller gelang es ein Protokoll zu entwickeln, das bereits die meisten Internetprotokolle lesen kann.

1.3 Probleme bei der Entwicklung

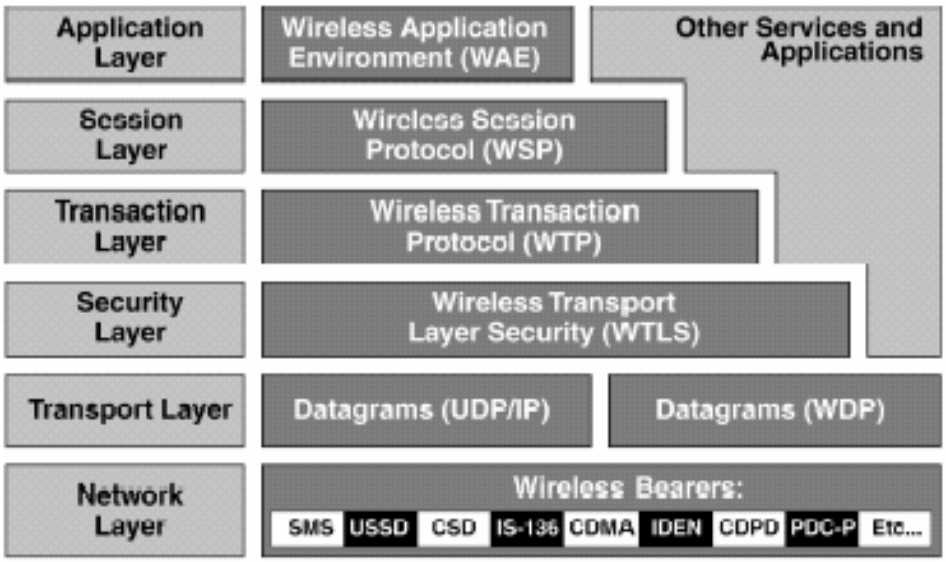
Von Anfang an sah man ein, dass das Übernehmen der Internetsprache HTML keinen Sinn machen würde, da technische Probleme wie

- schwächere CPUs
- begrenzter Speicherplatz
- zu kleine Displays
- unterschiedliche Handyeinstellungen (Bsp. Tastatur)
- niedrige Übertragungsrate und zudem keine Garantie für eine stabile Verbindung
- niedrige Auflösung und somit Graphiken nur schlecht einsetzbar

dies zu einer unlösbaren Aufgabe machen würden. Schliesslich entschloss man sich eine neue Programmiersprache zu entwickeln, die speziell auf das WAP zugeschnitten war. So entstand schliesslich die Sprache WML.

1.4 WAP-Aufbau

Der WAP-Aufbau basiert, wie auch diverse andere Protokollfamilien (Bsp: TCP/IP), auf einem Schichtenmodell. Der sogenannte „WAP-Stack“ ist in 5 Schichten unterteilt.



Application Layer (Anwendungsschicht)

Den Application Layer stellt das Wireless Application Enviroment (WAE) dar. Dessen Aufgabe ist die Bereitstellung von Mitteln zur Entwicklung von Anwendungen und Diensten, die über alle Datenstandards der mobilen Kommunikation hinweg tragbar sind. Primärer Nutzniesser dieser Schicht ist Micro-Browser, der im webbasierten Material navigiert.

Session Layer (Sitzungsschicht)
Das Session Layer regelt mit Hilfe des Wireless Session Protocol (WSP) den Ablauf einer Sitzung. Dies sind folgende Schritte:

- eine Sitzung starten
- Inhalte austauschen
- Sitzung beenden

Zusätzlich kann eine Sitzung auch abgebrochen und wieder aufgenommen werden.

Transaction Layer (Transaktionsschicht)
Im Transaction Layer befindet sich das Wireless Transaction Protocol WTP. Als Transaktion gilt die Kommunikation zwischen dem Initiator einer Anfrage und dem Antwortenden. Das WTP ermöglicht drei verschiedene Datenübertragungsmethoden:

- unzuverlässige Einweganfrage (das Mobilgerät sendet eine Anfrage, es wird aber nicht garantiert, dass diese beim Server ankommt oder beantwortet wird)
- zuverlässige Einwegsanfrage (das Protokoll garantiert den Empfang der Nachricht beim Server)
- zuverlässige Zweiwegsanfrage und –antwort (hier gilt die Übertragung erst dann als erfolgreich abgeschlossen, wenn am Mobilgerät die Antwort des Servers eingetroffen ist)

Security Layer (Sicherungsschicht)
Wireless Transport Layer Security (WTLS) ist eine optionale Schicht, die Verschlüsselungseinrichtungen beinhaltet. Ebenfalls enthält es Spezifikationen über Datenintegrität, Abhörsicherheit und Benutzer Authentifizierung.

Transport Layer (Transportschicht)
Das Wirelss Datagram Protocol (WDP) repräsentiert die Transportschicht und bildet die Schnittstelle zum physikalischen Netzwerk. Sie kann an die Vorgaben des Netzwerkanbieters angepasst werden, was das WAP damit völlig unabhängig von der Art der Netzwerkübertragung macht.

1.5 Datenübertragung (WAP-Request)

WAP-Inhalte und Anwendungen werden in webbasierenden Formaten auf einem Web-Server abgelegt. Das Anfragen einer solchen, speziell für Handys kreierten Page, nennt man WAP-Request. Zwischen dem Endgerät und dem Web-Server steht ein Gateway, das die Datenübertragung überhaupt ermöglicht.

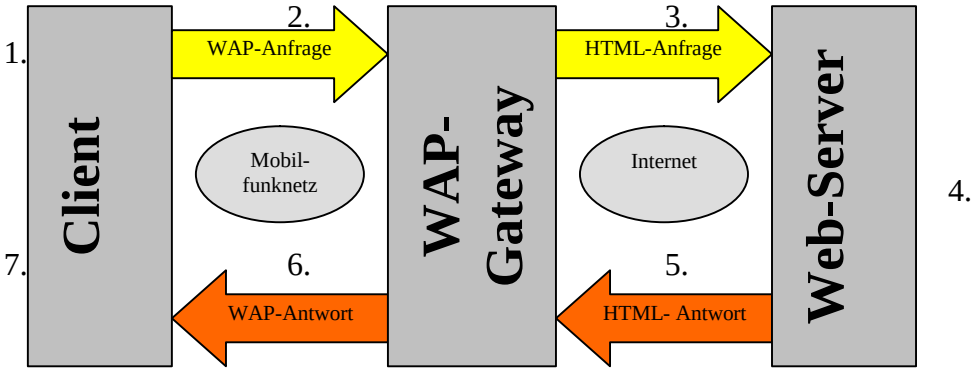


Bild2: Der WAP-Request

Bild 2 zeigt den typischen Ablauf eines WAP-Requests.

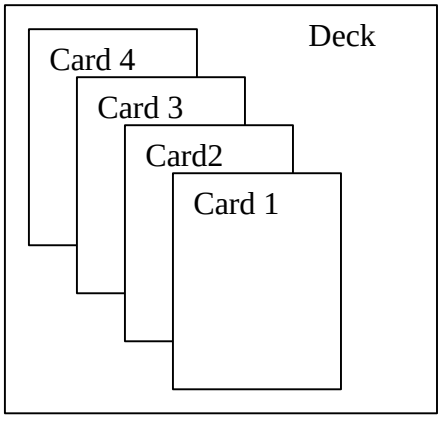
1. Eingabe einer URL, die zum Beispiel als Link im Browser hinterlegt sein kann, erzeugt innerhalb des Client einen Request.
2. Der Request wird mit Hilfe des WAP-Protokolls an ein WAP-Gateway übertragen.
3. Das WAP-Gateway wiederum transformiert den Request in einen herkömmlichen HTTP-Request und leitet diesen an den entsprechenden Web-Server weiter.
4. Der Web-Server bearbeitet den HTTP-Request wie gewohnt und gibt ein so genanntes WML-Deck mit zusätzlichem HTTP-Header an das WAP-Gateway zurück.
5. Innerhalb des WAP-Gateways erfolgt die Verifizierung des WML-Decks sowie des HTTP-Header und anschliessend die Kodierung in das binäre WML-Format.
6. Abschliessend erzeugt das WAP-Gateway eine WAP-Response und sendet diese an den Client.
7. Der Client empfängt die WAP-Response, arbeitet das binäre WML ab und stellt die erste Card des WML-Decks dar.

2. WML

So wie der Internetzugang HTTP mit der Programmiersprache HTML arbeitet, so steht hinter dem WAP die Programmiersprache WML (Wireless Markup Language), die aufgrund der Einschränkungen der mobilen Endgeräten entwickelt wurde und die es ermöglicht, Informationen aus dem Internet zum Mobiltelefon zu transportieren. WML kann als Zwischenstufe von HTML und XML angesehen werden. WML wurde speziell für die mobile Datenkommunikation und für kleine Endgeräte entwickelt. Das bisher einzige Grafikformat für WAP ist das WBMP (Wireless Bitmap).

2.1 Der WML-Aufbau

Anders als bei den HTTP-Seiten, wo jede einzelne Seite in ihrer eigenen Datei gespeichert ist, werden bei WML mehrere Seiten in einer Datei gespeichert, dem WML-Deck. Hier wird also die gesamte Seite als Deck bezeichnet.
Ein solches Deck besteht aus einer oder mehreren Cards (eine Card entspricht einer Seite).



Es wird immer ein komplettes Deck auf das Endgerät geladen, das logisch zusammenhängende Cards enthalten sollte. Durch geschicktes Navigieren ist ein Wechseln zwischen den verschiedenen Cards möglich.

2.2 Syntax

Da WML nah mit XML verwandt ist, besteht jedes Element aus einem öffnenden und einem abschliessenden Tag. Diese werden wie in der HTML-Version in eckigen Klammern geschrieben. Im Gegensatz zu HTML darf in WML aber der Endtag nicht vergessen werden. WML ist eine case-sensitive Sprache, das heisst, es muss besonders auf die Gross- und Kleinschreibung geachtet werden.
Die WML-Konstrukte lassen sich grob in die folgenden Bereiche unterteilen:

- **Elemente** sind alle Markups und strukturelle Informationen innerhalb eines WML-Dokumentes, umschlossen von Start- und Endtag.
- **Attribute:** Ein Grossteil der WML-Elemente kann durch Attribute genauer spezifiziert werden. Beispielsweise lässt sich mit Hilfe solcher Attribute einem Card-Tag ein eindeutigen Bezeichner zuweisen.
  <card id =“cardOne“ title=Learning WML“>
- **Comments:** Kommentare folgen der XML-Sprache und dienen dem WML-Autor als Information
  ***<!-** this is a comment ->*
- **Variable** dienen dem Austausch von Statusinformationen und Parametern zwischen verschiedenen Cards und Decks.
  $(variable1)

```
1.  <?xml version=“1.o“?>
2.  <!Doctype wml PUBLIC”-//WAP-Forum//DTD WML 1.1//EN”
    http://www.wapforum.org/DTD/wml_1.1.xml>

3.  <wml>
4.      <card id=”Card1” tittle=”WAP Vortrag”>
5.      <p>
6.          Ich lerne WML-Programmierung
7.      </p>
8.      </card>
9.  </wml>
```

Erläuterungen:
1.: Standardzeile für alle XML-Dokumente
2.: Das ist die Dokumenttypdefinition
3.: Das WML-Deck wird durch den WML-Tag definiert
4.: Jede WML-Card wird durch einen Card-Tag definiert
5.-7.: Einzublendender Text umrahmt von Paragraph-Tags
8.: Die WML-Card wird durch ein Card-Tag beendet
9.: Das WML-Deck wird durch ein WML-Tag beendet

3. Zukunftsaussicht

Das WAP hatte von Anfang an schon mit diversen Problemen zu kämpfen und hatte bis jetzt auch noch nicht den grossen Durchbruch erreicht. Das SMS ist nach wie vor beliebter und kann fast genauso viel wie das WAP. Auch an WAP-Kritikern mangelt es nicht. Vielen ist das WAP zu teuer, zu langsam und zu umständlich.
Dennoch stellt das WAP mit Sicherheit eine bedeutende Technologie dar. Auch sind auf Seiten der mobilen Infrastruktur mit dem GPRS (General Packet Radio Service) und dem UMTS (Universal Mobile Telecommunication System) richtungsweisende Wege eingeschlagen worden. Diese neuen Technologien ermöglichen Dank einer weit höheren Übertragungsrate ein wesentlich schnelleres Netz und öffnen somit eine neue Dimension für die mobile Kommunikation.
Auch wird das WAP ständig weiterentwickelt und verbessert und so kommt es nie zu einem Stillstand. Mit der Veröffentlichung des WAP2.0 2001 wurde Grundlegendes richtungsweisend verändert und forderte einiges Umdenken.
Fakt ist: Die WAP-Technologie wird vor allem in Zukunft noch wichtiger werden und der entgültige Durchbruch ist nur eine Frage der Zeit

4. Literaturverzeichnis

a. Andreas Hitzig: Mobilkommunikation, Drahtlos surfen im Internet, 2000)

b. Lars Röwekamp: WML-Programmierung, Handy HTML, 2000)

c. WAP-Forum: WAP Architecture, Version 12 Juli 2001, WAP-Forum, 2001)

d. WAP-Forum: WAP2.0 Technical White Paper, WAP-Forum 2001)

e. WAP-Forum: Wireless Application Protocol White Paper, WAP-Forum, 2000)

f. www.wapforum.org

g. www.ccwap.de