INSTITUT FÜR INFORMATIK

DER LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Praktikum IT-Sicherheit

Prof. Dr. H.–G. Hegering
Dr. H. Reiser
B. Oberhaitzinger, H. Gerloni

Wintersemester 2003/2004

IT-Sicherheit

Inhaltsverzeichnis

Literatur

[VVa 02] Vadim Kurland and Vadim Zaliva. *Firewall Builder.* `http://www.fwbuilder.org/`, 2002.

[7105 92] BSI 7105. *IT-Sicherhaitshandbuch.* Bundesanzeiger-Verlag, Köln, 1992.

[abu 02] `http://spam.abuse.net/`, 2002.

[AlLi 98] Paul Albitz and Cricket Liu. *DNS and BIND.* O'Reilly & Associates, 3nd edition, 1998.

[Amor 99] Edward Amoroso. *Intrusion Detection.* Intrusion.Net Books, 1st edition, 1999.

[Andr 02] Oskar Andreasson. *Iptables Tutorial.* `http://linux-sxs.org/iptables/`, 2002.

[ANKu 02] Glenn Brunette Alex Noordergraaf and Dina Kurktchi. *Solaris[tm] Security Toolkit ("jass").* Sun Microsystems, Inc., `http://www.sun.com/blueprints/tools/jass/jass.html`, 2002.

[apa 02] `http://www.apache.org/`, 2002.

[aps 02] `http://www.apache-ssl.org/`, 2002.

[Atki 95a] R. Atkinson. RFC 1825: Security Architecture for the Internet Protocol. RFC, IETF, August 1995.

[Atki 95b] R. Atkinson. RFC 1827: IP Encapsulating Security Payload (ESP). RFC, IETF, August 1995.

[Aust 01] Tom Austin. *PKI - A Wiley Tech Brief.* Wiley Computer Publishing, John Wiley & Sons, Inc., 2001.

[BaEd 95] F. Baker and Ed. RFC 1812: Requirements for IP Version 4 Routers. RFC, IETF, June 1995.

[Bart 01] Wolfgang Barth. *Das Firewall Buch.* SuSE-Press, Nürnberg, 2001.

[bin 02] `ftp://ftp.isc.org/isc/bind/src/cur/bind-8/`, 2002.

[Blac 91] U.D. Black. *OSI: A model for computer communications standards.* Prentice-Hall, 1991.

[boe 98] `http://www.lrz-muenchen.de/services/security/ssh/`, 1998.

[boe 02] `http://www.lrz-muenchen.de/services/security/links/`, 2002.

[Boro 93] Petra Borowka. *Brücken und Router - Wege zum strukturierten Netzwerk.* DATACOM, 1993.

[Boss 99] Antoon Bosselaers. *The hash function RIPEMD-160*. Katholieke Universiteit Leuven, `http://www.esat.kuleuven.ac.be/~bosselae/ripemd160.html`, 1999.

[Cerf 81] V.G. Cerf. RFC 794: Pre-emption. RFC, IETF, September 1981.

[Cost 97] Eric Costales, Bryan with Allman. *Sendmail*. O'Reilly & Associates, 2nd edition, 1997.

[CrGi 85] W.J. Croft and J. Gilmore. RFC 951: Bootstrap Protocol. RFC, IETF, September 1985.

[Croc 82] D. Crocker. RFC 822: Standard for the format of ARPA Internet text messages. RFC, IETF, August 1982.

[dan 02] `http://www.inet.no/dante/`, 2002.

[den 02] `http://www.denic.de/`, 2002.

[DiAl 99] T. Dierks and C. Allen. RFC 2246: The TLS Protocol Version 1.0. RFC, IETF, January 1999.

[Dobb 96] Hans Dobbertin. *Cryptanalysis of MD5 Compress*. Bundesamt für Sicherheit in der Informationstechnik (BSI), `http://www.informatik.uni-mannheim.de/informatik/pi4/projects/Crypto/rgp/md5`, 1996.

[Drom 93] R. Droms. RFC 1541: Dynamic Host Configuration Protocol. RFC, IETF, October 1993.

[EDZ 00] D. Brent Chapman Elizabeth D. Zwicky, Simon Cooper. *Building Internet Firewalls*. O'Reilly & Associates, 2nd edition, 2000.

[EgFr 94] K. Egevang and P. Francis. RFC 1631: The IP Network Address Translator (NAT). RFC, IETF, May 1994.

[fir 02] `http://www.first.org/`, 2002.

[FLYV 93] V. Fuller, T. Li, J. Yu, and K. Varadhan. RFC 1519: Classless Inter-Domain Routing (CIDR): an Address Assignment and Aggregation Strategy. RFC, IETF, September 1993.

[FMMT 84] R. Finlayson, T. Mann, J.C. Mogul, and M. Theimer. RFC 903: Reverse Address Resolution Protocol. RFC, IETF, June 1984.

[Fuhr 98] Kai Fuhrberg. *Internet - Sicherheit: Browser, Firewalls und Verschlüsselung*. Hanser Verlag, 1998.

[Geno 02] Luigi Genoni. *Knetfilter*. `http://expansa.sns.it/knetfilter/`, 2002.

[gnu 02] `http://www.gnupg.org/`, 2002.

[HAN 99] H.-G. Hegering, S. Abeck, and B. Neumair. *Integriertes Management vernetzter Systeme – Konzepte, Architekturen und deren betrieblicher Einsatz*. dpunkt-Verlag, 1999.

[Hedr 88] C.L. Hedrick. RFC 1058: Routing Information Protocol. RFC, IETF, June 1988.

[HeLa92] H.-G. Hegering and A Läpple. *Ethernet - Basis für Kommunikationsstrukturen*. DATACOM, 1992.

[Horn 84] C. Hornig. RFC 894: Standard for the transmission of IP datagrams over Ethernet networks. RFC, IETF, April 1984.

[Hunt 98] Craig Hunt. *TCP/IP Network Administration*. O'Reilly & Associates, 2nd edition, 1998.

[ica 02] `http://www.icann.org/`, 2002.

[Inc. 02a] RSA Security Inc. *Behind the Patent - Frequently Asked Questions*. RSA Security Inc., `http://www.rsasecurity.com/solutions/developers/total-solution/faq.html`, 2002.

[Inc. 02b] RSA Security Inc. *What is Diffie-Hellman?* RSA Security Inc., `http://www.rsasecurity.com/rsalabs/faq/3-6-1.html`, 2002.

[ISO/ 88] ISO/IEC. Information Processing Systems - Open Systems Interconnection Reference Model - Security Architecture, 1988.

[John 85] M. St. Johns. RFC 931: Authentication server. RFC, IETF, January 1985.

[John 93] M. St. Johns. RFC 1413: Identification Protocol. RFC, IETF, February 1993.

[Junn 02] Tomas Junnonen. *Firestarter*. `http://firestarter.sourceforge.net/`, 2002.

[KeAt 98] S. Kent and R. Atkinson. RFC 2402: IP Authentication Header. RFC, IETF, November 1998.

[KlEd 01] J. Klensin and Ed. RFC 2821: Simple Mail Transfer Protocol. RFC, IETF, April 2001.

[Kraw 95] Hugo Krawczyk. *SKEME: A Versatile Secure Key Exchange Mechanism for Internet*. Institute of Electrical and Electronics Engeneers, `IEEE,http://www.research.ibm.com/security/skeme.ps`, 1995.

[Lang 99] Nicolai Langfeldt. *DNS Howto*. `ftp://sunsite.unc.edu/pub/Linux/docs/HOWTO/`, 1999.

[Leec 96] M. Leech. RFC 1929: Username/Password Authentication for SOCKS V5. RFC, IETF, March 1996.

[LGL+ 96] M. Leech, M. Ganis, Y. Lee, R. Kuris, D. Koblas, and L. Jones. RFC 1928: SOCKS Protocol Version 5. RFC, IETF, March 1996.

[lin 01] `http://www.pathname.com/fhs/`, 2001.

[LoRe 91] K. Lougheed and Y. Rekhter. RFC 1267: Border Gateway Protocol 3 (BGP-3). RFC, IETF, October 1991.

[mac 02a] `http://www.gnu.org/manual/m4/index.html`, 2002.

[mac 02b] `http://www.sendmail.org/m4/readme.html`, 2002.

[Mart 01] Derek D. Martin. `http://rr.sans.org/DNS/sec_BIND.php`, 2001.

[McMa 96] P. McMahon. RFC 1961: GSS-API Authentication Method for SOCKS Version 5. RFC, IETF, June 1996.

[Moy 91] J. Moy. RFC 1247: OSPF Version 2. RFC, IETF, July 1991.

[net 00] `http://wp.netscape.com/security/techbriefs/`, 2000.

[Niko 02] Bastian Schmick Niko Schweitzer, Michael Schmidt. *Security Server*. Institut für Nachrichtenübermittlung der Universität Siegen, `http://www.infoserversecurity.org/`, 2002.

[Nort 99] Stephen Northcutt. *Network Intrusion Detection*. New riders Publishing, 1999.

[Patt 98] Michael A. Patton. *Ethernet Codes master page*. `http://www.cavebear.com/CaveBear/Ethernet/`, 1998.

[pgp 02] `http://www.pgpi.org/`, 2002.

[Plat 02] Prof. J. Plate. `http://www.netzmafia.de/skripten/`, 2002.

[Plum 82] D.C. Plummer. RFC 826: Ethernet Address Resolution Protocol: Or converting network protocol addresses to 48.bit Ethernet address for transmission on Ethernet hardware. RFC, IETF, November 1982.

[PoRe 85] J. Postel and J.K. Reynolds. RFC 959: File Transfer Protocol. RFC, IETF, October 1985.

[PoRe 88] J. Postel and J.K. Reynolds. RFC 1042: Standard for the transmission of IP datagrams over IEEE 802 networks. RFC, IETF, February 1988.

[Post 80] J. Postel. RFC 768: User Datagram Protocol. RFC, IETF, August 1980.

[Post 81a] J. Postel. RFC 791: Internet Protocol. RFC, IETF, September 1981.

[Post 81b] J. Postel. RFC 793: Transmission Control Protocol. RFC, IETF, September 1981.

[Post 82] J. Postel. RFC 821: Simple Mail Transfer Protocol. RFC, IETF, August 1982.

[pro 01] `http://proftpd.linux.co.uk/localsite/Userguide/linked/userguide.html`, 2001.

[pro 03] `http://www.proftpd.org/`, 2003.

[Raep 98] Martin Raepple. *Sicherheitskonzepte für das Internet*. dpunkt-Verlag, 1998.

[Raym 01] Eric S. Raymond. *Jargon File Resources*. `http://www.tuxedo.org/~esr/jargon/`, 2001.

[rea 02] `http://www.iss.net/`, 2002.

[ReEd 01] P. Resnick and Ed. RFC 2822: Internet Message Format. RFC, IETF, April 2001.

[Resc 99] E. Rescorla. RFC 2631: Diffie-Hellman Key Agreement Method. RFC, IETF, June 1999.

[rfc] *Request for Comments*. IETF, `ftp://ftp.isi.edu/in-notes/`.

[Rive 92] R. Rivest. RFC 1321: The MD5 Message-Digest Algorithm. RFC, IETF, April 1992.

[RMK$^+$ 96] Y. Rekhter, B. Moskowitz, D. Karrenberg, G. J. de Groot, and E. Lear. RFC 1918: Address Allocation for Private Internets. RFC, IETF, February 1996.

[RRW 01] Richard Forno Richard R.van Wyk. *Incident Response*. O'Reilly & Associates, 1st edition, 2001.

[Scha 02] Boris Schauerte. Feind im Dunkeln. *Linux Magazin*, 03(02):44–47, März 2002.

[Schn 96] Bruce Schneier. *Angewandte Kryptographie*. Addison-Wesley, 1996.

[Seli 02] Marc André Selig. Private Feuerwände. *Linux User*, 05(02):30–35, Mai 2002.

[sen 02] `http://www.sendmail.org/`, 2002.

[sha 01] `http://www.nswc.navy.mil/ISSEC/CID/`, 2001.

[sno 02] `http://www.snort.org/`, 2002.

[soc 02] http://www.socks.nec.com/, 2002.

[sou 02] http://sourceforge.net/projects/tripwire/, 2002.

[spa 02] http://www.spam.org/, 2002.

[squ 02a] http://www.squid-cache.org/, 2002.

[squ 02b] http://squid-docs.sourceforge.net/, 2002.

[Squi 01] Alicia Squires. http://rr.sans.org/DNS/BIND.php, 2001.

[Stev 96] W. Richard Stevens. *TCP/IP Illustrated, Volume 1, The Protocols*. Addison-Wesley, 1996.

[swo 02] http://www.swobspace.de/linux/das-firewall-buch/material-das-firewall-buch.h 2002.

[tha 00] http://www.thawte.com/, 2000.

[Thom 02] Rob Thomas. http://www.cymru.com/Documents/secure-bind-template.html, 2002.

[tri 02a] http://www.tripwire.org/, 2002.

[tri 02b] http://www.tripwire.com/, 2002.

[uTW 00] Nicolai Langfeldt und Thomas Walter. http://www.fokus.gmd.de/linux/dlhp/aktuell/DI 2000.

[WeCl 97a] D. Wessels and K. Claffy. RFC 2186: Internet Cache Protocol (ICP), version 2. RFC, IETF, September 1997.

[WeCl 97b] D. Wessels and K. Claffy. RFC 2187: Application of Internet Cache Protocol (ICP), version 2. RFC, IETF, September 1997.

[Wess 01] Duane Wessels. *Web Caching*. O'Reilly & Associates, 1nd edition, 2001.

[Wint 96] J. De Winter. RFC 1985: SMTP Service Extension for Remote Message Queue Starting. RFC, IETF, August 1996.

[wuf 02] http://www.wuftpd.org/, 2002.

[www 02] http://wwwhomes.uni-bielefeld.de/schoppa/saia-howto.html, 2002.

Einführung ins Praktikum

0.1 Termine und Gruppenverteilung

Im Wintersemester 2003/2004 findet die Theorie zum Praktikum IT-Sicherheit voraussichtlich an folgenden Terminen (Dienstag) statt:

0)	**21. Oktober 2003**	6)	**02. Dezember 2003**
1)	**28. Oktober 2003**	7)	**09. Dezember 2003**
2)	**04. November 2003**	8)	**16. Dezember 2003**
3)	**11. November 2003**	9)	**13. Januar 2004**
4)	**18. November 2003**	10)	**20. Januar 2004**
5)	**25. November 2003**	11)	**27. Januar 2004**

Veranstaltungsort ist die Oettingenstr. 67. Die Veranstaltungszeiten sind:

- **8:15 - 10:00h**: Theorie (Raum 1.14)
- **10:00 - 12:00h**: Praxis und Abnahme (Räume D.5, D.7, D.9)

Zusätzlich bekommen Sie pro Woche an vier Tagen betreuten Zutritt zu den Praktikumsräumen.

Für den praktischen Teil der Veranstaltung werden zwei Gruppen zu je 20 Personen gebildet. Diese beiden Gruppen bekommen abwechselnd Zugang zu den zehn Praktikumsrechnern. Zwei Leute arbeiten an einem Rechner, für den sie für das gesamte Praktikum verantwortlich sind. Je zwei dieser Zweiergruppen bilden eine Praktikumseinheit, die zusammenarbeitet und auf einander angewiesen ist.
Die Gruppeneinteilung sowie die Zuordnung zu den Rechnern wird während der Einführungsveranstaltung festgelegt und kann nachträglich nicht mehr geändert werden.

Die genauen Termine, die Gruppenverteilung sowie weitere Informationen zum Praktikum finden Sie unter

- `http://wwwhegering.informatik.tu-muenchen.de/Praktika/ws0304/secp/`

Für weitere Fragen zur Organisation und nach Praktikumsstart auch zum Inhalt sind wir über E-Mail erreichbar:

- **secp@nm.informatik.uni-muenchen.de**

Über die Mailingliste **secpteilnehmer@nm.informatik.uni-muenchen.de** sind alle Teilnehmer des Praktikums erreichbar.

Kritik, Anregungen und Verbesserungsvorschläge sind jederzeit willkommen.

0.2 Scheinvergabe

Für die Vergabe eines Scheins gelten folgende Voraussetzungen:

- **Regelmäßige Teilnahme**
  Die Termine sollten alle eingehalten werden. An einem Termin ist jedoch ein begründetes, entschuldigtes Fernbleiben erlaubt. Die Übungen sind jedoch in jedem Fall bis zum folgenden Termin nachzuholen.
- **Erfolgreiche Bearbeitung der praktischen Aufgaben**
  Die Aufgaben sind bis zum nachfolgenden Termin vollständig abzuarbeiten. Die im Praxisteil gestellten Fragen sind zudem schriftlich zu beantworten und eine kurze Beschreibung der Konfiguration mit Auszügen aus den wichtigen Konfigurationsdateien zu erstellen. Die Ausarbeitung ist am nachfolgenden Termin vorzulegen, die Form ist frei (auf Papier, elektronisch). Der Lernerfolg wird außerdem im Laufe der Praxistermine in mündlichen Dialogen überprüft.
- **Abschlußprüfung**
  Sollten wir mit Ihren Leistungen während des Praktikums nicht zufrieden sein, bekommen Sie in einer mündlichen Abschlußprüfung nochmal die Möglichkeit, uns von Ihrem Wissen zu überzeugen. Der Termin für diese Prüfung ist der **15. Juli 2003**.
- **Schein-Gültigkeit**
  Das Praktikum richtet sich an Studenten nach dem Vordiplom und gilt als Wahlpflichtpraktikum mit sechs Semesterwochenstunden für:
  - **LMU:** Bereich Systemnahe und Technische Informatik (ST) und/oder A
  - **TUM:** Bereich Informatik II - Technische Informatik

0.3 Die Rechner

Auf den Rechnern ist SuSE-Linux Version 8.0 installiert. Einige Werkzeuge (insbesondere YaST) werden Sie auf anderen Linux- oder Unix-Systemen nicht vorfinden, nach einiger Erfahrung im Unix-Umfeld eventuell auch nicht mehr vermissen. Wir haben uns für den Einsatz von SuSE-Linux entschlossen, da es relativ komfortabel ist und Sie wohl am ehesten mit dieser Linux-Distribution vertraut sind.
Die Rechner sind nicht als reine Firewalls installiert, da Sie sie ja auch als Arbeitsstation benutzen müssen. Natürlich haben im Normalfall ein Office-Paket oder eine grafische Oberfläche auf einem Firewall nichts verloren.
Sie arbeiten auf den Rechnern als User `root`, haben also volle Zugriffsrechte aufs Betriebssystem.

0.4 Boot- und Sicherungskonzept

Auf den Rechnern sind mehrere Installationen vorhanden. Die Dateisysteme sind verschlüsselt und können nur nach Eingabe eines Passwortes gelesen werden. Beim Hochfahren des Rechners wählen Sie im Boot-Manager Ihre Partition aus und geben Ihr Passwort ein. Zur Sicherung der Systeme wurde ein eigener Mechanismus implementiert.
Eine genaue Beschreibung des Konzeptes finden Sie im Anhang A.1
Bei Fragen diesbezüglich wenden Sie sich bitte an die Betreuer.

0.5 Rechtliches zum Hacking

Die im Rahmen dieses Praktikums gezeigten Methoden zum Ausspähen und Verändern von Daten dürfen nur innerhalb unserer Versuchsumgebung angewandt werden. Sie dienen nur der Verdeutlichung der Gefahren, welche vom Anschluß eines Rechners an ein Netzwerk ausgehen können und sind nicht als Wissensvermittlung zum Einbruch in andere Systeme zu verstehen.
Einbruchsversuche außerhalb des Praktikumsnetzes führen zum sofortigen Ausschluß vom Praktikum. Im schlimmsten Falle droht ein Verweis von der Universität und strafrechtliche Konsequenzen.

Einige Paragraphen zum Thema Computersicherheit und Datenschutz finden Sie im Anhang A.3.

0.6 Vorgehensweise bei Konfigurationen und Informationsbeschaffung

0.6.1 Konfiguration

Es empfiehlt sich, von jeder zu verändernden Datei eine Sicherungskopie des Orginals zu erstellen.
Bei den praktischen Aufgaben werden Sie eine Reihe von Konfigurationsschritten vorfinden, die am Ende der Aufgabe zusammen spielen und funktionieren sollen. Aus der Erfahrung heraus ist es aber nicht zu empfehlen, alle Schritte auf einmal zu konfigurieren und dann erst zu testen. In diesem Fall ist die Fehlersuche sehr aufwendig und oft auch nicht von Erfolg gekrönt. Es sollte also schrittweise vorgegangen werden. D.h. zuerst ist bei einer Aufgabenstellung das Grobgerüst zu konfigurieren. Erst wenn alle Tests hierzu erfogreich waren, ist schrittweise mit der Verfeinerung der Konfiguration fortzufahren, wobei nach jeder Erweiterung wieder Tests durchzuführen sind.
Ein Test, der bei allen TCP-Diensten funktioniert ist:
`telnet <IP-Adresse> <TCP-Port>`
Betrachten wir diesen Test einmal für HTTP:
Ist ein Webserver unter Port 80 gestartet, so sieht die Kommunikation so aus:

```
telnet 10.50.211.120 80
Trying 10.50.211.120...
Connected to 10.50.211.120.
Escape character is '^]'.



^]
telnet> q
Connection closed.
```

Diese Verbindung kann mit dem `^]` verlassen werden. `^]` erzeugt man unter Linux mit der Tastenkombination `Strg 5` und unter Windows mit `Strg +`. Läuft unter Port 80 kein Dienst, so bekommt man als Antwort ein `Connection refused`:

```
telnet 10.50.211.111 80
Trying 10.50.211.111...
telnet: Unable to connect to remote host: Connection refused
```

Wichtige Hinweisquellen bei der Fehlersuche sind die Logfiles des Systems. Standardsystemlogfile z.B. bei SuSE ist `/var/log/messages`, bei Debian `/var/log/syslog` und bei Solaris `/var/adm/messages`. Oft haben Dienste spezielle Logfiles, die aber dann bei den jeweiligen Kapiteln explizit erwähnt sind.

0.6.2 Informationsbeschaffung

Unter Unix gibt es als Informationsquelle die sogenannten **Man Pages**. Möchte man Informationen zu einem Befehl, so gibt man `man befehlsname` ein und erhält die in den Man Pages abgelegten Informationen. Am Ende einer Man Page stehen oft Verweise auf weitere, zu diesem Befehl gehörende oder verwandte Befehle oder Dateien.
Zusätzlich sollte man sich das Internet bei der Informationssuche zu Nutzen machen. Zu allen in diesem Praktikum angeschnittenen Bereichen gibt es weiterführende und ergänzende Informationen im Netz. Z.B. kann man über `www.google.de` mit ein paar Schlagworten sehr leicht die gewünschten Informationen finden.
Alle Protokolle und Vereinbarungen über das Internet sind in Dokumenten festgehalten. Ganz zu Anfang waren diese Dokumente als Diskussionsgrundlagen für einen weiten Kreis von Teilnehmern gedacht, weshalb sie den Namen **Request for Comment**, kurz **RFC**, erhielten. Es stellte sich jedoch heraus, daß der Kreis der Diskutierenden relativ klein war und daher auf das RFC-Dokument kein eigenes Standard-Dokument folgen mußte. Die Dokumente sind durchnummeriert und im Netz frei erhältlich (z. B. auf `ftp://ftp.leo.org/`), können aber auch auf CD-ROM erworben werden. Ein RFC wird nicht ungültig, sondern bei Änderungen im Protokoll durch ein Nachfolgedokument (**Son of RFC xxx**) ergänzt oder ersetzt. Die RFCs enthalten nicht nur technische Dokumente, sondern auch für den

technischen Laien geeignete Bedienungsanleitungen. Wer sich mit den RFCs zu gebräuchlichen Diensten beschäftigt, wird feststellen, daß viele Protokolle sehr einfach gehalten sind und daher die Implementierung auf verschiedenen Rechnern schnell möglich ist. Die RFCs werden vom **InterNIC** verwaltet. RFCs finden Sie z.B. unter `http://www.faqs.org/rcf/`.

0.6.3 Der Editor `vi`

Zum Editieren der Konfigurationsfiles gibt es eine Vielzahl von Editoren, die man verwenden kann. Man sollte sich aber darüber im Klaren sein, dass nicht jeder Editor auf jedem System zur Verfügung steht. Ein Standardserver hat in der Regel keinen X Server, da für einen Server keine grafische Benutzeroberfläche nötig ist. Somit fallen alle Editoren, die nur unter grafischen Oberflächen laufen, weg. Der Editor, der auch bei einem Minimalsystem mitgeliefert und installiert ist, ist der **vi**. Somit sollte man sich schon von Beginn an mit diesem Tool auseinaner setzen, das es unter Unix immer zur Verfügung steht.

Aufruf

Der Aufruf des `vi` erfolgt mit

```
vi [Option...] [Datei...]
```

wobei z.B. folgende Optionen möglich sind:

- `-wn`: Fenstergröße wird auf `n` Zeichen gesetzt.
- `-R`: nur lesender Zugriff.
- `-t marke`: Datei, die `marke` enthält, wird editiert und der Kurser auf `marke` positioniert.
- `+ datei`: Bearbeiten der Datei `datei` beginnt am Ende der Datei.
- `+n datei`: Bearbeiten der Datei `datei` beginnt ab Zeile `n` der Datei.

vi-Bearbeitung

- **beenden**:
  - `:wq`: speichern und beenden
  - `:q!`: beenden ohne speichern
  - `:x`: speichern und beenden
- **speichern**

  - `:w`: speichern der aktuellen Datei
  - `:w datei`: speichern der aktuellen Datei in `datei`
  - `:w! datei`: speichern der aktuellen Datei in `datei`, die überschrieben wird

- **Kursor-Bewegungen**
  - nach rechts: `l`, Leertaste, $\rightarrow$
  - nach links: `h`, $\leftarrow$
  - nach oben: `k`, $\uparrow$
  - nach unten: `j`, $\downarrow$
  - Anfang nächste Zeile: `+`
  - Anfang vorherige Zeile: `-`
  - Wort nach rechts: `w`
  - Wort nach links: `b`
  - ans Ende des nächsten Wortes
  - an den Zeilenanfang: `0`
  - an das Zeilenende: `$`
  - an das Dateiende: `G`
  - zur Zeile `n` springen: `nG`
  - Bildschirmseite vorwärts: `Strg F`
  - Bildschirmseite rückwärts: `Strg B`
  - halbe Bildschirmseite vorwärts: `Strg D`
  - halbe Bildschirmseite rückwärts: `Strg U`
  - erste Bildschirmzeile: `H`
  - letzte Bildschirmzeile: `L`
  - mittlere Bildschirmzeile: `M`

- **Eingabemodus**
  - `i`: Text vor dem Kursor schreiben
  - `a`: Text nach dem Kursor schreiben
  - `A`: Text am Ende der Zeile schreiben
  - `o`: neue Zeile unter dem Kursor einfügen
  - `O`: neue Zeile über dem Kursor einfügen
  - `R`: Text überschreiben

  - `cw`: Wort ab Kursorposition überschreiben
  - `C`: Zeile ab Kursorposition überschreiben
  - `Esc`: Eingabemodus verlassen

- **löschen**
  - `x`: Zeichen löschen
  - `dw`: Wort löschen
  - `dd`: Zeile löschen

- **Textbearbeitung**
  - `rc`: Zeichen an Kursorposition mit Zeichen `c` überschreiben
  - `u`: letzte Änderung rückgängig machen
  - `U`: Zeile wiederherstellen

- **Textsuchen**
  - `/text`: suche Stelle `text` vorwärts
  - `?text`: suche Stelle `text` rückwärts
  - `n`: Suche fortsetzen
  - `N`: Suche in anderer Richtung fortsetzen

- **Text in Puffer speichern**
  - `yy`: Kursorzeile speichern
  - `Y`: Kursorzeile speichern
  - `nY`: `n` Zeilen ab Kursorzeile speichern
  - `nyy`: `n` Zeilen ab Kursorzeile speichern
  - `yw`: Wortrest ab Kursorposition speichern
  - `nyw`: `n` Worte ab Kursorposition speichern
  - `y$`: Wortrest ab Kursorposition speichern

- **Text aus Puffer einsetzen**
  - `pp`: Pufferinhalt hinter Kursor einsetzen
  - `P`: Pufferinhalt vor Kursor einsetzen

- **suchen, ersetzen**
  - `:s/text1/text2/`: in der aktuellen Zeile wird das erste `text1` durch `text2` ersetzt

  - `:s/text1/text2/g`: in der aktuellen Zeile wird jedes `text1` durch `text2` ersetzt
  - `:s/text1/text2/gc`: in der aktuellen Zeile wird jedes `text1` durch `text2` ersetzt, wobei aber eine Abfrage vorgeschalten ist
  - `:x,ys/text1/text2/g`: in Zeile `x` bis `y` wird das jedes `text1` durch `text2` ersetzt
  - `:%s/text1/text2/g`: in jeder Zeile der Datei wird jedes `text1` durch `text2` ersetzt

- **andere Dateien einlesen**
  - `:r datei`: Datei `datei` wird an Kursorposition eingefügt

1 Grundlagen von TCP/IP Netzwerken

Aufgrund der unterschiedlichen Architekturen von verschiedenen Herstellern wurden in der Vergangenheit mehrere Netzwerkprotokolle für die Kommunikation in Rechnernetzen entwickelt. Als De-Facto-Standard hat sich aber mittlerweile die Internet-Protokoll-Familie durchgesetzt. Im LAN (Local Area Network, Netzwerk mit kurzen Entfernungen und hohen Übertragungsgeschwindigkeiten) ist auf der Netzanschlußebene zudem fast ausschließlich Ethernet im Einsatz. Die für dieses Praktikum relevanten Eigenschaften der beiden Architekturen werden im Folgenden in Verbindung mit dem OSI-Schichtenmodell dargestellt.

1.1 OSI- und Internet-Schichtenmodell

Das OSI (Open Systems Interconnection) Basisreferenzmodell [HAN 99] der ISO (International Standards Organisation) besteht aus 7 Schichten (layers). Jede dieser Schichten repräsentiert nicht etwa ein Protokoll, sondern vielmehr eine Funktion, die beim Austausch von Daten zwischen Anwendungen über ein dazwischen liegendes Netzwerk hinweg von beliebig vielen Protokollen ausgeführt wird.
Jedes Protokoll kommuniziert logisch nur mit dem entsprechenden Protokoll seines Kommunikationspartners (Peer). Damit ein erfolgreicher Datenaustausch stattfinden kann, muß die Kommunikation zwischen den Peers über die Protokoll-Definitionen standardisiert sein.

Im Folgenden werden die 7 Schichten des OSI-Modells kurz beschrieben. Es folgt ein Vergleich des OSI-Modells mit dem Internet-Modell (vgl. Abbildung 1).

- **Anwendungsschicht (Application Layer)**: beinhaltet die Anwendungen zur Nutzung oder Bereitstellung von Netzwerkdiensten
- **Darstellungsschicht (Presentation Layer)**: standardisiert das Format der Daten auf dem Netz. Auf dieser Schicht werden Codierung und Datentransfersyntax definiert.
- **Kommunikationssteuerungsschicht (Session Layer)**: Einrichtung, Strukturierung und Verwaltung der logischen Verbindungen (Sessions) zwischen den Anwendungen. Des weiteren werden Dienste für die Steuerung und Strukturierung von Sitzungen, Rechtevergabe und Synchronisation bereitgestellt.
- **Transportschicht (Transport Layer)**: garantiert eine Ende-zu-Ende-Verbindung zwischen zwei Systemen mit Fehlererkennung und -korrektur. Zu den Aufgaben gehören auch die Bereitstellung von Güteparametern (Durchsatz, Verzögerung, Verfügbarkeit, Restfehlerrate), das Multiplexen von Verbindungen sowie die Verkehrsflußsteuerung.
- **Vermittlungsschicht (Network Layer)**: verwaltet die logischen Verbindungen zwischen den Rechnern im Netz. Insbesondere werden hier auch Funktionen zur Wegewahl und Vermittlung bereitgestellt.

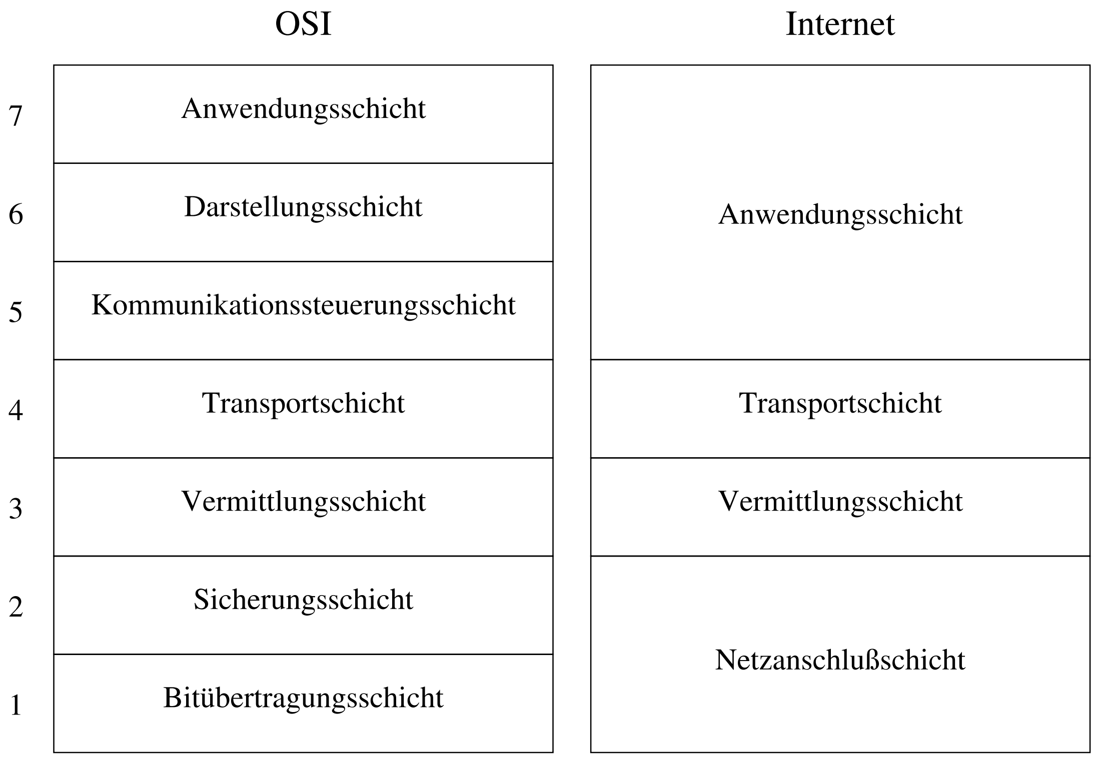

Abbildung 1: OSI- und Internet-Schichtenmodell

- **Sicherungsschicht (Data Link Layer)**: sorgt für eine zuverlässige Übertragung der Daten über die physikalischen Verbindungen. Die Schicht kann in zwei Unterschichten aufgeteilt werden. Die Medienzugangsschicht (MAC-Layer, Media Access Control Layer) regelt in von mehreren Stationen gemeinsam benutzten Übertragungsmedien die statische oder dynamische Zuteilung des Mediums an die einzelnen Stationen. Die LLC-Schicht (Logical Link Control Layer) ist zuständig für die Zusammenfassung von Bitsequenzen zu Blöcken (Frames), die Blocksynchronisation sowie die Fehlererkennung und eventuell Korrektur auf Blockebene.

- **Bitübertragungsschicht (Physical Layer)**: definiert die physikalischen Eigenschaften der Übertragungswege (Hardware, Signalpegel), die Übertragungsarten (z.B. analog/digital, synchron/asynchron) sowie Modulations- und Codierungsverfahren.

In der Praxis hat das OSI-Modell vor allem in Telekommunikationsnetzen Bedeutung erlangt, in Rechnernetzen hat sich die Internet-Architektur durchgesetzt.

Das Internet-Modell kennt weniger Schichten als das OSI-Modell, die Grundaufgaben der Schichten sind aber vergleichbar.

- **Anwendungsschicht**: Die Funktionen der OSI-Schichten 5 bis 7 werden im Internet-Modell in den Anwendungsprotokollen zusammengefaßt. Diese implementieren die

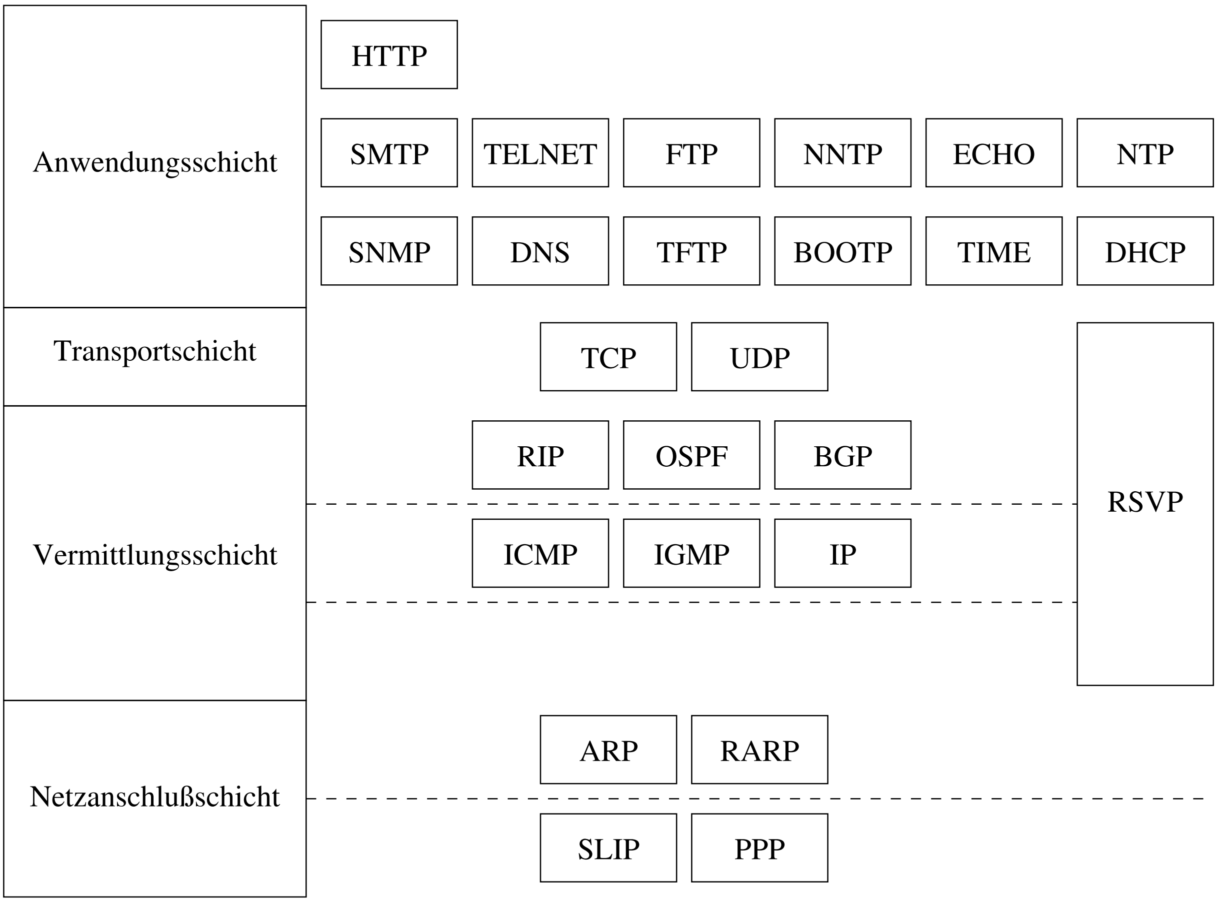

Abbildung 2: Auswahl von Internetprotokollen (aus [HAN 99, Seite 25])

Dienste unabhängig voneinander auf spezifische Weise. Beispiele für Protokolle auf dieser Schicht sind HTTP, FTP, SMTP, TELNET, DNS und NTP.

- **Transportschicht**: Die wichtigsten Protokolle dieser Schicht sind in der Internetwelt TCP (Transmission Control Protocol, RFC 793 [Post 81b]), ein gesichertes, verbindungsorientiertes Protokoll, und UDP (User Datagram Protocol, RFC 768 [Post 80]), ein schneller aber ungesicherter Datagramm-Dienst.

- **Vermittlungsschicht**: Der Vermittlungsdienst wird von IP (Internet Protocol, RFC 791 [Post 81a]) bereitgestellt. Die Identifizierung von Rechnern geschieht über IP-Adressen, die Wegewahl zwischen unterschiedlichen Netzen über IP-Router. Für die Verbreitung der Wege- und Steuerinformationen gibt es auf dieser Schicht spezielle Hilfs-Protokolle.

- **Netzanschlußschicht**: Diese Schicht vereint die Funktionen der OSI-Schichten 1 und 2.

Beim Transport von Daten zwischen Anwendungen auf zwei über ein Netzwerk verbundenen Rechnern wird jede Schicht durchlaufen. Ausgehend von der Quell-Anwendung werden die Daten von einer Schicht an die jeweils darunterliegende Schicht weitergeleitet. Auf der untersten Schicht werden die Daten über das Netzwerk zum Zielrechner transportiert, wo die Daten dann wiederum durch alle Schichten hindurch nach oben zur entprechenden Ziel-Anwendung weitergereicht werden (vgl. Abbildung 6).
Für die Kommunikation zwischen den Schichten dienen fest definierte Dienstzugangspunkte (SAP, Service Access Point), über welche jede Schicht der jeweils übergeordneten Schicht ihre Dienste bereitstellt. Diese Isolierung einzelner Funktionen der Datenübertragung in den einzelnen Schichten bedingt eine weitgehende Unabhängigkeit der in den Schichten laufenden Prozesse. Festgeschrieben sind nur die Schnittstellen zwischen den Schichten, nicht aber die konkrete Implementierung oder die darauf aufsetzenden Anwendungen.

Abbildung 2 zeigt eine Auswahl von Internetprotokollen. Auf die im Rahmen des Praktikums relevanten Protokolle werden wir im Folgenden kurz eingehen.

1.2 Ethernet-Überblick

Das im LAN-Bereich heute gebräuchlichste Protokoll der Bitübertragungs- und Sicherungsschicht ist das Ethernet-Protokoll (RFC 894 [Horn 84][1]). Andere LAN-Topologien wie Token Ring sind heute nur mehr vereinzelt im Einsatz.
Jeder Rechner (genauer: jedes Netzwerk-Interface) wird im Ethernet über seine MAC-Adresse identifiziert. Im Normalfall erhält jedes Netzwerk-Interface vom Hersteller eine weltweit eindeutige MAC-Adresse zugewiesen. Sie hat eine Länge von sechs Bytes, die ersten drei Bytes beinhalten die Herstellerkennung, die folgenden drei Bytes eine fortlaufende Nummer. Z.B. steht bei der MAC-Adresse `08:00:20:F5:BE:3C` die Herstellerkennung `08:00:20` für Sun Microsystems, `F5:BE:3C` ist die fortlaufende Nummer.
Bei Bedarf kann die ursprüngliche MAC-Adresse mit dem Kommando `ifconfig` mit einer beliebigen neuen MAC-Adresse überschrieben werden.

Einige Ethernet-Adressbereiche sind für spezielle Zwecke (meist Multicasts für Protokolle auf höheren Schichten zur gleichzeitigen Adressierung mehrerer Rechner im LAN, z.B. alle Router) reserviert, die Adresse `ff:ff:ff:ff:ff:ff` ist die Broadcast-Adresse und dient zum Adressieren aller Geräte im LAN-Segment. Sie wird z.B. vom ARP-Protokoll (Siehe Abschnitt 1.3) verwendet.

Eine Auflistung der Herstellerkennungen, der Ethernet-Adressbereiche, der Ethernet-Types sowie weitere Informationen zu Ethernet finden Sie unter [Patt 98].

Im Ethernet sind die Zugriffe auf das von mehreren Stationen benutzte (shared) Medium (früher einfach ein gemeinsames Kabel) nicht streng geregelt. Im Gegensatz zu Token Ring, wo garantiert wird, daß zu einem Zeitpunkt immer nur eine Station sendet, kann bei Ethernet jede Station, die das Medium als frei erachtet, Datenpakete senden. Dazu implementiert

[1] Eine etwas andere Spezifikation wird von IEEE 802 in RFC 1042 [PoRe 88] festgelegt, die Spezifikation nach RFC 894 ist aber die weitaus gebräuchlichere und wird hier dargestellt.

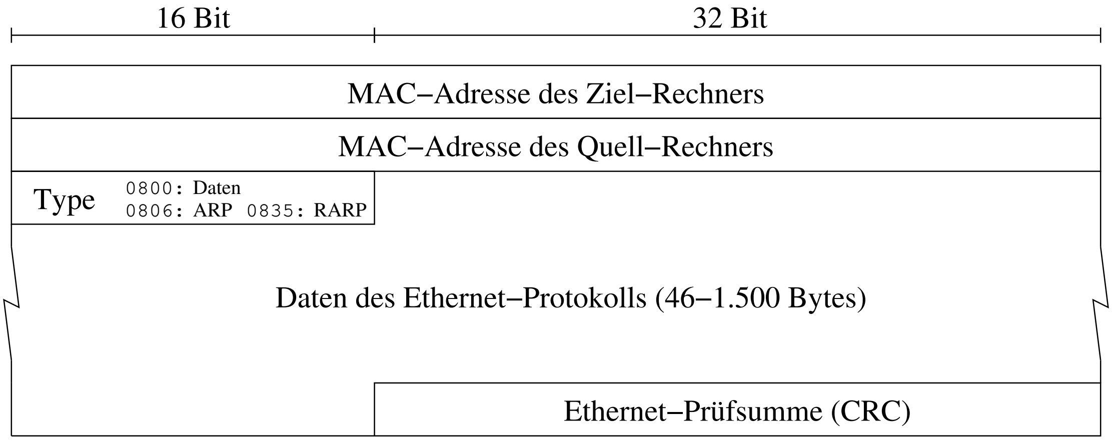

Abbildung 3: Aufbau des Ethernet-Headers nach RFC 894

Ethernet spezielle Mechanismen, die Mehrfachzugriffe erkennbar und korrigierbar machen (CSMA/CD, Carrier Sense, Multiple Access, Collision Detection). Diese Mechanismen sollen hier nicht näher betrachtet werden, wichtig ist allerdings die Unterscheidung der zwei heute gebräuchlichsten Techniken zur Bildung eines Ethernet-Segmentes, Repeater (Hubs) und Switches.

Ein **Repeater** ist ein Gerät, welches es erlaubt, mehrere Geräte auf Netzanschlußebene miteinander zu verbinden. Dabei verhält sich der Repeater grob gesagt wie ein Kabelstück, welches alle Signale aufnimmt und an alle angeschlossenen Stationen wieder ausgibt. Er führt jedoch auch eine Verstärkung und ggf. Rekonstruktion der Datenimpulse auf Signalebene durch, kann die Datenpakete aber nicht interpretieren.
Beim Repeater gelten alle Annahmen für ein von mehreren Stationen gemeinsam benutztes Medium. Alle Stationen empfangen alle im Medium übertragenen Pakete, verarbeiten (im Normalfall) aber nur für sie bestimmte Pakete und Broadcasts.

Ein **Switch** dient dem selben Zweck wie ein Repeater, er führt aber auf Netzanschlußebene eine Filterung des Verkehrs nach MAC-Adressen durch. Dazu merkt sich der Switch in einer Tabelle für jeden Port (hier: Netzwerkanschluß) alle angeschlossenen MAC-Adressen und gibt Pakete nur an dem Port aus, an welchem die MAC-Adresse auch wirklich angeschlossen ist. Nur Pakete für bisher noch nicht aufgetretene MAC-Adressen, Broadcasts und Multicasts werden auf allen Ports ausgegeben. Des Weiteren überprüft der Switch die Pakete auf Korrektheit und verwirft unvollständige oder fehlerhafte Datenpakete (siehe Abbildung 3, CRC, Cyclic Redudancy Check).[2]

Bei einem Repeater kann der gesamte Verkehr im LAN-Segment an einem Port abgehört werden. Dazu muß lediglich die Netzwerkkarte des Rechners so konfiguriert werden, daß

[2]Einige Switches haben an allen Ports eigene MAC-Adressen. Diese spielen jedoch bei der Rechner-Rechner-Kommunikation keine Rolle sondern werden nur für spezielle Ethernet-Management-Protokolle benötigt.

sie nicht nur die für sie bestimmten, sondern alle Pakete ans Betriebssystem weitergibt (Promiscuous Mode). Mit geeigneten Programmen können diese Daten dann aufgezeichnet und analysiert werden. Bei einem Switch ist dies im Normalfall nicht möglich[3].

Die MAC-Adressen ermöglichen eine eindeutige Identifizierung eines Rechners im lokalen Netzwerk. Da sie jedoch aufgrund ihrer zufälligen Verteilung auf viele verschiedene Netze keine Adressgruppierung und -strukturierung gestatten, sind sie nicht dazu geeignet, eine Adressierung in großen, weltweiten Netzen bereitzustellen. Die Kopplung von lokalen Netzen zu einem großen, weltweiten Netz erfolgt daher über IP-Router, die Adressierung über IP-Adressen auf höherer Ebene (Vermittlungsschicht). An der Schnittstelle zwischen Ethernet und IP ist also eine Adressumsetzung zwischen MAC- und IP-Adresse nötig. Diese Aufgabe erfüllen die Protokolle ARP und RARP.

1.3 Address Resolution Protocol (ARP)

Für die dynamische Zuordnung der IP-Adressen zu den dazugehörigen MAC-Adressen ist ARP[4] zuständig (RFC 826 [Plum 82]).
Um innerhalb eines lokalen Netzes IP-Adressen in MAC-Adressen übersetzen zu können, wird auf jedem Rechner eine Tabelle (ARP-Tabelle, ARP-Cache) benutzt, welche die entsprechenden Daten aufnimmt (vgl. Abbildung 4). Will ein Rechner eine Verbindung zu einem Rechner im selben Subnetz aufbauen, versucht er zunächst, anhand der Daten in der ARP-Tabelle die passende MAC-Adresse zu ermitteln. Ist die Adresse dort nicht vorhanden, schickt er eine Anfrage nach der zur IP-Adresse des Ziel-Rechners passenden MAC-Adresse an alle Rechner im Netz (Ethernet-Broadcast). Der Rechner, welcher die IP-Adresse besitzt antwortet auf diese Anfrage mit seiner MAC-Adresse. Alle anderen Rechner ignorieren die Anfrage. Die so ermittelte MAC-Adresse wird für eine gewisse Zeit (typischerweise ca. 5 Minuten) in der ARP-Tabelle vorgehalten und kann zur weiteren Kommunikation dort ausgelesen werden.
Auf Unix-Systemen können feste Zuordnungen von einer MAC-Adresse zu einer IP-Adresse (statische ARP-Einträge) in der Datei `/etc/ethers` eingetragen werden.

1.4 Reverse Address Resolution Protocol (RARP)

RARP (RFC 903 [FMMT 84]) ist das Gegenstück zu ARP, es wandelt MAC-Adressen in IP-Adressen um. Es wird nicht direkt für die IP-Kommunikation im Ethernet benötigt, sondern im Wesentlichen dazu, Endgeräten ohne permanenten Speicher (z.B. X-Terminals) aufgrund ihrer eindeutigen MAC-Adresse eine IP-Adresse zuzuteilen. Dazu antwortet

[3] Die meisten Switches bieten allerdings die Möglichkeit, sogenannte Mirroring-Ports zu konfigurieren, an welchen alle Pakete wie bei einem Hub ausgegeben werden. Dies wird z.B. für Intrusion Detection-Systeme (IDS) oder für die Fehlersuche benötigt.

[4] ARP ist prinzipiell nicht nur auf Ethernet und IP beschränkt, sondern kann auch für andere Protokollkombinationen verwendet werden.

```
sol:~# arp -a
Net to Media Table
Device   IP Address               Mask      Flags   Phys Addr
------ -------------------- --------------- ----- ---------------
hme0   192.168.90.250       255.255.255.255       00:00:0c:07:ac:e7
qfe1   10.0.51.27           255.255.255.255       00:00:00:00:b0:03
hme1   10.0.50.41           255.255.255.255       00:00:00:00:b0:07
hme1   10.0.50.21           255.255.255.255       00:50:8b:e2:bc:46
qfe0   10.0.50.1            255.255.255.255 SP    08:00:20:d1:40:44
hme0   224.0.0.0            240.0.0.0       SM    01:00:5e:00:00:00
```

Abbildung 4: ARP-Tabelle unter Solaris

ein dafür konfigurierter Server auf die RARP-Anfrage mit den Daten in seiner Datei `/etc/ethers`. Das Endgerät kann daraufhin die weitere Konfiguration und das gesamte Betriebssystem von einem Boot-Server beziehen.
RARP wird zu diesem Zweck vor allem von den beiden Protokollen BOOTP (RFC 951 [CrGi 85]) und DHCP (RFC 1541 [Drom 93]) verwendet.

1.5 Internet Protocol (IP) und Routing

Alle folgenden Beschreibungen des Internet-Protokolls beziehen sich auf die aktuelle Version 4 (IPv4). Diese Version soll insbesondere aufgrund der IPv4-Adressverknappung langfristig durch die Folgeversion 6 (IPv6) abgelöst werden. Von einer Migration sind fast alle Bereiche des Netzes aber noch weit entfernt, daher werden wir hier nicht weiter auf IPv6 eingehen.

Adressierung

Alle Rechner, die IP verwenden, werden im IP-Header (Abbildung 5) durch eindeutige 32-bit Adressen, den sogenannten Internetadressen, identifiziert. Die 4 Bytes der IP Adresse werden meist als durch Punkte getrennte Dezimalzahlen geschrieben (z.B. `192.168.215.81`).

Eine IP-Adresse besteht aus zwei Teilen: einem Netzadress-Teil und einem Hostadress-Teil. Der erste Teil der Adresse bestimmt das Netz, in dem sich der Zielrechner befindet. Der zweite Teil der Adresse identifiziert den Rechner innerhalb des Netzes. Die Länge der Netzadresse variiert in Abhängigkeit von der Größe des Netzes. Es gibt zwei Möglichkeiten, die Länge der Netzadresse festzulegen. Früher wurden die IP-Adressen in Klassen (A, B, C und weitere) unterteilt, die jeweils festgelegte Längen für die Netzadressen und somit feste Netzmasken hatten (vgl. Tabelle 1).

Dieses Verfahren wurde mittlerweile von CIDR (Classless Inter-Domain Routing, RFC 1519 [FLYV 93]) abgelöst. Hierbei wird die Länge der Netzadresse durch die explizite Angabe der Netzmaske bestimmt. Die Netzmaske gibt an, wieviele Bits der IP-Adresse das Rechnernetz identifizieren. Die Netzmaske ist ebenfalls ein 32-bit Wert, bei dem alle Bits, die

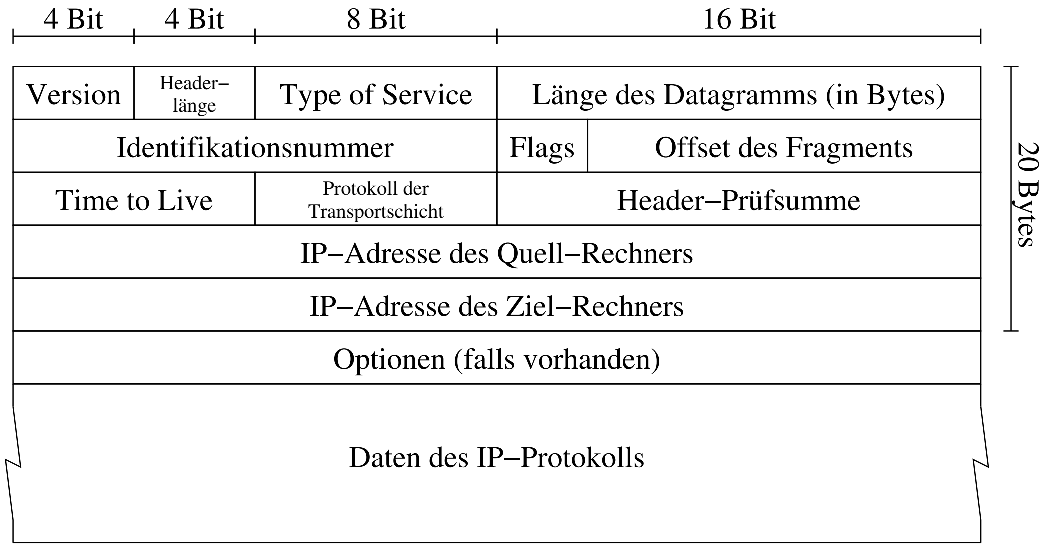

Abbildung 5: Aufbau des IP-Headers

Klasse	**Netzadressen**	**Standard-Netzmaske**	**reservierte Netze**	
`A`	`0-127.0.0.0`	`255.0.0.0`	`0.0.0.0`	Default-Route
			`10.0.0.0`	privater Adressbereich
			`127.0.0.1`	Loopback-Interface
`B`	`128-191.xxx.0.0`	`255.255.0.0`	`172.16-31.0.0`	privater Adressbereich
`C`	`192-223.xxx.xxx.0`	`255.255.255.0`	`192.168.0.0`	privater Adressbereich
`D, E`	`224-255.xxx.xxx.xxx`	spezielle Multicast-Adressen bzw. reserviert für zukünftige Anwendungen		

Tabelle 1: IP-Netzwerkklassen und reservierte Netzbereiche

das Netz identifizieren, auf 1 und alle Bits für die Hostadresse auf 0 gesetzt werden. Das heißt, ein Netz mit einer 16 Bit Netzadresse hat die Netzmaske `255.255.0.0`.
Tabelle 2 zeigt ein Beispiel für die Unterteilung einer IP-Adresse in Netz- und Host-Teil, die Tabelle 14 im Anhang listet alle möglichen Netzmasken und die unterschiedlichen Notationen auf.

Einige IP-Adressbereiche sind für spezielle Zwecke reserviert. Die Adresse `0.0.0.0` bezeichnet auf jedem Rechner seine jeweilige Default-Route[5].
`127.0.0.1` ist die Adresse des loopback Device mit Namen `localhost`. Sie addressiert immer den eigenen Rechner.
Für eine ausschließlich interne Verwendung sind Blöcke von privaten IP-Adressen reserviert, die beliebig verwendet werden können (RFC 1918 [RMK$^+$ 96]). Diese Netze werden im Internet nicht geroutet und können daher auch nicht für eine direkte Kommunikation mit Rechnern im Internet verwendet werden.
Die Adressen der Klasse D werden z.B. von Routing-Protokollen für den Austauch von

[5] auf einigen Systemen erscheint dafür in der Routing-Tabelle der Eintrag `default`

	Dezimalformat	**Binärformat**	
		Netzadress-Teil	**Hostadress-Teil**
Netzmaske	`255.255.224.0`	`11111111.11111111.111`	`00000.00000000`
Netzadresse	`192.168.192.0`	`11000000.10101000.110`	`00000.00000000`
erste IP-Adresse	`192.168.192.1`	`11000000.10101000.110`	`00000.00000001`
...	...	...	...
n-te IP-Adresse	`192.168.215.81`	`11000000.10101000.110`	`10111.01010001`
...	...	...	...
letzte IP-Adresse	`192.168.223.254`	`11000000.10101000.110`	`11111.11111110`
Broadcast-Adresse	`192.168.223.255`	`11000000.10101000.110`	`11111.11111111`

Tabelle 2: Beispiel für die Einteilung von IP-Adressen mit Netzmasken

Routeninformationen verwendet, sollen hier jedoch wie die Adressen der Klasse E nicht weiter behandelt werden.

Soll ein Rechnernetz in das Internet integriert werden, so muß der Administrator einen Block von offiziellen IP-Adressen beantragen. Er erhält dann von seinem Internet-Provider neben der zugewiesen Netzadresse auch eine Netzmaske. Die Adressen von Rechnern innerhalb eines Netzes können frei vergeben werden. Nur zwei Werte innerhalb eines jeden Netzes sind für spezielle Zwecke reserviert:

- die Netzadresse, bei der alle Hostbits auf 0 gesetzt sind, und
- die Broadcastadresse, bei der alle Hostbits auf 1 gesetzt sind.

Die Broadcast-Adresse wird verwendet, um alle Rechner innerhalb eines Netzes anzusprechen.

Durch die Verwendung von Netzmasken ist es auch möglich, ein Netz in weitere kleinere Teilnetze zu unterteilen, deren Verwaltung an andere übergeben werden kann.

Routing

Doch wie findet ein Paket sein Ziel, wenn nur die Zieladresse bekannt ist? Da das Internet aus vielen einzelnen autonomen Netzen besteht, die alle miteinander verbunden sind, gibt es von einem Sender zu einem Zielrechner oft mehrere Wege. Das Internet besteht heute aus mehreren Millionen Rechnern. Diese zwei Tatsachen verdeutlichen, daß es unmöglich ist, daß jeder Rechner im Internet den Weg zu allen anderen Rechnern kennt, mit denen er jemals kommunizieren möchte. Damit die Pakete dennoch ihren Weg zum Ziel finden, wurden an den Übergangspunkten zwischen den einzelnen Netzen Router eingerichtet.

Router sind Rechner, die Pakete in Abhängigkeit von der Zieladresse in ein anderes Netz weiterleiten. Jeder Router kennt die Netze, an die er angeschlossen ist, und für alle ihm bekannten Ziel-Adressen den nächsten Nachbarn (Next Hop), an welchen er ein ankommendes Paket weiterleiten muß. Durch die Weiterleitung von einem Router zum nächsten kommt das Paket seinem Ziel Schritt für Schritt näher, bis es im Zielnetz angekommen ist

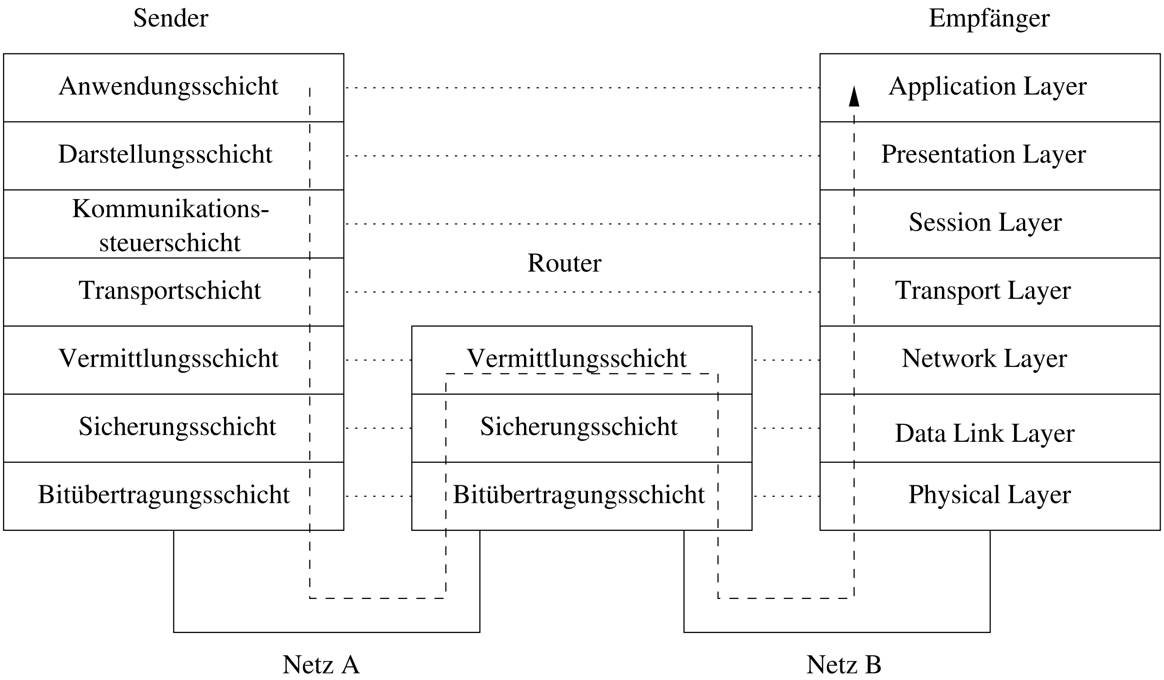

Abbildung 6: IP-Routing

und an den richtigen Rechner geliefert wird.
Für die Verteilung der Wegeinformationen gibt es in der Vermittlungsschicht neben dem IP-Protokoll spezielle Routing-Protokolle. Beispiele sind OSPF (Open Shortest Path First, RFC 1247 [Moy 91]), RIP (Routing Information Protocol, RFC 1058 [Hedr 88]) sowie BGP (Border Gateway Protocol, RFC 1267 [LoRe 91]).
Durch die Verwendung von Routern muß jeder Rechner im Internet nur noch wissen, an welche Netze er angeschlossen ist und über welchen direkt erreichbaren Router er die Netze seiner Kommunikationspartner erreichen kann. Der Router analysiert die an ihn geschickten Pakete auf der Vermittlungsschicht (siehe Abbildung 6) und entscheidet dann, an welchen Rechner oder weiteren Router die Pakete im nächsten Schritt gesendet werden müssen.
In den meisten LANs ist es so, daß jeder Rechner nur eine Route für die Rechner innerhalb seines Netzes kennt (diese ist implizit durch die eigene IP-Adresse und Netzmaske festgelegt) und alle anderen Pakete an seinen Default-Router im eigenen LAN-Segment weiterleitet. Die Route, die festlegt, an welchen Router alle Pakete mit Zieladressen ohne explizite Route gesendet werden, heißt Default-Route.

In unserem Beispiel aus Tabelle 2 erreicht der Rechner mit der IP-Adresse `192.168.215.81` die Adressen von `192.168.192.1` bis `192.168.223.254` direkt, alle anderen Adressen über seinen Default-Router, also z.B. die `192.168.192.2`.

```
sol:~# netstat -rvn

IRE Table: IPv4
  Destination             Mask           Gateway          Device Mxfrg  Rtt   Ref Flg  Out  In/Fwd
-------------------- --------------- -------------------- ------ ----- ----- --- --- ----- ------
10.40.119.80        255.255.255.240  10.40.119.82         sbif3   1500*     0   1 U    17642      0
10.12.205.64        255.255.255.192  10.12.205.66         sbif1   1500*     0   1 U     9166      0
10.12.136.0         255.255.255.192  10.12.136.60         sbif2   1500*     0   1 U    12359      0
10.0.2.0             255.255.255.0   10.0.2.1             sbif0   1500*     0   1 U        7      0
10.146.181.0         255.255.255.0   10.40.119.94                 1500*     0   1 UG     119      0
10.146.211.0         255.255.255.0   10.40.119.94                 1500*     0   1 UG   26075      0
10.146.208.0         255.255.255.0   10.40.119.94                 1500*     0   1 UG       0      0
default              0.0.0.0         10.12.205.126                1500*     0   1 UG    8941      0
127.0.0.1            255.255.255.255 127.0.0.1            lo0     8232*     0  11 UH  252272      0
```

Abbildung 7: Routing-Tabelle unter Solaris

1.6 Zusammenspiel Ehternet/IP/ARP

Anhand des Beispiels aus Abbildung 8 wird das Zusammenspiel der Protokolle Ethernet, IP und ARP genauer erläutert. Der Client `53.122.1.2` will ein IP-Paket zum Server `53.122.2.2` schicken. Die ARP-Tabellen aller Geräte sei anfangs leer.

Nach Absetzen des Kommandos auf dem Client, in unserem Beispiel sei es `ping 53.122.2.2`, erreichen die Daten (ICMP-Echo-Request, s. Seite 31) die Vermittlungsschicht. Dort muß anhand der Routing-Tabelle entschieden werden, an welches Gerät die Daten als nächstes weitergeleitet werden sollen. Da die IP-Adresse `53.122.2.2` nicht im Netz `53.122.1.0/24` des Clients liegt greift die default-Route zu `53.122.1.1`, das Datenpaket muß also an den Router geschickt werden. Die Adressierung des Routers `53.122.1.1` kann jedoch nicht auf IP-Ebene erfolgen, da ja die eigentliche Ziel-IP-Adresse, die `53.122.2.2` des Servers, nicht aus dem Paket entfernt werden darf. Das Paket bleibt also auf IP-Ebene an die `53.122.2.2` adressiert, muß jedoch auf Ethernet-Ebene (Netzanschlußschicht) an den Router adressiert werden.
Der Client benötigt im nächsten Schritt die MAC-Adresse des Routers. Da diese noch nicht in seiner ARP-Tabelle enthalten ist wird er eine ARP-Anfrage (Quell-MAC-Adresse: `00:10:83:01:EE:A2`, Ziel: `ff:ff:ff:ff:ff:ff`) in sein Ethernet-Segment schicken. Diese Anfrage enthält die IP-Adresse des gesuchten Ziels (`53.122.1.1`). Der Router antwortet nun als Besitzer der angesprochenen IP-Adresse auf diese Anfrage mit einer ARP-Antwort (Quell-MAC-Adresse: `00:00:0C:07:AC:B5`, Ziel: `00:10:83:01:EE:A2`) und seiner MAC-Adresse von `e0`.
Der Client trägt nun die Adresskombination `53.122.1.1/00:00:0C:07:AC:B5` in seine ARP-Tabelle ein, wo sie eine Zeit lang zwischengespeichert wird. Erst jetzt kann er das urspüngliche ICMP-Echo-Paket abschicken.
Der Router nimmt das Paket entgegen und weiß aufgrund seiner Routing-Tabelle, daß die Ziel-IP-Adresse `53.122.2.2` direkt über sein Interface `e1` erreichbar ist. Über den oben beschriebenen ARP-Mechanismus ermittelt er die zur Ziel-IP-Adresse passende MAC-Adresse und kann das ICMP-Paket nun direkt an den Server zustellen. Als Absender bleibt im

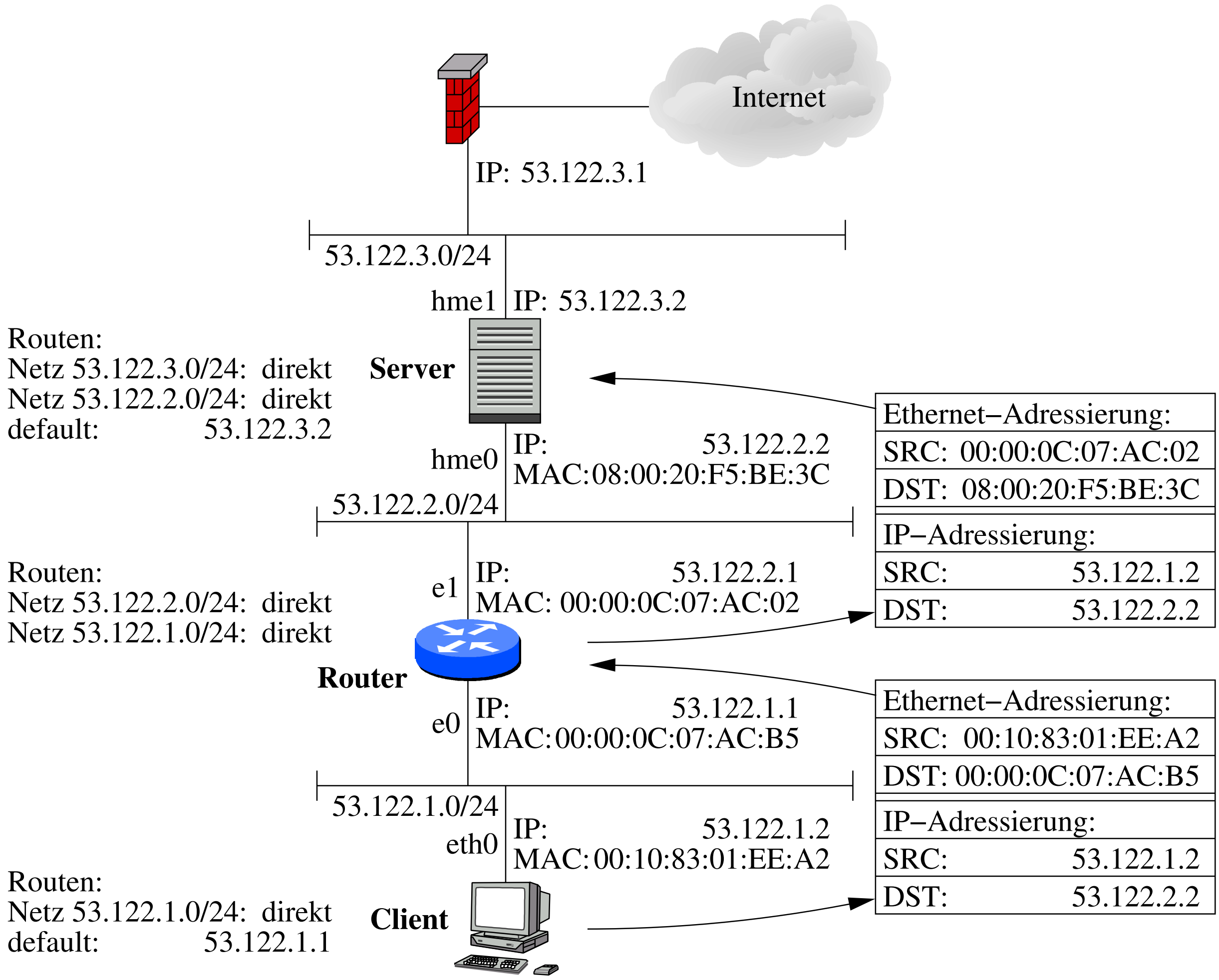

Abbildung 8: Zusammenspiel Ethernet/IP/ARP

IP-Header die Adresse des Clients, `53.122.1.2`, erhalten, als Absender-Adresse auf MAC-Ebene wird jedoch jetzt die Adresse von `e1` des Routers (`00:00:0C:07:AC:02`) im Ethernet-Header eingetragen. Die Ziel-Adressen sind auf beiden Ebenen die des Servers.
Der Protokollstapel des Servers erkennt das Paket als für ihn selbst bestimmt und antwortet darauf mit einem ICMP-Echo-Reply-Paket. Dieses findet über die selben Mechanismen wieder seinen Weg zum Client. Es entfallen allerdings alle ARP-Anfragen, da die beteiligten Geräte aufgrund der vorangegangenen Kommunikation schon alle benötigten MAC/IP-Adresskombinationen in deren ARP-Tabellen vorliegen haben.

1.7 Protokollnummern

Nachdem die Datenpakete den Zielrechner erreicht haben müssen sie bekanntlich den Protokollstapel wieder nach oben wandern. Zur Identifikation der Protokolle auf den einzelnen Schichten im TCP/IP-Stapel dienen dabei eigene Nummernfelder in den einzelnen Protokoll-Headern.

Die Protokollnummer (8 Bit) im Protokoll-Feld des IP-Headers (vgl. Feld "Protokoll der Transportschicht" aus Abbildung 5) bestimmt, an welches Protokoll der Transportschicht (TCP, UDP, ICMP und weitere) die Daten vom IP-Protokoll zu übergeben sind. Diese Daten enthalten wiederum die Header aller Protokolle der darüberliegenden Schichten und die eigentlichen Nutzdaten.
Die einem Unix-System bekannten Protokollnummern findet man in der Datei `/etc/protocols`.

```
linux:~# cat /etc/protocols
# /etc/protocols:
# $Id: protocols,v 1.1 1995/02/24 01:09:41 imurdock Exp $
#
# Internet (IP) protocols
#
#       from: @(#)protocols     5.1 (Berkeley) 4/17/89
#
# Updated for NetBSD based on RFC 1340, Assigned Numbers (July 1992).

ip      0       IP              # internet protocol, pseudo protocol number
icmp    1       ICMP            # internet control message protocol
igmp    2       IGMP            # Internet Group Management
ggp     3       GGP             # gateway-gateway protocol
ipencap 4       IP-ENCAP        # IP encapsulated in IP (officially ``IP'')
st      5       ST              # ST datagram mode
tcp     6       TCP             # transmission control protocol
...
```

Abbildung 9: Datei `/etc/protocols` unter Linux (debian)

1.8 Transmission Control Protocol (TCP)

Anwendungen, welche auf eine zuverlässige Übertragung der Daten angewiesen sind, benutzen TCP als Transportprotokoll (Protokollnummer 6).
TCP benutzt für eine zuverlässige Übertragung einen Mechanismus namens Positive Acknowledgement with Re-Transmission (PAR, positive Bestätigung mit Neuübertragung). Im Wesentlichen bedeutet dies, daß ein Rechner die Daten nach einer gewissen Zeit erneut sendet, solange er nicht von der Gegenstelle die Bestätigung erhält, daß sie korrekt empfangen wurden. Jedes Datensegment enthält eine Prüfsumme anhand welcher der Empfänger die Integrität der Daten prüfen kann. Gültig empfangene Datensegmente werden dem Absender mit einer Meldung positiv bestätigt. Ungültige Datensegmente werden vom Empfänger ignoriert.

TCP arbeitet verbindungsorientiert, das Protokoll stellt also eine logische Verbindung zwischen den beiden Prozessen her. Dafür werden spezielle Bits oder Flags im TCP-Header (siehe Abbildung 10) verwendet.

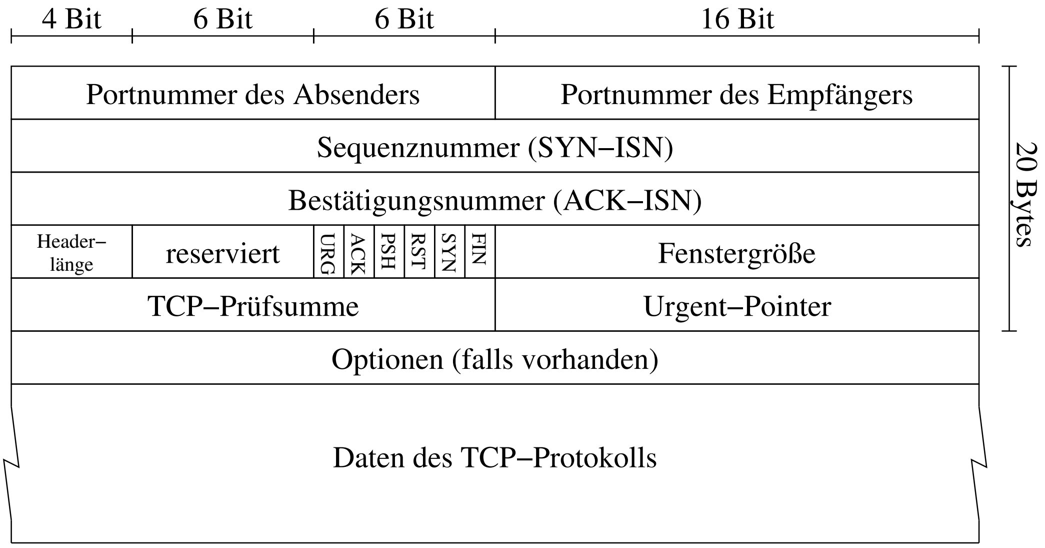

Abbildung 10: Aufbau des TCP-Headers

- **SYN**: Synchronize Sequence Numbers, Aufforderung zur Synchronisation der ISNs (Initial Sequence Numbers) für den Verbindungsaufbau
- **ACK**: Acknowledgement, Bestätigung einer Anfrage des Kommunikationspartners oder für den korrekten Empfang von Netzdaten
- **FIN**: Finished, Aufforderung zum Beenden der Verbindung
- **RST**: Reset Connection, Verbindung zurücksetzen/ablehnen (Connection refused)

Zwei weitere Flags, **PSH**, Push Data (Aufforderung an den Empfänger, die Daten an die darüberliegende Schicht weiterzuleiten) und **URG**, Urgent Data (kennzeichnet einen bestimmten Datenbereich als "dringend") dienen der Datenflußsteuerung und haben für den Verbindungsauf- und -abbau keine Bedeutung.

Vor der Übertragung der Nutzdaten findet ein sogenannter 3-Wege-Handshake statt. In Abbildung 11 ist als Beispiel eine Telnet-Verbindung vom Client-Port 1045 zum Server-Port 23 dargestellt. Der Quellrechner (Client) schickt für den Verbindungsaufbau ein Paket (Segment 1) mit gesetztem SYN-Bit (SYN-Flag) und seiner ISN an den Zielrechner (Server). Der Server quittiert den Erhalt der Nachfrage auf Verbindungsaufbau durch ein Paket mit gesetztem SYN-Bit, seiner ISN sowie mit gesetztem ACK-Bit und eine um eins erhöhte Client-ISN. Nachdem der Client den Verbindungsaufbau nochmal durch ein ACK-Paket (Segment 3) mit um eins erhöhter Server-ISN bestätigt hat, kann die eigentliche Datenübertragung beginnen.
Die ISNs werden während der Datenübertragung ständig inkrementiert und garantieren die Einhaltung der richtigen Paket-Reihenfolge beim jeweiligen Empfänger.

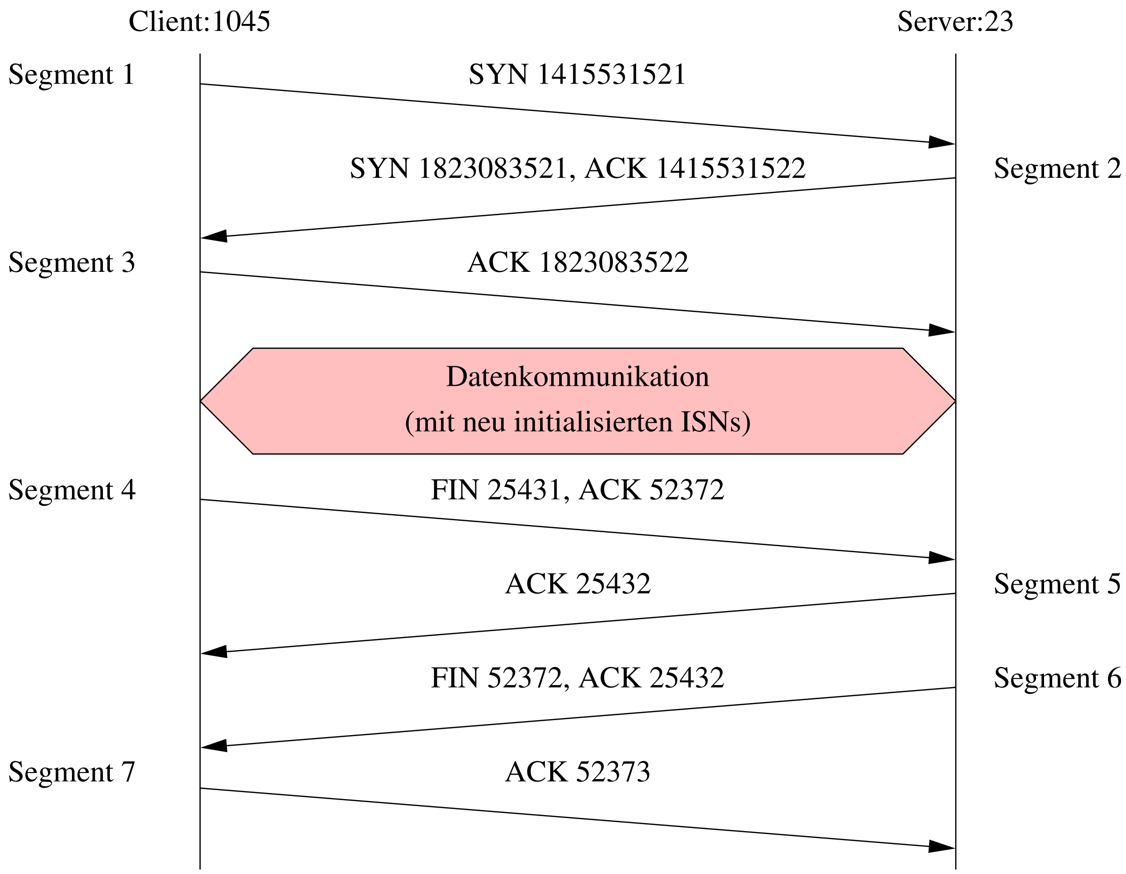

Abbildung 11: TCP-Verbindungsauf- und -abbau mit Flags und ISNs [Stev 96, Seite 232]

Zum Abbau der Verbindung vom Client aus sendet dieser ein Paket mit gesetztem FIN-Bit an den Server. Dieser bestätigt den Abbau der Client-Server-Verbindung (Segment 5) und initiiert gleich darauf den Abbau der Server-Client-Verbindung[6] (Segment 6). Bei einigen Implementierugen werden die Segmente 5 und 6 auch zu einem Segment zusammengefaßt, welches sowohl den Abbau der Client-Server-Verbindung bestätigt als auch die Server-Client-Verbindung abbaut. Der Verbindungsabbau kann sowohl vom Client als auch vom Server initiiert werden.
TCP enthält noch eine Reihe weiterer Funktionen zur Verkehrsfluß-Kontrolle, die aber hier nicht weiter betrachtet werden.

[6] Im Prinzip könnten nach Abbau der Client-Server-Verbindung in der Gegenrichtung noch Daten übertragen werden (TCP half close), in der Praxis gibt es jedoch kaum Applikationen, die diese Möglichkeit nutzen.

1.9 User Datagram Protocol (UDP)

UDP mit der Protokollnummer 17 ist ein verbindungsloses Protokoll und beinhaltet im Gegensatz zu TCP keine Mechanismen zum Verbindungsaufbau, zur Verkehrsflußsteuerung oder für die Festlegung der Paketreihenfolge.
Die Anwendungsprogramme bekommen über UDP einen schnellen Zugriff auf die IP-Ebene, müssen jedoch auf höheren Ebenen selbst für die benötigte Datenzuverlässigkeit sowie die Einhaltung der logischen Paketreihenfolge sorgen.

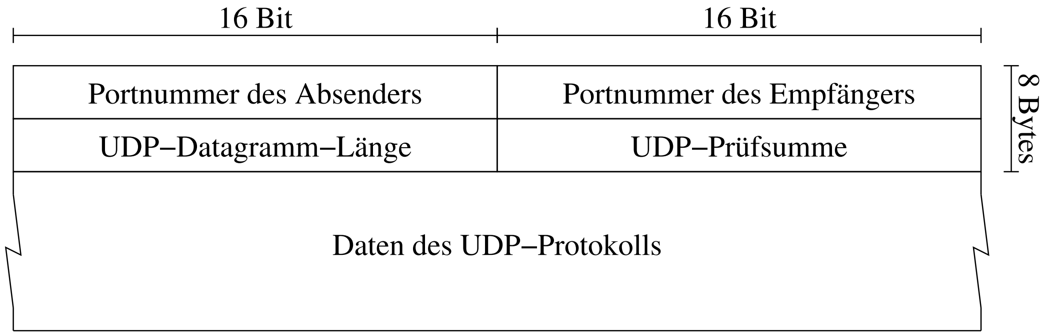

Abbildung 12: Aufbau des UDP-Headers

UDP wird unter anderem von SNMP und für DNS-Anfragen (siehe Abschnitt 6.1.1) verwendet.

1.10 Ports und Sockets

Zur Weiterleitung der Datenpakete vom Transportprotokoll an den richtigen Anwendungsprozeß dienen bei TCP und UDP die Portnummern (16 Bit) oder Ports. Dabei enthält jedes Datenpaket den Port des Absender-Prozesses (source port number) und den des Ziel-Prozesses (destination port number). Der Port des Absender-Prozesses dient dazu, dem Zielprozeß die Nummer seines Kommunikationspartners zu übermitteln. Über den Port des Ziel-Prozesses wird der Prozeß ermittelt, für welchen das Datenpaket bestimmt ist. Analog zu den Protokollnummern aus der Datei `/etc/protocols` (Abbildung 9) enthält die Datei `/etc/services` in Abbildung 13 alle dem Unix-System bekannten Portnummern.

Die Portnummern unterhalb von 1024 sind für Standard-Dienste wie Telnet (Port 23), HTTP (Port 80) oder SSH (Port 22) reserviert (well known ports) und in der Regel statisch zugewiesen[7]. Diese Zuweisung ist allerdings nicht bindend und bedeutet nicht, daß z.B. hinter Port 80 nur HTTP als Protokoll möglich ist. Prinzipiell kann jeder Dienst hinter jedem Port konfiguriert sein.

[7]Nur der Benutzer `root` darf unter Unix Ports kleiner 1024 zuweisen, eine Ausnahme bilden hier die sogenannten R-Kommandos (rsh, rlogin, rcp usw.)

```
linux:~# cat /etc/services
# /etc/services:
# $Id: services,v 1.4 1997/05/20 19:41:21 tobias Exp $
#
# Network services, Internet style
#
# Note that it is presently the policy of IANA to assign a single well-known
# port number for both TCP and UDP; hence, most entries here have two entries
# even if the protocol doesn't support UDP operations.
# Updated from RFC 1700, ``Assigned Numbers'' (October 1994).  Not all ports
# are included, only the more common ones.

tcpmux          1/tcp                           # TCP port service multiplexer
echo            7/tcp
echo            7/udp
discard         9/tcp           sink null
discard         9/udp           sink null
systat          11/tcp          users
daytime         13/tcp
daytime         13/udp
netstat         15/tcp
qotd            17/tcp          quote
msp             18/tcp                          # message send protocol
msp             18/udp                          # message send protocol
chargen         19/tcp          ttytst source
chargen         19/udp          ttytst source
ftp-data        20/tcp
ftp             21/tcp
fsp             21/udp          fspd
ssh             22/tcp                          # SSH Remote Login Protocol
ssh             22/udp                          # SSH Remote Login Protocol
telnet          23/tcp
...
```

Abbildung 13: Datei `/etc/services` unter Linux (debian)

Alle Ports über 1024 können von normalen Benutzern dynamisch zugewiesen werden (dynamically allocated ports). Diese dynamischen Ports ermöglichen mehrere parallele Verbindungen eines Dienstes zwischen zwei Systemen und somit auch mehrere gleichzeitige Benutzer. Jede neue Verbindung erhält einen noch freien dynamischen Port zugewiesen und ist damit eindeutig im System identifizierbar.

Die Kombination aus IP-Adresse und Portnummer wird als Socket bezeichnet und identifiziert eindeutig einen Netzwerkprozeß im gesamten Netz. Eine Kommunikationsbeziehung kann über die Quell- und Ziel-IP-Adresse zusammen mit dem Quell- und Ziel-Port eindeutig identifiziert werden.

1.11 Internet Control Message Protocol (ICMP)

ICMP (Protokollnummer 1) dient zur Steuerung des IP-Verkehrs sowie zur Übermittlung von Netzwerk-Statusinformation. Dazu beinhaltet ICMP mehrere Arten von Meldungen. Diese werden über den sogenannten ICMP-Type unterschieden, eine genauere Klassifizierung innerhalb der ICMP-Typen erfolgt über den ICMP-Code.

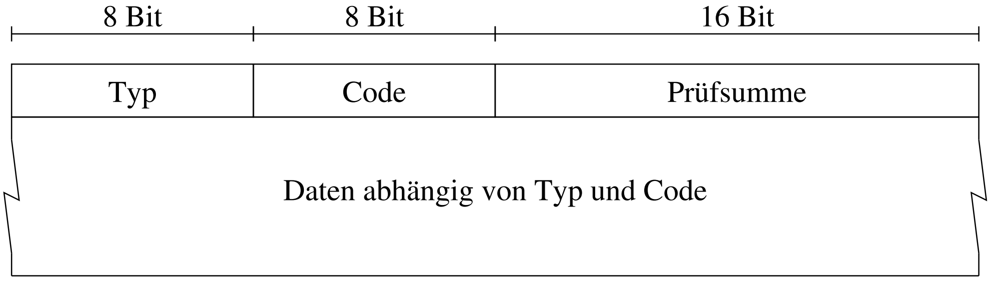

Abbildung 14: Aufbau des ICMP-Headers

Zum Testen der Erreichbarkeit eines Systems auf IP/ICMP-Ebene dient ein Echo-Request (Kommando `ping`, Type 8, Code 0). Im Normalfall antwortet das angesprochene System darauf mit einem Echo-Reply (Type 0, Code 0).

Über die Meldung Destination-Unreachable (Type 3, Code 0 bis 15) kann ein System dem Absender signalisieren, daß das angesprochene Ziel nicht erreichbar ist.
Das Time to Live-Feld im IP-Header (TTL, siehe Abbilung 5) gibt die maximal erlaubte Anzahl von Routern an, welche ein Paket noch passieren darf. Jeder Router verkleinert beim Weiterleiten des Paketes diese TTL um 1. Pakete mit einer TTL gleich Null werden verworfen, an den Absender wird eine ICMP-Time-Exceeded-Meldung (ICMP-Type 11, ICMP-Code 0 oder 1) gesendet.
Die letzten beiden Mechanismen verwendet das Unix-Kommando `traceroute` zur Verfolgung der Route zu einem Zielsystem. Durch Absenden von UDP-Testpaketen[8] zum Ziel, in der Regel auf Ports größer 33000, und schrittweisen Erhöhen der Time to Live wird von jedem Router auf dem Weg zum Zielsystem eine ICMP-Time-Exceeded-Meldung zum Quellrechner geschickt. Ist das Zielsystem erreicht, antwortet dies auf die UDP-Anfrage mit einer ICMP-Destination-Unreachable/Port-Unreachable-Meldung (Type 3, Code 2).

Erkennt ein Router, daß das Zielsystem über einen anderen Weg besser erreichbar ist, kann er über eine Route-Redirect-Meldung (Type 5, Code 0 bis 3) den Absender auffordern, die Verbindung über diesen Weg laufen zu lassen.

Weitere ICMP-Meldungen dienen der Verkehrsfluß-Steuerung und der Abfrage einiger Netzwerkparameter.

[8]Das `tracert`-Kommando unter Windows verwendet ICMP-Echo-Request-Testpakete.

1.12 Praktische Aufgaben

1.12.1 Der Versuchsaufbau

Abbildung 15 zeigt das Netz, welches nach Abschluß der Übungen dieses Kapitels entstehen soll.

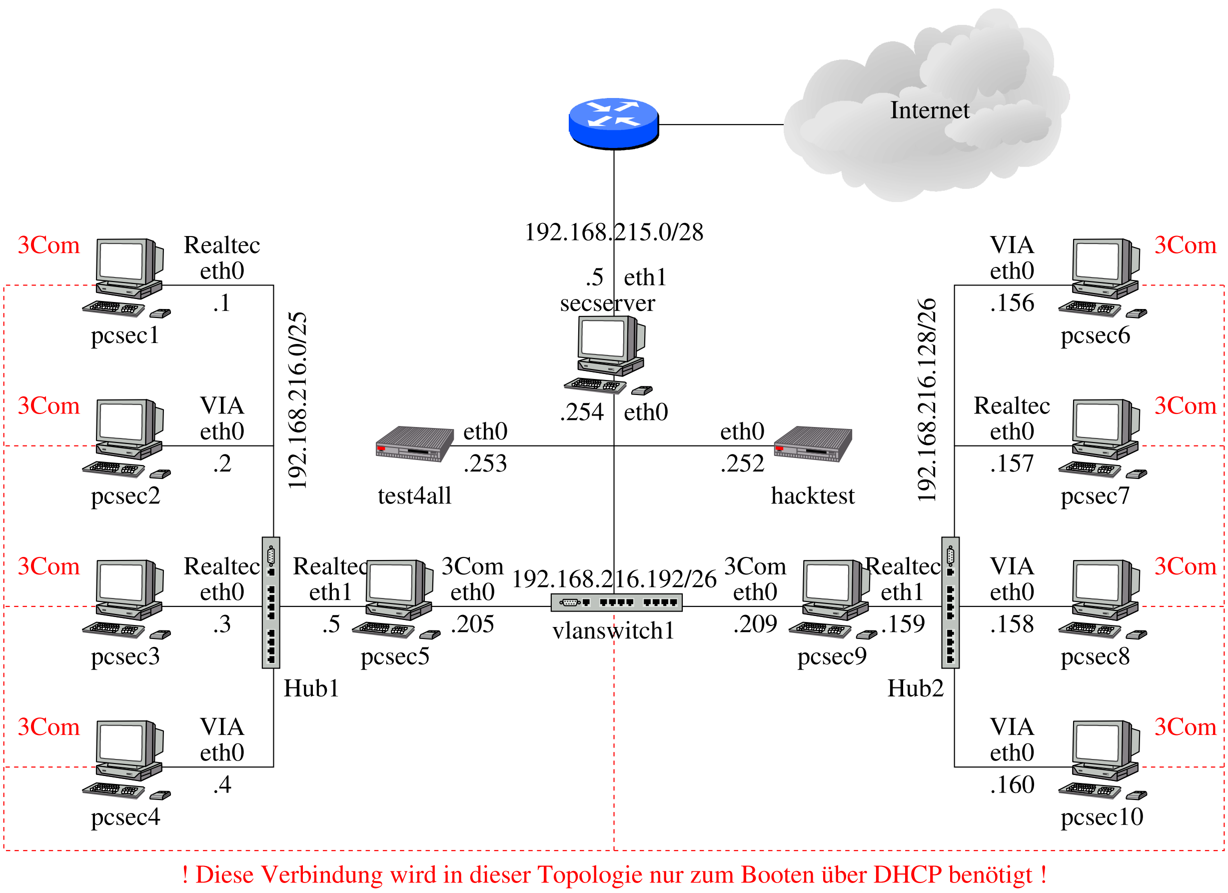

Abbildung 15: Der Versuchsaufbau

Der Rechner `test4all` kann für Tests der Konfigurationen verwendet werden (Benutzer `secpgast`, Passwort `pcsec`).

Sobald der Netzzugang zum `secserver` funktioniert haben Sie über NFS[9] Zugriff auf die SuSE 8.0 CDs unter `/opt/SuSE8.0-CDs`. Im `YaST` tragen Sie zur Installation neuer Software die IP-Adresse des `secserver` sowie den Pfad `/opt/SuSE8.0-CDs/CD1` als Installationsquelle ein.

[9]Network File System zum Einhängen von entfernten Dateisystemen am lokalen Rechner über das Netzwerk.

Des Weiteren können Sie den `secserver` als Internet-Proxy verwenden. Tragen Sie dazu die IP-Adresse der `secserver` und den Port 3128 in Ihrem Browser als HTTP/HTTPS/FTP-Proxy ein.

1.12.2 IP-Adressen und Netzmasken

Bestimmen Sie für die Netze des Versuchsaufbaues aus Abbildung 15 die in den Netzen verwendbaren IP-Adressen und die Broadcast-Adresse.

Welche kleinst mögliche Netzadress/Netzmasken-Kombination beinhaltet alle IP-Adressen der drei internen Netze?

1.12.3 Konfiguration der Netzwerkkarten

1. Lassen Sie sich die ARP- (`man arp`, vgl. Abbildung 4) und die Routing-Tabelle (`man netstat`, vgl. Abbildung 7) sowie die Liste aller konfigurierten Interfaces (`man ifconfig`) Ihres Rechners anzeigen.
2. Nennen Sie ein Beispiel für die praktische Verwendung des Loopback-Interfaces.
3. Konfigurieren Sie nun die Netzwerkkarte(n) Ihres Rechners. Die Rechner sind mit zwei bzw. drei unterschiedlichen Karten ausgerüstet. Welche der Karten Sie wie konfigurieren müssen entnehmen Sie bitte der Abbildung 15. Die dort nicht aufgeführten Karten werden in dieser Topologie nicht benötigt. Die rot eingezeichnete Verbindung dient nur zum Booten und ist für die Versuche nicht weiter von Bedeutung.
   Versuchen Sie, andere Rechner im Netz zu erreichen. Welche Rechner antworten, welche nicht? Welche Meldung erhalten Sie, wenn sie versuchen, einen Rechner außerhalb Ihres Subnetzes zu erreichen?
4. Wie haben sich ARP- und Routing-Tabelle verändert?

1.12.4 Konfiguration der statischen Routen

1. Konfigurieren Sie nun die statischen Routen (mit `YaST`) des Rechners so, daß sie alle Rechner im Netz erreichen können. Arbeiten Sie dabei nicht mit Host-Routen für die einzelnen Rechner sondern mit Netzrouten für die Netze aus Abbildung 15 und verwenden Sie keine Default-Route. Lassen sich Routen zusammenfassen?
2. Auf den Rechnern, die als Router arbeiten sollen, muß auch das Routing aktiviert werden[10] (`YaST`).

[10]Einige Firewall-Produkte erledigen das Routing unabhängig vom Betriebssystem. Das Routing des Betriebssystems darf in so einem Fall nicht aktiv sein um auszuschließen, daß bei nicht laufender Firewall-Software Pakete ungefiltert weitergeleitet werden.

3. Überprüfen Sie nochmal ARP- und Routingtabelle. Welche Änderungen stellen Sie fest?

1.12.5 Überwachung des Netzwerkverkehrs

Für die folgenden Versuche müssen auf Ihren Rechnern noch der FTP- und Telnet-Server-Prozess installiert werden. Installieren Sie dazu mit `YaST` die Pakete `ftpd` und `telnet-server` und aktivieren Sie die beiden Dienste ggf. in der Datei `/etc/inetd.conf`.

1. Starten Sie nun in einem weiteren Terminal-Fenster das Programm `tcpdump` oder `ngrep`. Sinnvolle Optionen entnehmen Sie bitte den Man-Pages.

2. Loggen Sie sich mit ftp auf einem benachbarten Rechner ein und laden Sie einige der dort gespeicherten Dateien herunter (User: `secpgast` Passwort: `pcsec`). Achten Sie dabei darauf, sich keine lokalen Dateien zu überschreiben.
   Was sehen Sie mit `tcpdump`/`ngrep` bei aktiver und nicht aktiver ftp-Verbindung? Versuchen Sie, das ftp-Passwort aufzuzeichnen.

3. Was können Sie daraus bzgl. der Schicht 2-Infrastruktur in Ihrem Netzsegment schließen?

4. Zeichnen Sie den TCP-Verbindungsauf- und -abbau zu einem beliebigen über IP erreichbaren Rechner im Netz (`<Ziel-IP>` auf und kennzeichnen Sie die mitgeschnittenen Pakete als zugehörig zu

   - Verbindungsaufbau
   - Datenübertragung
   - Verbindungsabbau.

   Starten Sie dazu in einem Terminal-Fenster zuerst das Kommando

   ```
   tcpdump -n host <Ziel-IP>
   ```

   Über folgende Kommandos (Beispiel in Abbilung 16) können Sie nun in einem weiteren Terminal-Fenster eine TCP-Verbindug (`telnet`, Port 23) zur (`<Ziel-IP>`) sauber auf- und wieder abbauen.

   `telnet <Ziel-IP>`
   `Control-5` oder `Control-AltGr-9`
   `quit`

   `tcpdump` können Sie mit `Control-C` abbrechen.

```
linux:~#  telnet 10.11.12.13
Trying 10.11.12.13...
Connected to linuxserver.
Escape character is '^]'.
Welcome to SuSE Linux 7.0 (i386) - Kernel 2.2.16 (0).

linuxserver login:
telnet> quit
Connection closed.
linux:~#
```

Abbildung 16: TCP-Verbindungsauf- und Abbau