

FIREWALLS UND SICHERHEIT IM INTERNET

1. EINFÜHRUNG

1.1 UNIX

Der größte Teil der am Internet teilnehmenden Rechner sind UNIX Plattformen. Daher sind für das Verständnis des Internet und insbesondere seiner Firewalls gewisse UNIX Kenntnisse unabdingbar.

1.1.1 Übersicht

root #	Supervisor mit Administrationsrechten
sh	UNIX Kommando Shell
uid	User ID - "s" Bit
/etc/passwd	Passwortdatei für alle Useraccounts
.rhosts	Privilegiert andere Netzwerkrechner
die r-Befehle	remote: rsh, rcp, rlogin, rexec

1.1.2 Komplexität der UNIX Landschaft

Viele der frühen Kommunikationsprogramme wurden ohne großes Sicherheitsbewußtsein geschrieben. Das Internet war neu, die Autoren der Programm waren nicht primär an Sicherheit interessiert, und es war erfreulich, daß die Programme überhaupt existieren. Sie wurden unkritisch in viele kommerzielle Systeme übernommen und haben seither eine Menge Ärger verursacht. Nur sehr wenige Hersteller haben die Grundentwurfsprinzipien überdacht.

Traditionell geht man auch heute noch teilweise von einer wohlgesonnenen Benutzergemeinschaft aus. Die Maschinen sind werkseitig mit sehr freizügigen Rechten ausgestattet.

1.2 Grundsätzliche Gefahren

1.2.1 Netzwerkkommunikation

In Computernetzen wird Information von einem Computer zum nächsten gereicht. Auf jedem dieser Computer liegt die Information daher mindestens einmal kurz auf der Festplatte oder im Speicher, und wird von mindestens einem zentralen Programm weitergereicht.

Das ist eine der Stellen, wo in Netzwerkkommunikation eingegriffen und Information abgefangen werden kann.

1.2.2 Schutz durch Passworte

Der einfachste Weg in ein System ist in der Regel die Vordertür, soll heißen, der *login*-Befehl. Bei fast allen Systemen erfordert eine erfolgreiche Anmeldung, binnen einer angemessenen Zahl von Versuchen das richtige Passwort anzugeben.

Frühe System speicherten die Passworte im Klartext in einer Datei. Die Sicherheit eines solchen Systems beruhte auf der Geheimhaltung des Namens dieser Paßwortdatei: Sie war für jeden lesbar, der ihren Namen kannte. Der "Schutz" bestand darin, daß das Kommando für Inhaltsverzeichnisse diesen Namen unterdrückte. Systemaufrufe lieferten allerdings den Dateinamen.

Mache frühe Login-Programme erlaubten unbegrenzt viele Fehlversuche, was Trial-and-Error Attacken Tür und Tor öffnete. Nichts wurde protokolliert.

Später würde die Paßwortdatei kryptographisch verschlüsselt, blieb aber dennoch für jeden lesbar. Dies führte zu Angriffen über Paßwortlisten ("dictionary attack"). Diese kryptographischen Analysen anhand von mehreren */etc/passwd*, benötigten zwar mehr CPU Zeit, ließen sich aber auch tausende Kilometer entfernt durchführen. Die *shadow*-Paßwortdatei versucht heir, Abhilfe zu schaffen, aber sie ist nicht auf allen, bzw. auf manchen UNIX-Systemen schlecht implementiert.

Wenn keine weiteren Schutzmaßnahmen getroffen werden, ist das Abhören von Paßworten schlichtweg trivial.

1.2.3 Social Engineering

Zum *Social Engineering* gehört meist ein Telefon:

“Ken Thomson. Guten Tag. Jemand hat mich wegen eines Problems mit dem *ls*-Befehl und gebeten, es zu beheben.”

“Oh, OK. Was soll ich tun?”

“Ändern Sie bloß mein Passwort auf ihrer Maschine, es ist eine Weile her, daß ich den Login benutzt habe.”

“Kein Problem.”

1.2.4 Fehler und Hintertürchen

Einer der Ausbreitungswege des Internet Worm war die Übermittlung neuen Programmcodes an einen Demon. Natürlich wartet der Dämon nicht auf Codemodifikationen übers Netz, und im Protokoll gibt es auch gar keine Vorkehrungen hierzu. Der Dämon enthielt einen *gets*-Aufruf ohne Angabe der maximalen Pufferlänge; der Internet Worm schrieb den gewünschten Code über das Ende des Lesepuffers hinaus, bis er auch die Rücksprungadresse im Stackframe von *gets* modifiziert hatte. Ist der erste Dienst einmal gekapert, fallen sämtliche anderen Dienste und schließlich das komplette System wie Dominaosteine hinterher.

Auch wenn dieses spezielle Loch inzwischen längst von den meisten Anbietern gestopft wurde, bleibt doch das grundsätzliche Problem bestehen: Das Schreiben korrekter Software scheint ein für die Informatik unlösbares Problem darzustellen. Es wimmelt vor Fehlern. Desto komplexer die Software wird, desto mehr Fehler und Hintertürchen sind auch enthalten.

1.2.5 Denial-of-Service

Nicht immer dient ein Angriff dem Erlangen neuer Informationen. Manche Leute stehen darauf, Autoreifen aufzuschlitzen oder Wände zu verunstalten. Vandalismus ist ein uraltes Phänomen. Die primitivste und einfachste Form im high-tech Zeitalter des Internets ist es, fremde Festplatten zum Überlaufen zu bringen, indem mittels e-mail mehrer hundert MB übermittelt werden. Zusätzlich zur Verschwendung des Plattenplatzes bedeutet dies, eine allgemeine Lähmung der Maschine durch eine Vielzahl von empfangenden Prozessen.

Auf der Festplatte sollten unbedingt verschiedene Partitionen für den Empfang und für zB die wertvollen Protokolldateien angelegt werden.

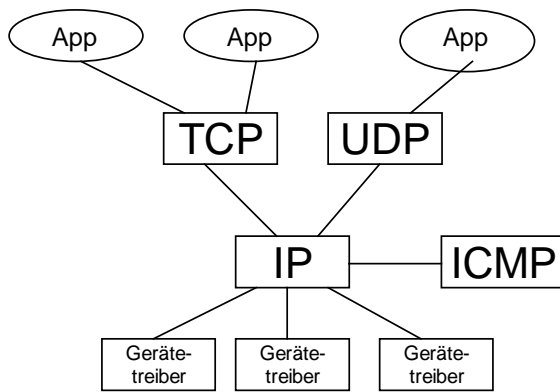
1.3 Wiederholung IP

IP-Pakete sind das Fundament der TCP/IP Protokollfamilie. Jedes Paket ist zusammengesetzt aus einem Kopf, dem Header (bestehend aus der jeweils 32 Bit breiten Quell- und Zieladresse, einigen Optionsbits und einer Prüfsumme), und dem Rumpf mit den Nutzdaten. Ein typisches IP-Paket umfaßt einige hundert Byte. Milliarden solcher Pakete sind rund um die Welt auf Ethernet- oder seriellen Leitungen, FDDI-Ringen, “Packet Radio” oder ATM Verbindungen unterwegs.

Da IP nicht verbindungsorientiert ist, gibt es kein Konzept einer virtuellen Verbindung in der IP-Schicht: Jedes Paket ist eine Übertragungseinheit für sich. IP ist ein ungesichertes Datagramdienst. Das heißt, Pakete können verloren gehen, sie können mehrfach zugestellt werden, sie können einander überholen. Auch die Integrität der Nutzdaten wird nicht geprüft. Die Prüfsumme im IP-Header sichert nur den Paketkopf.

Tatsächlich ist selbst die Korrektheit der Quelladresse nicht garantiert. Theoretisch kann jeder Host ein Paket mit einer beliebigen Quelladresse versenden.

Lange Strecken legen Pakete in vielen Etappen, sogenannten Hops zurück. Jede Etappe endet in einem Vermittlungscomputer oder Router, der das Paket aufgrund der Wegwahlinformationen an den nächsten Hop weiterleitet. Ein Router kann auch Pakete wegwerfen, wenn er überlastet ist. Die Pakete können mehrfach ankommen oder in anderer Reihenfolge als sie gesendet wurden. All dies geschieht in der Regel stillschweigend: Es wird davon ausgegangen, daß höhere Protokollschichten (dh TCP) den Applikationen eine gesicherte Verbindung bereitstellen, indem sie solche Probleme erkennen und behandeln.



Referat von Gerri Kropitz

Layer:

- Application
- Transport
- Network
- Link

Meistens verwendet man nicht die tatsächlichen IP-Adressen sondern einen symbolischen Namen. Die Umsetzung von Namen in Adressen erfolgt in der Regel durch eine spezielle verteilte Datenbank, das *Domain Name System*.

1.3.1 Authentifikation im Netz

Addressbasierte Authentifikation

Es scheint einfach eine IP-Verbindung zur Quelle zurückzuverfolgen, immerhin trägt jedes Paket die IP-Adresse des Absenders. Was könnte leichter sein?

Eine Menge ...

Der sendende Computer kann jede beliebige IP-Quelladresse eintragen. Die meisten Betriebssysteme untersagen unprivilegierten Benutzern eine solche Operation, nicht so jedoch PCs. Dieses Problem ist offensichtlich und bekannt, und stellt die Grundlage einiger schwieriger und obskurer Angriffe dar.

Angriffe beruhen auf der Fälschung der IP-Quelladresse.

Namensbasierte Authentifikation

Namensbasierte Authentifikation ist noch schwächer. Hier muß nicht nur die Adresse, sondern auch der damit verknüpfte Name korrekt sein. Dies bietet dem Angreifer eine weitere Einfallsroute: Unterwandern des Mechanismus, der IP-Adressen in Host-Namen umsetzt. Die Angriffe auf DNS versuchen diesen Weg.

2. PROTOKOLLE UND DIENSTE

2.1 Sicherheitsperspektive von IP

2.1.1 ARP - Address Resolution Protocol

Im allgemeinen werden IP-Pakete über ein Ethernet verschickt. Die Ethernet Geräte verstehen allerdings die 32 Bit breiten IP-Adressen nicht: Sie übertragen Ethernet-Pakete mit 48 Bit breiten Ethernet-Adressen. Daher müssen die IP-Treiber die IP Zieladressen mittels einer Tabelle in Ethernet-Zieladressen umsetzen. Das ARP liefert derartige Zuordnungen.

Dazu sendet ARP einen Ethernet-Broadcast, der die gewünschte IP-Adresse enthält. Der Host mit dieser Adresse oder ein stellvertretendes System antwortet mit einem Paket, welches das IP-Ethernet-Adresspaar enthält. Dieses merkt sich das fragende System, um unnötige ARP-Anfragen zu vermeiden.



Das Verfahren ist nur solange sicher, solange ausschließlich vertrauenswürdige Maschinen auf dem lokalen Datennetz senden können. Es ist nämlich möglich, daß eine Maschine getürkte ARP-Antworten gibt und auf diese Weise allen Datenverkehr auf sich selbst umlenkt. Dann kann sie sich sowohl als andere Hosts ausgeben als auch Datenströme en passant modifizieren.

Gewöhnlich schaltet man das automatisch ARP aus, und verwendet für sein Netzwerk fixierte Tabellen, um solche Attacken zu vereiteln.

2.1.2 TCP - Transmission Control Protocol

TCP stellt gesicherte virtuelle Verbindungen bereit. Verlorene oder verstümmelte Pakete werden nochmal übertragen und die Pakete in der gleichen Reihenfolge abgeliefert, in der sie gesendet wurden.

Abbildung Seite 26

Die Reihenfolge der Pakete wird durch die Laufnummer bestimmt. Jedes übermittelte Byte wird gezählt. Alle TCP-Pakete, außer dem allerersten einer Sitzung, enthalten eine Quittungsnummer, welche die Laufnummer des letzten in Folge korrekt empfangenen Byte zurückgibt. Die Startlaufnummer (initial sequence number) wird zeitabhängig zufällig bestimmt. Das sich die Startlaufnummer für neue Verbindungen ständig ändert, ist es TCP möglich, alte Pakete aus vorangegangenen Inkarnationen derselben virtuellen Verbindung zu erkennen.

Jede TCP-Nachricht enthält den 4-Tupel

<Quellsystem, Quellport, Zielsystem, Zielport>

Durch diese Kombination aus Quell- und Zielsystem und jeweiligen Port-Nummern wird sie eindeutig einer bestimmten virtuellen Verbindung zugeordnet.

Es ist nicht nur erlaubt, sondern durchaus üblich, mehrere verschiedene Verbindungen über die gleiche lokale Port-Nummer abzuwickeln. Solange sich Zielsystem oder Zielport dieser Verbindungen unterscheiden, gibt es keine Probleme.

Server-Prozesse, die einen Dienst über TCP anbieten, lauschen auf bestimmten Port-Nummern. Dies wird TCP-Listen genannt. Per stiller Übereinkunft haben die Server-Ports niedrige Nummern. Diese Übereinkunft wird allerdings nicht immer eingehalten, was zu Sicherheitsproblemen führen kann.

Die Port-Nummern der Standarddienste werden als bekannt vorausgesetzt. Ein Port im Listen-Modus stellt im gewissen Sinn eine halboffene Verbindung dar: Nur Quellsystem und Quellport sind bekannt. Geht ein Paket mit einer Verbindungsanfrage ein, so werden die fehlenden Einträge ergänzt. Der Server-Prozeß kann vom Betriebssystem dupliziert werden, so daß weitere Anfragen auf den selben Port auch behandelt werden können.

Port	Dienst
25	smtp
80	http
119	nntp

Die meisten TCP Versionen für UNIX Systeme stellen sicher, daß nur der Systemverwalter (root) Port-Nummern unterhalb von 1024 nutzen kann. Dies sind die *privilegierten Ports*. Fremde System sollen der Authentizität von Informationen, die sie von diesen Ports erhalten, vertrauen können. Diese Einschränkung ist allerdings nur eine Konvention, deren Einhaltung von der Protokollspezifikation nicht verlangt wird. Die Konsequenz ist klar: Sie können privilegierten Ports nur dann trauen, wenn sie absolut sicher sind, daß das Quellsystem die Konvention einhält und korrekt administriert wird.

Die schon erwähnten Laufnummern haben auch einen gewissen Sicherheitseffekt: Eine Verbindung kommt erst dann zustande, wenn beide Seiten jeweils den Empfang der Startlaufnummern quittiert haben.

☛ Da lauert aber auch die Gefahr: Wenn ein Angreifer die Auswahl der Startlaufnummern bei seinem Opfer vorhersagen kann - Erfahrungen haben gezeigt, daß dies unter bestimmten Bedingungen tatsächlich möglich ist -, dann kann er seinem Opfer eine Verbindung mit einer vertrauenswürdigen Maschine vortäuschen.

2.1.3 UDP - User Datagram Protocol

UDP stellt Applikationen die Datagram-Dienste von IP direkt zur Verfügung. Die Paketzustellung erfolgt ungesichert: verlorene, duplizierte oder in der Reihenfolge vertauschte Pakete werden nicht erkannt, selbst die Erkennung von Übertragungsfehlern ist optional und eine Fehlerkorrektur nicht vorhanden.

UDP tendiert zu ungünstigem Verhalten, wenn es für umfangreiche Übertragungen benutzt wird. Da dem Protokoll eine Flußkontrolle fehlt, kann es das Datennetz lahmlegen, indem es Router und Hosts mit Paketen überflutet. Durch diese Überflutungen steigen die Paketverluste im Netz drastisch.

☛ UDP-Pakete sind viel leichter zu fälschen als TCP-Pakete, weil es weder Quittungs- noch Laufnummern gibt. Es ist daher äußerste Vorsicht geboten, wenn man die Quelladressen solcher Pakete verwendet. Die Applikationen selbst müssen geeignete Sicherheitsvorkehrungen treffen.

2.1.4 ICMP - Internet Control Message Protocol

Mit ICMP läßt sich das Verhalten von TCP- und UDP-Verbindungen beeinflussen. Es dient dazu, Hosts günstigere Routen zu einem Ziel bekanntzugeben, über Routing-Probleme zu informieren oder Verbindungen wegen Problemen im Datennetz abubrechen.

☛ Viele ICMP Nachrichten, die einen Host erreichen, sind nur für eine bestimmte Verbindung relevant oder durch ein bestimmtes Paket ausgelöst. So sollte eine Redirect- oder Destination Unreachable-Nachricht sich bloß auf eine bestimmte Verbindung beziehen. Unglücklicherweise nutzen alte ICMP-Implementierungen diese zusätzliche Information nicht. Wenn solche Nachrichten empfangen werden, wirken sie auf alle Verbindungen zwischen den beteiligten Hosts. Wenn ihr Hosts Antwort auf ein Paket zum Host PING.CO.AT ein Destination Unreachable erhält, weil dieses ein Paket PING.CO.AT nicht erreichen konnte, dann werden alle Verbindungen zu PING.CO.AT abgebrochen. Manche Hacker finden sogar Gefallen daran, durch Mißbrauch von ICMP Verbindungen zu kappen.

Router sollten niemals einer Redirect Message Glauben schenken, da es dadurch einem Angreifer möglich ist, den Verkehr zu sich selbst umzuleiten.

2.1.5 RIP - Routing Information Protocol

RIP ist ein Protokoll zur Steuerung der Wegwahl im Datennetz.

☛ Es ist recht einfach, falsche Nachrichten für RIP in ein Datennetz einzuschleusen. Befindet sich der Angreifer näher am Ziel als das Quellsystem, so kann er den Datenverkehr leicht umlenken. Folgeversionen von RIP stellen, um dem entgegenzuwirken, ein Authentifikationsfeld zur Verfügung.

2.1.6 DNS - Domain Name System

Das DNS ist ein verteiltes Datenbanksystem, welches (leicht merkbare, wie zB PING.CO.AT) Host-Namen in verwendbare IP-Adressen umsetzt und umgekehrt. Im Normalbetrieb senden Hosts UDP-Anfragen an DNS-Server. Dieses antworten entweder mit der richtigen Antwort oder verweisen auf besser informierte Server.

Im DNS wird eine Reihe verschiedener Datensätze gespeichert:

Typ	Funktion
A	Adresse eines bestimmten Hosts
NS	Name-Server. Verweist auf den für den Teilbaum zuständigen Server
SOA	Start of authority. Markiert den Anfang eines Teilbaumes, enthält neben Caching und Konfigurationsparametern auch die Adresse der für diese Zone verantwortlichen Person.
MX	Mail Exchange. Name des Systems, welches die eingehende Post für das angegebene Ziel annimmt. Bei der Zielangabe können Jokerzeichen, wie etwa *.ATT.COM, verwendet werden, so daß ein einziger MX-Eintrag Post für einen ganzen Teilbaum umlenken kann.
HINFO	Information über Betriebssystem und Maschinentyp eines Hosts.
CNAME	Alternative Namen für einen Host.
PTR	Umsetzung von IP-Adressen in Host-Namen

Der Namesadreibraum von DNS ist baumförmig. Um den Betrieb zu erleichtern, können ganze Teilbäume, sogenannte Zonen, an andere Server delegiert werden. Es werden zwei logisch getrennte Bäume verwendet. Der eine setzt Systemnamen wie etwa PING.CO.AT auf IP-Adressen wie 192.20.225.3 um. Es können weitere Informationen über den Host vorhanden sein, beispielweise HINFO- oder MX-Einträge. Der andere Baum enthält PTR-Datensätze und beantwortet *inverse queries*, also Anfragen in die Gegenrichtung, bei denen auf eine IP-Adresse mit dem Namen geantwortet wird. Zwischen beiden Bäumen wird keine Konsistenz garantiert.

☛ Dieser Mangel an Konsistenz kann Probleme bereiten. Erlangt ein Hacker Kontrolle über den inversen DNS-Baum, kann er ihn für seine Zwecke mißbrauchen. Enthält der inverse Eintrag mit der Adresse des Angreifersystems den Namen eines Hosts, dem das System vertraut (.rhosts), so wird es einem rlogin-Versuch des Angreifers stattgeben, weil es dem gefälschten Eintrag Glauben schenkt.

Die meisten neueren Systeme sind gegen diesen Angriff immun. Nachdem sie von DNS den mutmaßlichen Host-Namen erhalten haben, prüfen sie im Gegenzug die diesem Namen zugeordneten IP-Adressen. Ist die Quelladresse der Verbindung nicht dabei, platzt der Verbindungsversuch und wird als sicherheitsrelevant protokolliert.

Eine weitere Gefahr liegt in der Eigenschaft vieler Implementierungen von DNS-Resolvern, fehlende Teile eines Namens aus dem eigenen Namen versuchsweise abzuleiten.

Beispielsweise versucht FOO.DEPT.BIG.EDU das Ziel BAR.COM anzusprechen. Der DNS-Resolver wird Teile des eigenen Namens ergänzen, um Benutzern abkürzende Schreibweisen zu erlauben, und erst

- BAR.COM.DEPT.BIG.EDU,
- BAR.COM.BIG.EDU
- BAR.COM.EDU

probieren, ehe er (korrekt) BAR.COM verwendet. Darin liegt die Gefahr: Erzeugt jemand eine Domain COM.EDU, könnte er allen Datenverkehr für .COM abfangen.

2.2 Sicherheitsperspektive der Standard-Dienst

2.2.1 SMTP - Simple Mail Transport Protocol

SMTP ist das Standard Protokoll für den Transport von e-mail im Internet. SMTP ist ein einfaches, geheimnisumwittertes Protokoll zum Transport von 7-Bit-Zeichen des ASCII-Zeichensatzes.

←	220 INET.IBM.COM SMTP
⇒	HELO NT.MICKEYSOFT.COM
←	250 INET.IBM.COM
⇒	MAIL FROM:<Bill.Gates@NT.MICKEYSOFT.COM>
←	250 OK
⇒	RCPT TO:<Lou Gerstner@OS2.IBM.COM>
←	250 OK
⇒	DATA
←	354 Start mail input; end with <CRLF>.<CRLF>
⇒	Bla Bla Bla
⇒	Bla Bla
⇒	Bla
⇒	Billy Boy
⇒	.
←	250 OK
⇒	QUIT
←	221 INET.IBM.COM Terminating

Die fremde Anlage NT.MICKEYSOFT.COM sendet Post an die lokale Maschine INET.IBM.COM. Das Protokoll ist sichtbar einfach. Postmaster und Hacker kennen diese Befehle und geben sie gelegentlich auch selber ein.

☛ Das fremde System gibt im "MAIL FROM"-Befehl eine Absenderadresse an. Das lokale System hat in diesem Rahmen keine zuverlässige Möglichkeit, diese Adresse zu überprüfen. Falls Authentizität oder Vertraulichkeit notwendig ist, so ist diese auf höheren Protokollebenen zu implementieren.

Vom Sicherheitsstandpunkt aus ist SMTP an und für sich recht harmlos. Es kann aber - wie bereits erwähnt - das Einfallstor für Denial-of-Service-Angriffe sein.

Häufig werden Mail-Prozessoren auf einer Gatewaymaschine gefahren. Hier ist auch der ideale Platz um unternehmensweite Mail-Aliase für die Mitarbeiter einzurichten.

In *sendmail* findet sich die verbreitetste SMTP-Implementierung. Auch wenn *sendmail* mit den meisten UNIX-Systemen ausgeliefert wird, ist es das Geld nicht wert: es ist ein Sicherheitsalptraum. Das Programm besteht aus zehntausenden von Zeilen C-Code und läuft häufig unter *root*. Eine der Sicherheitslücken, die der Internet Worm ausnutzte fand sich in *sendmail* und war der *New York Times* eine Meldung wert. Privilegierte Programme sollten so klein und modular wie möglich sein. Ein SMTP-Dämon benötigt keine *root* Privilegien. Da es auf dem Gateway läuft, benötigt es Schreibrechte auf ein Spool-Verzeichnis, Leserechte auf */dev/kmem*, um die aktuelle Last (load average) des Systems zu bestimmen, sowie die Möglichkeit, den Port 25 zu belegen.

☛ Auch der Inhalt der Post kann gefährlich sein. Abgesehen von möglichen Fehlern im empfangenden Mailer, ist die vertrauensselige Ausführung von Nachrichten, die nach *Multipurpose Internet Mail Extensions (MIME)* kodiert sind, selbst eine Gefahrenquelle. Diese können nämlich Angaben enthalten, die den Mailer zu Aktionen veranlassen.

Beispiel

```
Content-Type: Message/External-body;
  name="angebot1.txt";
  site="PING.CO.AT";
  access-type="anon-ftp";
  directory="angebote"
```

Content-Type: text/plain

Ein MIME-fähiger Mailer würde "angebot1.txt" automatisch beschaffen.

```
Content-Type: Message/External-body;
  name=".rhosts";
  site="PING.CO.AT";
  access-type="anon-ftp";
  directory="."
```

Content-Type: text/plain

Es besteht die Gefahr, daß der MIME-Agent sorglos die vorhandene `.rhosts`-Datei im aktuellen Verzeichnis überschreibt.

2.2.2 Telnet

telnet bietet einen einfachen Terminalzugang zu einem System. Das Protokoll enthält Steuerungsmöglichkeiten für verschiedenen Terminaleinstellungen. In der Regel rufen *telnet*-Dämonen zum Authentifizieren und Initialisieren einer Sitzung *login* auf, welches Benutzernamen und meist auch Paßwort enthält.

☛ Die meisten *telnet*-Sitzungen werden auf Maschinen gestartet, zu denen kein Vertrauensverhältnis besteht. Weder dem sendenden Programm, oder Betriebssystem, noch den dazwischenliegenden Datennetzen kann getraut werden. *Paßwort und Inhalt der Sitzung sind neugierigen Augen preisgegeben*. Paket-Sniffer an strategisch günstigen Knotenpunkten (zB Backbones) haben in der Geschichte zigtausende Paßwörter gesammelt.

Wenn sich die beiden Endpunkte trauen, kann man Kryptographie für die komplette Sitzung verwenden.

2.2.3 NTP - Network Time Protocol

☛ NTP synchronisiert, wie der Name schon sagt, die Systemuhren untereinander. Es ist kein demokratisches Protokoll, sondern glaubt an die Idee einer absolut korrekten Zeit, die dem Datennetz durch Maschinen mit Atomuhren offenbart wird.

NTP kann das Ziel verschiedener Angriffe sein. Im allgemeinen beabsichtigt eine solche Attacke, dem Ziel eine falsche Uhrzeit vorzutauschen. Dies ist insbesondere dann problematisch, wenn das Opfer ein Authentifikationsverfahren, das auf Zeitstempeln basiert, einsetzt. Wenn jemand die Systemuhr zurückdrehen kann, kann er alte Authentizitätsnachweise wiederverwenden.

Um sich gegen solche Angriffe zu schützen, unterstützen neuere NTP-Versionen kryptographische Authentifikation der ausgetauschten Nachrichten.

2.3 RPC-basierte Protokolle

2.3.1 RPC und Protmapper

Das *Remote Procedure Call (RPC)*-Protokoll von Sun ist die Grundlage vieler neuerer Dienste. Unglücklicherweise stellen viele dieser Dienste ein potentiell Sicherheitsrisiko dar.

Der Entwickler eines neuen Dienstes definiert in einer speziellen Sprache die Namen und Parameter der externen Funktionsaufrufe. Ein Präcompiler übersetzt diese Spezifikation in sogenannte *stub-Routinen* für die Clienten- und Servermodule. Diese "stubs" ragen gleichsam auf Clienten- und Server-Seite heraus und "kleben" beide Seiten über das Datennetz zusammen. Der Client ruft seinen "stub" wie ein normales Unterprogramm auf und

aktiviert damit auf dem Server den zugehörigen Aufruf. Die meisten Probleme verteilter Programmierung werden durch RPC-Abstraktion umgangen.

RPC kann sowohl auf TCP als auch auf UDP aufsetzen. Die meisten grundlegenden Charakteristika des Transportmechanismus scheinen durch. Ein Subsystem, welches RPC über UDP nutzt, muß sich also mit verlorenen, duplizierten oder umsortierten Nachrichten herumschlagen.

RPC-Nachrichten haben einen eigenen Header. Er enthält die *Programmnummer*, die *Procedurnummer*, welche den gewünschten Aufruf kennzeichnet und einige Versionsnummern. Weiterhin enthält der Header eine Laufnummer, die die Zuordnung von Antworten zu Anfragen erlaubt.

☛ Es gibt auch ein Authentifikationsfeld. Es enthält die sogenannte UNIX-Authifikation, bestehend aus der Benutzer- und Gruppennummer des rufenden Prozesses und den Namen des Quellsystems. Große Vorsicht ist hier angebracht. Dem Quellsystemnamen sollte man nie trauen. Auch die Benutzer- und Gruppennummern sind nicht sehr viel wert. Auf solchen Nachrichten basierend sollte man niemals sicherheitsrelevante Aktionen einleiten.

Neuere Versionen von RPC (*secure RPC*) unterstützen auch kryptographische Authentifikation mittels DES, dem Data Encryption Standard. Das *Distributed Computing Environment (DCE)* von OSF setzt zur Authentifizierung Kerberos ein.

RPC-basierte Server sind normalerweise nicht an bestimmte Port-Nummern gebunden. Sie akzeptieren die Port-Nummern, die ihnen das Betriebssystem zuteilt, und lassen diese Zuordnung vom *portmapper* registrieren. Der *portmapper* wirkt als Vermittler zwischen RPC-Servern und -Clienten. Um einen Server zu kontaktieren, erkundigt sich der Client erst bei dem *portmapper* der Zielmaschine nach Portnummer und Protokoll (UDP/TCP) des Dienstes. Der eigentliche RPC-Aufruf wird auf Grundlage dieser Information ausgelöst.

Der *portmapper* hat weiter, nicht unbedingt zuträgliche Fähigkeiten. So gibt es beispielsweise einen Aufruf, um einen Dienst abzumelden. Dies ist ein gefundenes Fressen für Denial-of-Service-Angriffe, da der Aufruf unzureichend gesichert ist. Der *portmapper* teilt darüberhinaus jedem im Datennetz freudig mit, welche Dienste angeboten werden. Dies ist eine extrem nützliche Planungshilfe für den Angreifer. (In sichergestellten Hacker-Protokollen haben sich viele solcher *portmapper*-Tabellen gefunden, die der Freigebigkeit des normalen *rxinfo*-Befehls zu verdanken sind.

☛ Das größte Problem des *portmapper* ist aber seine Fähigkeit, indirekte RPC-Aufrufe zu ermöglichen. Um den Aufwand einer Anfrage nach der tatsächlichen Port-Nummer, der Antwort und dem eigentlichen RPC-Aufruf zu vermindern, kann der Client den *portmapper* beauftragen, den RPC-Aufruf an den gewünschten Server weiterzuleiten. Diese weitergeleitete Nachricht trägt aber notwendigerweise die Quelladresse des *portmapper*. Damit wird es den Applikationen unmöglich, diesen Auftrag von echten lokalen Aufträgen zu unterscheiden und dessen Vertrauenswürdigkeit zu beurteilen.

Es ist darauf zu achten, nur gesicherte *portmapper* Versionen zu verwenden.

2.3.2 NIS - Network Information Service

Eine der gefährlichsten RPC-Applikationen ist der NIS. NIS dient der Verteilung wichtiger, zentraler Konfigurationsdateien von einem Server an seine Clienten. Darunter fallen Paßwortdatei, die Tabelle der Host-Adressen sowie die öffentlichen und privaten Tabellen der Schlüsselverwaltung für *secure RPC*.

☛ Eine Reihe von Risiken ist offensichtlich. Kommt ein Angreifer in Besitz der Paßwortdatei, hat er einen kostbaren Fang gemacht. Die Schlüsseltabellen sind beinahe so gut: Die privaten Schlüssel der Benutzer sind in der Regel mit ihren Paßworten verschlüsselt.

Falls der zentrale Server ausfällt, brauchen NIS-Clienten einen Stellvertreter. Bei einigen Versionen kann man die Clienten - ferngesteuert - anweisen, auf einen anderen, eventuell betrügerischen NIS - Server umzuschalten. Dieser kann dann erschwandelte Einträge für */etc/passwd*, Host-Adressen usw. bereitstellen.

Die gefährlichsten Operationen lassen sich bei manchen NIS-Versionen abschalten

2.3.3 NFS - Network File System

Ursprünglich von Sun entwickelt, wird NFS heute von den meisten Computern unterstützt. NFS basiert auf UDP-RPC. Ein NFS-Server merkt sich keinen Kontext, sondern behandelt jede Anfrage unabhängig. Damit muß auch jede Anfrage eigens authentifiziert werden.

Grundlegend ist das Konzept des *File Handle*, eines eindeutigen Kennzeichen für jede Datei oder jedes Verzeichnis auf einer Festplatte. Alle NFS-Aufträge bestehen aus einem File Handle, der gewünschten Operation und den notwendigen Parametern. Anfragen, die Zugriff auf eine neue Datei, wie etwa `open`, liefern dem Clienten einen neuen File Handle zurück. Die File Handle werden vom Clienten nicht interpretiert. Sie bestehen aus den vom Server benötigten Verwaltungsinformationen und einer zufälligen Komponente.

Wird ein Dateisystem gemountet, so erhält man das Handle für sein Wurzelverzeichnis. Der Mount-Dämon des Servers, ein RPC-Dienst, prüft den Host-Namen des Clienten, das angeforderte Dateisystem und den gewünschten Zugriffsmodus (read/write - read only) anhand einer Konfigurationsdatei. Zulässige Anfragen werden mit dem File Handle für die Wurzel des Dateisystems beantwortet.

☛ Solange ein Client über ein solches Wurzel-Handle verfügt, hat er permanenten Zugriff auf das Dateisystem. Zwar bitten normale Clienten bei jedem Mount, also meistens beim Systemstart, erneut um Zugriffserlaubnis, aber nichts zwingt sie zu dieser freundlichen Geste. (Der Kernel könnte dies prinzipiell durch eigene Zugriffskontrolllisten durchsetzen. Dies wird aus Effizienzgründen häufig unterlassen.) Die Zugriffskontrolle von NFS beim Einhängen von Dateisystemen erweist sich damit als unzureichend. Weder ist es möglich, die Befugnisse von Clienten, die schon NFS-Zugriff hatte, nachträglich einzuschränken, noch gibt es Schutz gegen Benutzer, die File Handle für Dateisystemwurzeln bekanntgeben.

File Handle werden normalerweise bei der Erzeugung eines Dateisystems mit Hilfe eines Pseudozufallsgenerator bestimmt. (Bei einigen älteren NFS-Versionen war der Startwert unzureichend zufällig - und damit vorhersagbar.)

2.4 Dateitransferprotokolle

2.4.1 TFTP - Trivial File Transport Protocol

TFTP ist ein einfaches UDP-basiertes Dateitransferprotokoll ohne jede Authentifikation. Es wird häufig benutzt, Diskless Workstations zu booten.

Ein richtig TFTP konfigurierter TFTP-Dämon beschränkt den Dateitransfer auf ein bis zwei Verzeichnisse, meistens `/usr/local/boot`. Es gab Zeiten, da lieferten die Hersteller ihre Software mit unbegrenzten TFTP-Zugang aus. Hacken war dann kinderleicht:

```
$ tftp ping.co.at
tftp> get /etc/passwd /tmp/passwd
Received 1205 bytes in 0.5 seconds
tftp> quit
$ crack </tmp/passwd
```

☛ Fast zu einfach, denn bei einer typeischen Erfolgsquote von 25 % bei Raten von Paßworten mittels entsprechender Wörterlisten ist diese Maschine und alle, die ihr vertrauen erledigt. TDTP sollte nur auf Systemen laufen, die es wirklich brauchen. Und dann sollten Sie sicherstellen, daß es korrekt konfiguriert ist und nur die richtigen Daten an die richtigen Clienten ausliefert.

2.4.2 FTP - File Transfer Protocol

FTP unterstützt den Transport von Text- und Binärdateien. Während einer typischen Sitzung baut der gestartete `ftp` Befehl einen *Kontrollkanal* zum Zielsystem auf.

Die eigentlichen Daten, sei es eine Datei oder ein Verzeichnisisinhalt, werden über einen separaten *Datenkanal* transportiert. Üblicherweise wartet der Client auf einer zufälligen Port-Nummer, die er dem Server durch einen `PORT`-Befehl mitteilt, auf die Daten.

Im Startzustand erfolgt die Übertragung im ASCII-Modus. Zum Transfer von Dateien, die nicht aus (systemdefinierten) Zeilen von druckbaren ASCII-Zeichen bestehen, müssen beide Seiten mit `TYPE I` in den Binärmodus (bekannt als *image*- oder *binary*-Modus)

Anonymouse FTP ist der wichtigste Mechanismus zur Verteilung von Programmen und Daten. Wer will, kann seinen FTP-Server so konfigurieren, daß jeder Dateien aus einem festgelegten Systembereich ohne Voranmeldung oder Befugniserteilung kopieren kann. Per Konvention meldet sich der Benutzer dazu als *anonymous*. Manche Betreiber verlangen vom Benutzer als Paßwort die e-mail Adresse, ein Wunsch, welcher meist mit Mißachtung bedacht wird, auch wenn einige FTP-Server diese Regel durchzusetzen versuchen.

```
$ ftp ping.co.at
220      inet FTP server ready.
```

```

=> USER anonymous
331 Guest login ok, send ident as password.
=> PASS guest
230 Guest login ok, access restrictions apply.
=> SYST
215 UNIX Type: L8 Version BSD-43
Remote system type is UNIX.
ftp> ls
=> PORT 192, 20, 225, 3, 5, 163
200 PORT command successful.
=> TYPE A
200 Type set to A.
=> NLST
150 Opening ASCII mode data connection for /bin/ls.
bin
dist
etc
doc.Z
netlib
pub
226 Transfer complete.
=> TYPE I
200 Type set to I.
=> STOR doc.Z
226 Transfer complete.
ftp> bye
=> QUIT
221 Goodbye.
$

```

☛ Die allerwichtigste Regel lautet, keine Datei und kein Verzeichnis im Bereich für anonymous FTP darf der ftp-Kennung (unter der anonymouse FTP ausgeführt) gehören oder von ihr beschreibbar sein. Ein Hacker könnte sonst eine Datei namens `.rhosts` mit geeignetem Inhalt in das Heimatverzeichnis von ftp schreiben. Damit kann man nun mittels `rsh` eine Sitzung als ftp auf der Zielmaschine starten.

☛ Es sollte sich keine echte `/etc/passwd` Datei im Bereich für anonymous FTP befinden. Eine echte `/etc/passwd` wäre ein großes Geschenk für einen Hacker. Wenn die UNIX Implementierung es zuläßt, sollte sie komplett gelöscht werden. Andernfalls sollte sie durch ein Phantasiegebilde ohne echte Kennungen und ohne echte Paßworteinträge ersetzt werden.

Schließlich sollte vorsichtshalber alles im FTP-Bereich mit Mißtrauen betrachtet werden. Dies gilt besonders im Hinblick auf ausführbare Programme, namentlich die Kopie von `ls`, da diese durch ein Trojanisches Pferd ersetzt worden sein könnten.

2.5 Die "r"-Befehle

Die "r"-Befehle verlassen sich auf die Betriebssystem-Authentifikation. Man kann einen `rlogin` auf eine fremde Maschine ohne Paßwortabfrage durchführen, wenn folgende Authentifikationskriterien erfüllt sind:

- Die Anfrage muß von einem privilegierten TCP-Port ausgehen. Auf manchen Systemen (wie PCs) gibt es solche Beschränkungen nicht, und sie wären auch sinnlos. Daraus folgt, daß Anfragen per `rlogin` und `rsh` nur Maschinen gestattet werden sollten, die die Vergabe privilegierter Ports wirksam kontrollieren.
- Die Kombination von Benutzer und System, von der die Anfrage ausgeht, muß dem Zielsystem als vertrauenswürdig erklärt worden sein, entweder Systemweit (`/etc/hosts.equiv`, oder auf Benutzerebene `$HOME/.rhosts`).
- Name und IP-Adresse des Quellsystems müssen übereinstimmen

Aus Benutzersicht funktioniert dieses Verfahren prächtig. Ein Benutzer kann die Maschinen, die er benutzen will, absegnen und wird nicht mehr mit Paßwortabfragen belästigt, wenn er andere Computer erreichen will. Auch aus Hackersicht funktioniert es prächtig: Es bietet ein Einfallstor in ein System und von dort Zugang zu weiteren

Hosts, die dem ersten vertrauen. Ein Hauptziel von Hackern ist, geeignete Einträge in `/etc/hosts.equiv` oder `.rhosts` eines Benutzers zu deponieren.

Wenn Hacker Zugang zu einem System erlangt haben, streben sie zuerst danach, ihre Spuren zu verwischen, indem sie Protokolldaten löschen. Dann versuchen sie, `root`-Zugang zu bekommen und sich Hintertüren anzulegen, falls der Weg durch den Vordereingang blockiert werden sollte.

Ein Gutteil der Sicherheit eines Systems liegt also in der Hand der Benutzer, die fremden Maschinen mit ihren `.rhosts` Dateien das Vertrauen aussprechen können. Solche Entscheidungen sollten der Systemadministration vorbehalten sein. Einige Versionen der `rlogin` und `rsh`-Dämonen können entsprechend konfiguriert werden. Ansonsten ist wohl ein Batchjob, der das System regelmäßig von `.rhosts`-Dateien säubert, angebracht.

Angesichts der vielen Schwächen dieses Authentifikationsverfahrens ist es ratsam, diese Dienste nicht auf Systemen anzubieten, die direkt vom Internet aus zugänglich sind.

2.6 Informationsdienste

2.6.1 WWW - World Wide Web

In letzter Zeit war ein explosives Wachstum dessen zu verzeichnen, was man vielleicht am besten als *Informationsprotokolle* bezeichnet. Darunter fallen *gopher*, *Wide Area Information Server (WAIS)* und andere, die unter dem Oberbegriff *World Wide Web (WWW)* zusammengefaßt werden.

In der Regel kontaktiert ein Host einen Server, sendet eine Anfrage oder eine Referenz auf eine Information und erhält eine Antwort. Diese kann entweder aus einer Datei, die angezeigt werden kann, oder aus einer oder mehrer Referenzen auf andere Server bestehen. Die Anfragen, Dokumente und Referenzen sind allesamt mögliche Gefahrenquellen.

☛ Der Server ist in Gefahr, wenn er Referenzen blinden Glauben schenkt. Solche Referenzen (URLs, unified resource locators) enthalten häufig Dateinamen. Auch wenn der Server prüft, ob die angeforderten Dateien zum Transfer freigegeben sind, so kann es vorkommen, daß diese Prüfung fehlerhaft erfolgt.

☛ Eine der größten Gefahren ergibt sich, wenn der Informationsserver seine Verzeichnisse mit anonymous FTP teilt. In diesem Fall kann der Angreifer zuerst Steuerdateien deponieren und kann dann den Informationsserver veranlassen, sie zu interpretieren. Dies kann vermieden werden, indem *alle* öffentlich beschreibbaren Verzeichnisse im Bereich für anonymous FTP der gleichen Gruppe wie der Informationsserver angehören, wobei die Suchberechtigung für die Gruppe abgeschaltet ist. Damit kann der Informationsserver nicht auf die Inhalte dieser Verzeichnisse zugreifen.

2.6.2 Multicasting

Der *Multicast* ist eine Verallgemeinerung des *Unicast* und des *Rundrufes*. Statt ein Paket nur an eine bestimmte Station oder aber an alle im Datennetz zu verschicken, wird ein Sammelruf an eine Teilmenge beliebiger Zusammensetzung versendet. Ein System kann mehreren Sammelrufgruppen angehören, aber auch keiner.

Die Multicast-Router verkapseln die Sammelrufpaketet inklusive IP-Header in ganz normalen IP-Paketet mit normalen Zieladressen. Auf der Zielmaschine wird das Sammelrufpaket wieder ausgepackt, an andere Multicast-Router weitergeleitet oder lokal im Netz verteilt. Die Endziele sind im allgemeinen UDP-Ports.

Eine Reihe interessanter Datendienste nutzen *Multicast* um ein weites Publikum zu erreichen. Dazu gehören Audio- und Video-Übertragungen und -Konferenzen, NASA-Berichterstattung, Präsidentenansprachen. Ein Dienst namens *Session Dictionary* versorgt mit Informationen über die verfügbaren Kanäle.

☛ Für geschützte Anlagen ist *Multicasting* ein Problem. Die Verkapselung verbirgt das eigentliche Ziel des Pakets. Damit erlaubt *Multicasting* das Umgehen der Kontrolle am Gateway.

Abhilfe könnte geschafft werden, wenn Multicast-Router Sammelrufpakete nur an Ports ausliefern, die diese ausdrücklich angefordert haben.

3. FIREWALL GATEWAYS

3.1 Sicherheitsphilosophie

Axiom 1 (Murphy):

Alle Programme haben Fehler.

Satz 1 (Gesetz der großen Programme)

Große Programme sind noch fehlerhafter als ihre Größe vermuten läßt.

Konklusion 1

Sicherheitsrelevante Programme haben sicherheitsrelevante Fehler.

Satz 2

Es spielt keine Rolle, ob ein Programm Fehler hat oder nicht, wenn Sie es nicht benutzen.

Konklusion 2

Es spielt keine Rolle, ob ein Programm sicherheitsrelevante Fehler hat, wenn Sie es nicht benutzen.

Satz 3

Gefährdete Computer sollten so wenig Programme wie möglich fahren. Die eingesetzten Programme sollten so klein wie möglich sein.

Korollar 3 (Fundamentalsatz für Firewalls)

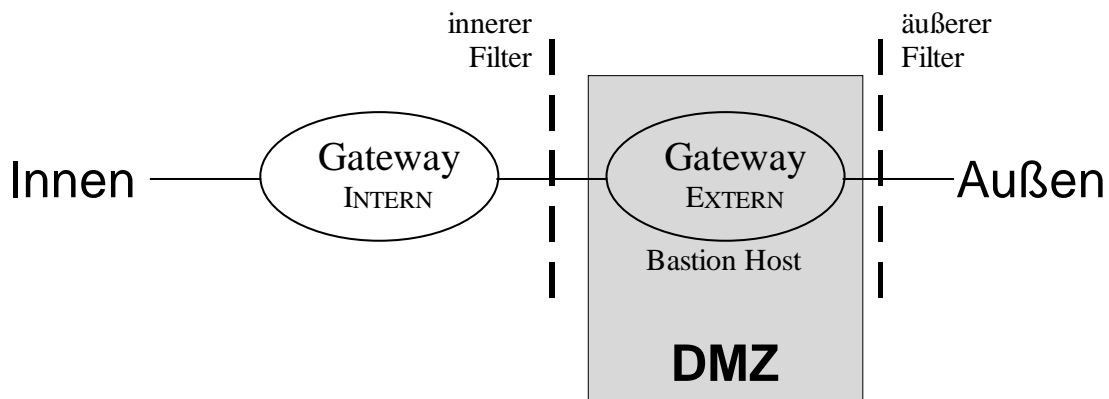
Die meisten Hosts genügen unseren Anforderungen nicht: Sie fahren zu viele, zu große Programme. Die einzige Lösung ist daher, sie mit einer Firewall abzuschotten, wenn Sie überhaupt irgendwelche Programme benutzen wollen.

3.2 Grundidee

Eine *Firewall* besteht im allgemeinen aus verschiedenen Komponenten. "Filter" (engl. "Screens") schleusen nur ganz bestimmte Klassen von Verkehr durch und blockieren alle anderen. Ein *Gateway* besteht aus einer oder mehreren Maschinen, die als Relais für bestimmte, durch Filter blockierte Dienste dienen. Das Gateway-Datennetzsegment wird oft auch *demilitarisierte Zone (DMZ)* genannt.

Ein Gateway in der DMZ wird häufig durch den *internen Gateway* ergänzt. Im allgemeinen gesteht man dem äußeren Gateway durch den inneren Filter hindurch eine freizügigere Kommunikation zu dem internen Gateway zu, als zu anderen inneren Host. In der Regel dient der äußere Filter dem Schutz des Gateways selbst, während der innere das interne Datensegment schützt, falls der Gateway gekapert wird. Jeder einzelne oder auch beide Filter können das interne Datennetz von Angriffen von außen schützen. Einen exponierten Gateway nennt man *Bastion Host* (Vorwerk).

3.3 Schema einer Firewall



Man unterscheidet drei Hauptkategorien von Firewalls:

- Paketfilter
- Vermittler- oder Transportschicht-Gateway (circuit gateway)
- Anwendungsschicht-Gateway (application gateway)

Meist werden mehrere Typen gleichzeitig eingesetzt. Wie schon erwähnt wird Mail häufig selbst dann über ein Gateway geleitet, wenn gar keine Firewall benötigt wird.

3.4 Kosten

3.4.1 Kosten einer Firewall

- Kauf der Hardware
- Wartung der Hardware
- Entwicklung oder Kauf der Software
- Wartung der Software
- Installation und Personalschulung
- kontinuierliche Administration und Troubleshooting
- Geschäftseinbußen durch blockierte Dienste oder Störungen im Gateway
- Verzicht auf die Dienste oder Annehmlichkeiten einer offenen Anbindung

3.4.2 Mögliche Kosten des Verzichtes

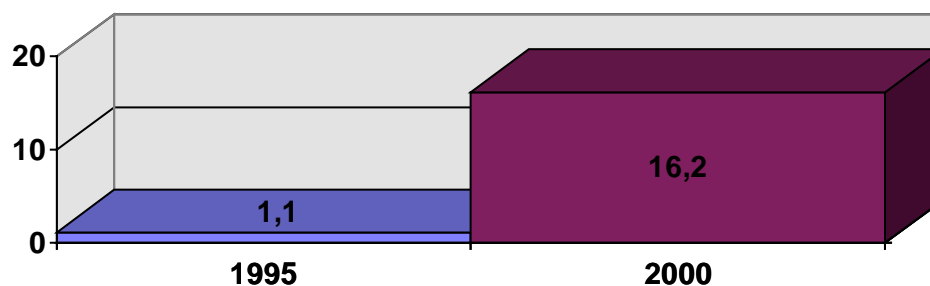
- nötige Maßnahmen zur Bekämpfung eines erfolgreichen Einbruchs und zumWiederanlauf des Betriebs inklusive der Geschäftseinbußen
- juristische und andere Konsequenzen wegen der Duldung oder Unterstützung von Hackern

3.5 Firewall Marktentwicklung

Marktforscher vermuten, daß sich der Markt für Firewalls und damit verbundenen Dienstleistungen weltweit von derzeit 1,1 Mrd. Dollar bis zum Jahr 2000 auf 16,2 Mrd. wachsen wird. Dies entspricht einer jährlichen Steigerungsrate von 70 Prozent.

3.5.1 Firewall Software und -Dienstleistungen

in Mrd. Dollar weltweit



Bisher hat sich noch kein Marktführer für "schlüsselfertige" Firewalls herauskristallisiert. Digital, Sun, IBM, Trusted Information, CheckPoint werden jedoch als vielversprechende Anwärter auf diesen Titel gehandelt.

3.6 Platzierung von Firewalls

Traditionell werden Firewalls zwischen der Organisation und dem Rest der Welt errichtet. Große Organisationen können aber durchaus interne Firewalls benötigen, um administrative oder Sicherheitsbereiche, "security domains" zu isolieren.

Datennetz Abbildung 3.2, S. 64

Der Pfeil zeigt nach außen, in Richtung der Bedrohung. Im Beispiel traut NETZ 1 keinem anderen Netz. Die Firewall kontrolliert Zugang und Vertrauen auf nachvollziehbare Weise.

3.6.1 Transitives Vertrauen

Transitives Vertrauen kann sich als Problem erweisen. Angenommen, Maschine A in

Abbildung 3.3 vertraut in völligem Einklang mit ihrem lokalen Sicherheitskonzept der Maschine B. Diese wiederum vertraut gemäß ihrem Sicherheitskonzept der Maschine C. Die Konsequenz ist, daß Maschine A, ob sie will oder nicht, und möglicherweise ohne ihr Wissen, nun Maschine C traut. Eine Firewall verhindert dies. Die Dioden in Abbildung 3.2 deuten an, daß Maschine A der Maschine B oder diese der Maschine C kein Vertrauen aussprechen kann. Die Maschine C könnte aber durchaus der Maschine B vertrauen. Vertrauensverhältnisse können also mittels Firewalls kontrolliert werden.

3.7 Paketfilter

Paketfilter sind ein preiswerter und nützlicher Sicherheitsmechanismus für Gateways. Sie sind billig oder gar kostenlos, denn die Router-Software hat schon die Fähigkeit zur Paketfilterung, und man braucht meist sowieso einen Router, um sich überhaupt ans Internet anzuschließen. Selbst wenn der Router dem Provider gehört, wird dieser die Filter gerne installieren.

Paketfilter werfen Pakete in Abhängigkeit ihrer Quell- oder Zieladressen oder -Ports. Dies geschieht im allgemeinen kontextfrei, nur auf den Inhalt des aktuellen Paketes gestützt. Je nach Typ des Routers erfolgt die Filterung entweder bei eingehenden oder abgehenden Pakete oder bei beiden. Der Systemverwalter gibt eine Positivliste akzeptabler, und eine Negativliste der verbotenen an. Damit läßt sich auf Host- und Datennetz der Zugang leicht steuern. So kann man beispielsweise beliebigen IP-Verkehr zwischen den Hosts A und B zulassen oder außer für A jeglichen Zugang zu B sperren.

Die meisten Sicherheitskonzepte benötigen feinere Kontrollmöglichkeiten. Nicht vertrauenswürdige Maschinen soll der selektive Zugang zu einzelnen Diensten gewährt werden. So möchte man vielleicht jedem Host erlauben, den Mail-Service der Maschine A zu kontaktieren. Der Zugang zu anderen Diensten soll unabhängig hiervon regelbar sein. Dies läßt sich zum Teil mit Paketfiltern erreichen, doch die Konfiguration ist schwierig und fehleranfällig. Um sie korrekt durchzuführen, braucht man detailliertes Fachwissen über TCP- und UDP-Port-Belegung in mehreren Betriebssystemen.

Dies ist einer der Gründe, der gegen Paketfilter spricht. Champman [1992] hat gezeigt, daß Konfigurationsfehler bei der Paketfilterung leicht Hintertürchen für den Angreifer öffnen.

Selbst wenn der Filter völlig korrekt eingestellt wurde, können manche Kompromisse gefährliche Konsequenzen haben - mehr dazu später.

Die Konfiguration eines Paketfilters ist ein dreistufiger Prozeß. Zu allererst muß man sich darüber klar sein, was erlaubt und was verboten sein soll. Man muß also ein Sicherheitskonzept haben.

Als nächstes müssen die verschiedenen Paketklassen formal als logische Ausdrücke über Paketinhalte spezifiziert werden. Schließlich müssen die Ausdrücke in die vom Router unterstützte Syntax übersetzt werden.

3.7.1 Beispiel

Sicherheitskonzept

- eingehende Mail (SMTP, Port 25), allerdings nur von der Gateway Maschine
- keine Mail von QUAXI, weil er immer Gigabyte schickt

Filer

Zugang	Wir	Port	Sie	Port	Kommentar
blockiert	*	*	QUAXI	*	nicht vertrauenswürdig
erlaubt	Unser-GW	25	*	*	Zugang zum SMTP Port

Die Regeln werden gemäß ihrer Reihenfolge angewandt. Pakete, deren Zugang nicht explizit erlaubt ist, werden blockiert. Dies entspricht folgender impliziter Regel am Ende des Regelwerks:

Zugang	Wir	Port	Sie	Port	Kommentar
blockiert	*	*	*	*	Default

und folgt der generellen Entwurfsphilosophie: Was nicht explizit erlaubt ist, ist verboten.

Unterschiede

Folgendes Regelwerk soll "jeder Host innen darf Mail nach außen schicken" implementieren.

Zugang	Wir	Port	Sie	Port	Kommentar
erlaubt	*	*	*	25	Verbindung zum fremden SMTP Port

Die Verbindung kann von einem beliebigen Port auf einer Maschine ausgehen, muß aber außen den Port 25 als Ziel haben. Das Regelwerk scheint klar und offensichtlich - und es ist falsch.

Der Fehler liegt darin, daß die Filterung ausschließlich auf Port-Nummern fremder Maschinen basiert. Port 25 ist zwar tatsächlich der normale Mail-Port, bloß können wird dies auf fremden Hosts nicht sicherstellen. Ein Angreifer könnte jeden beliebigen Port unserer internen Computer attackieren, indem er seine Verbindung vom Port 25 der externen Maschine ausgehen läßt.

Besser wäre, nur *abgehende* Verbindungen mit Zielport 25 zuzulassen. Wir erlauben also unseren Computern, fremde Ports 25 zu kontaktieren, da wir ihr Anliegen kennen: Mail-Zustellung. *Eingehende* Verbindungen von Port 25 könnten einen beliebigen Dienst nach Wahl des Angreifers realisieren. Glücklicherweise kann die Unterscheidung zwischen eingehenden und abgehenden Paketen leicht durch einen einfachen Paketfilter getroffen werden, indem wir unsere Notation erweitern.

Bei einer TCP-Verbindung fließen Pakete immer in beide Richtungen. Selbst wenn die Daten ausschließlich in eine Richtung strömen, sind Quittungs- und Steuerpakete in die Gegenrichtung unterwegs. Wir können unser Ziel erreichen, indem wir die Flußrichtung des Pakets und einige Steuerfelder untersuchen. Insbesondere hat das erste Paket zum Aufbau einer Verbindung einer TCP-Verbindung im Gegensatz zu allen anderen TCP-Paketen im Kopf kein ACK-Bit. Pakete mit gesetztem ACK-Bit gehören also zu einer laufenden Verbindung, Pakete ohne es stellen einen nur für interne Hosts erlaubten Verbindungsaufbau dar. Externen Computern soll der Verbindungsaufbau verweigert, aber die Fortführung von Verbindungen ermöglicht werden. Interne Kernel akzeptieren keine Fortsetzungspakete für nicht vorhandene TCP-Verbindungen.

Damit können wir mit den Begriffen Quell- und Zieladresse statt des nebulösen "WIR" und "SIE" folgendes Regelwerk formulieren:

Zugang	Quelle	Port	Ziel	Port	Attr	Kommentar
erlaubt	{unsere Hosts}	*	*	25		unsere Pakete zu ihrem SMTP-Port
erlaubt	*	25	*	*	ACK	ihre Antworten darauf

Die Schreibweise "{unsere Hosts}" steht für eine Menge zulässiger Maschinen.

3.7.2 Filtern von FTP-Sitzungen

Dateien werden bei FTP über eine zweite Verbindung transportiert. Nutzt der Kontrollkanal zum Server IHRHOST die Verbindung

<UnserHost, UnserPort, IhrHost, 21>

so erfolgt der Dateitransfer üblicherweise über den Datenkanal

<UnserHost, UnserPort, IhrHost, 20>

Zudem geht der Verbindungsaufbau vom Server aus. Wir sehen uns mit einem bekannten Problem konfrontiert, diesmal aber ohne die Möglichkeit, anhand der Richtung des Verbindungsaufbaus zu selektieren.

Ein mögliches Selektionskriterium wäre der Wertebereich für UnserPort. Die meisten Server, und damit die meisten Angriffsziele, finden sich auf niedrigen Port-Nummern, während abgehende Verbindungen eher von hohen (also oberhalb von 1023) Port-Nummern ausgehen. Ein Regelsatz könnte dann etwas so aussehen:

Zugang	Quelle	Port	Ziel	Port	Attr	Kommentar
erlaubt	{unsere Hosts}	*	*	*		abgehende Verbindungen
erlaubt	*	*	*	*	ACK	Antworten für uns
erlaubt	*	*	*	>1023		nichtprivilegierte Verbindungen

Pakete werden also durchgeschleust, wenn sie eine der folgenden Bedingungen erfüllen:

- Sie stammen von einer unserer Maschinen
- Es sind Antworten innerhalb einer von uns initiierten Verbindung

- Sie zielen auf eine hohe Port-Nummer bei uns

Genaugenommen beziehen sich die beiden letzten Regeln nicht nur auf auswärtige Pakete, sondern auf alle. Allerdings werden Pakete von innen schon von der ersten Regel akzeptiert und daher die folgenden nicht mehr angewandt.

Eine andere Möglichkeit wäre der FTP-PASV Befehl. Man könnte den Client dahingehend modifizieren, dem Server einen PASV zu geben. Der Server wählt dann ein beliebige Port Nummer und fordert den Client auf, die Verbindung zu initiieren.

3.7.3 Die Zählung des DNS

Der Umgang mit DNS ist eines der kniffligsten Probleme beim Aufbau einer Firewall. Der Dienst selbst ist für ein Gateway unerlässlich, doch er birgt viele Gefahren.

Chapmans Ansatz besteht darin, Name-Server für die Domain sowohl auf dem Gateway als auch auf einer internen Maschine zu fahren. Letztere hat die echten Informationen - der Name-Server auf dem Gateway (der "öffentliche") liefert nur minimale Auskünfte wie zB:

bar.com.	IN	NS	foo.bar.com.
bar.com.	IN	NS	x.trusted.edu.
foo.bar.com	IN	A	2000.2.3.4
x.trusted.edu	IN	A	5.6.7.8
foo.bar.com	IN	MX	foo.bar.com.
*.bar.com	IN	MX	foo.bar.com.
bar.com	IN	MX	foo.bar.com.
ftp.bar.com	IN	CNAME	foo.bar.com

Es werden nur die Name-Server selbst (FOO.BAR.COM und X.TRUSTED.EDU) sowie das Relaisystem für Mail und FTP (wieder FOO.BAR.COM) angegeben und alle Mail für Maschinen in der BAR.COM-Domain über das Relais geleitet.

Knifflige Details sind:

- der Zugriff des Gateways auf interne Namen, etwa für die Mail-Zustellung
- der Zugriff interner Maschinen auf externe Namen
- das Durchschleusen der nötigen UDP-Pakete durch die Firewall

Das erste Problem wird durch eine `/etc/resolv.conf` Datei auf dem Gateway, die auf den internen DNS-Server verweist, erschlagen. Applikationen auf dem Gateway, nicht aber der Name-Server selbst, lösen ihre Adreßanfragen anhand dieser Datei. Wenn also beispielsweise *mail* die IP-Adresse zu einem Namen sucht, fragt es den internen DNS-Server.

Der Name-Server-Prozeß selbst ignoriert die `/etc/resolv.conf` Datei. Er bearbeitet Anfragen allein anhand der Baumstruktur des Namensadressbaumes und seines Wissens für Teilbaume zuständige Name-Server. Anfragen nach unbekannt Namen werden auf diese Weise korrekt aufgelöst.

Das zweite Problem sind an den internen DNS-Server gerichtete Anfragen nach externen Namen. Natürlich kennt dieser Server keine externen Maschinen. Statt mit den wirklichen, externen DNS-Servern direkten Kontakt aufzunehmen (das können wir nicht zulassen, da wir Antworten nicht auf sichere Weise durch die Firewall schleusen können), kontaktiert er das Gateway, welches in seiner Konfigurationsdatei in einem `forward` vermerkt ist. Dieser Befehl gibt an, welcher Server wegen unbekannter Namen befragt werden soll. Damit beantwortet er also Fragen nach internen Maschinen unmittelbar und leitet Anfragen nach externen Maschinen an den Name-Server des Gateways weiter.

Folien

Gateway ruft intern/extern
Interne ruft intern/extern

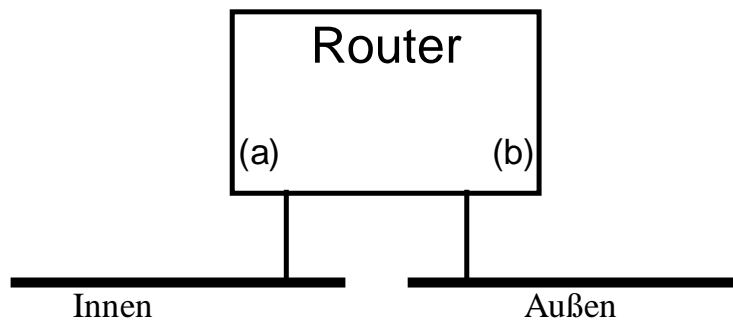
Insbesondere interessant ist dieses Vorgehen dann, wenn Prozesse auf dem Gateway nach externen Namen fragen. Zuerst geht die Anfrage zum internen Server, der die Antwort gar nicht kennen kann. Dann springt sie über die Firewall zurück zum Name-Server des Gateway und von dort weiter zu externen DNS-Servern, von denen schließlich eventuell einer die Antwort kennt. Diese Antwort kehrt schließlich denselben verschlungenen Pfad zurück.

Der interne und der Gateway-Name Server können sich deswegen durch den Paketfilter hindurch unterhalten, weil beide den offiziellen DNS-UDP-Port(53) als Quellport verwenden, wenn sie Anfragen weiterleiten. Dies erlaubt eine sichere Filterregel, die den Durchgang von Paketen vom Gateway zum Port 53 des internen Servers gestattet. Die Konfiguration des Routers stellt sicher, daß falsche Quelladressen für solche Pakete unterbunden werden.

Dies löst das dritte Problem.

3.7.4 Platzierung der Filter

Bei einem typischen Firewall-Router



kann die Paketfilterung an mehreren Stellen erfolgen. Pakete können entweder am Punkt (a) oder (b) oder an beiden gefiltert werden, und zwar entweder die eingehenden oder die abgehenden Pakete oder auch beide Richtungen. Die Filterstrategie hängt von den Fähigkeiten des Routers ab: Nicht jeder erlaubt all diese Möglichkeiten, und manche besitzen noch einige mehr.

Filtern am Router-Ausgang effizienter sein, da sich die Anwendung der Selektionskriterien meist mit Wegwahloperationen kombinieren läßt. Andererseits gehen Informationen verloren, etwa über welche Leitung das Paket empfangen wurde. Dieses Wissen ist besonders wichtig bei der Abwehr von Adreßfälschungen. Filtern am Router-Eingang bietet auch dem Router selbst Schutz vor Angriffen. Verdächtige Pakete sollten so früh wie möglich ausgesondert werden. Es ist nicht ausreichend, TCP-Antworten abzufangen, manche Angriffe funktionieren, ohne daß der Angreifer jemals eine Antwort zu Gesicht bekommt. Entsprechend sollte also der erste Regelsatz so formuliert werden:

Zugang	Quelle	Port	Ziel	Port	Kommentar
blockiert	QUAXI	*	*	*	nicht vertrauenswürdig
erlaubt	*	*	UNSER-GW	25	Zugang zu unserem SMTP
erlaubt	UNSER-GW	25	*	*	unsere Antworten

statt unsere Antworten zu filtern:

Zugang	Quelle	Port	Ziel	Port	Kommentar
blockiert	*	*	QUAXI	*	
erlaubt	*	*	UNSER-GW	25	
erlaubt	UNSER-GW	25	*	*	

Netztopologie und Adreßschwindeleien

Aus wirtschaftlichen Gründen ist es manchmal wünschenswert, denselben Router als Firewall und für internes Routing zu verwenden.

Sicherheitskonzept:

- Sehr eingeschränkte Verbindungen zwischen GW und der Außenwelt werden durch den Router geschleust

- Zwischen GW und Systemen in NETZ 2 oder NETZ 3 werden sehr eingeschränkte Verbindungen zugelassen. Dies ist ein Schutz, falls GW gekapert wird.
- Zwischen NETZ 2 und NETZ 3 herrscht uneingeschränkter Verkehr.
- Zwischen dem externen Netz und NETZ 2 oder NETZ 3 sind nur abgehende Verbindungen erlaubt.

Welche Sorte von Filterregeln wollen wir benutzen? Wenn wir nur am Ausgang filtern, wird es ziemlich schwierig. Eine Regel, die offenen Zugang zum NETZ 2 gestattet, muß sich darauf verlassen können, daß eine Quelladresse auch wirklich zu NETZ 3 gehört. Nichts hindert aber einen Angreifer daran, von außen Pakete zu senden, die behaupten, von einer internen Maschine zu stammen. Wichtige Information - nämlich, daß rechtmäßige Pakete von NETZ 3 nur über eine bestimmte Leitung kommen können - würde vernachlässigt. Solche Adressschwindeln sind nicht ganz einfach, aber keineswegs unrealistisch.

Wir wollen also annehmen, daß wir an den Eingängen filtern und daß wir jede abgehende Verbindung, aber eingehende Verbindungen nur für Mail und nur zu unserem Gateway zulassen. Der Regelsatz an der externen Schnittstelle sollte dann lauten:

Zugang	Quelle	Port	Ziel	Port	Attr	Kommentar
blockiert	{NETZ 1}	*	*	*		Fälschungen ausperren
blockiert	{NETZ 2}	*	*	*		
blockiert	{NETZ 3}	*	*	*		
erlaubt	*	*	GW	25		eingehende Mail-Verb
erlaubt	*	*	{NETZ 2}	*	ACK	Rückantworten für abg. Verb.
erlaubt	*	*	{NETZ 3}	*	ACK	

In Worten:

- Adressfälschungen vermeiden
- eingehende Mail Verbindungen zum Gateway
- Rückantworten innerhalb beliebiger abgehender Verbindungen erlauben
- Alles andere wird abgewiesen

Paketfilter und UDP

Es ist schwierig TCP-Verbindungen zu filtern. Es ist beinahe unmöglich, UDP-Pakete ohne Funktionalitätsverluste zu filtern. Dies liegt an einem grundsätzlichen Unterschied zwischen TCP und UDP: TCP ist verbindungsorientiert und merkt sich daher Kontext, während UDP paketorientiert ist, und daher die Datagramme unabhängig voneinander betrachtet werden. Bei TCP erlaubt uns das ACK-Bit die Unterscheidung zwischen eingehenden und abgehenden Verbindungen. Doch bei UDP fehlt uns solches Selektionskriterium.

Angenommen, ein interner Host möchte den UDP-Dienst Echo eines externen Systems in Anspruch nehmen. Das abgehende Paket hat das Adreßtupel

<interner Host, interner Port, externer Host, 7>

wobei interner Port im unprivilegierten Bereich liegt. Die Antwort ist

<externer Host, 7, interner Host, interner Port>

Für den Router ist nicht erkennbar, ob interner Port wirklich ein harmloses Ziel ist.

Ein eingehendes Paket

<externer Host, 7, interner Host, 2049>

stellt wahrscheinlich einen Angriff auf unseren NFS-Server dar. Wir könnten zwar die Angriffsziele explizit aufführen, aber wissen wir, welche neuen Schwachstellen irgendein Systemverwalter in irgendeiner abgelegenen Ecke unseres Datennetzes nächst Woche installiert.

Ein konservatives Sicherheitskonzept fordert daher, abgehenden UDP-Verkehr praktisch vollständig zu verbieten. Nicht, daß im abgehenden Verkehr selbst irgendeine Gefahr begründet wäre, doch können wir den Antworten darauf nicht trauen.

3.8 Anwendungsschicht Gateways

Anwendungsschicht Gateways stellen das andere Extrem beim Firewall Entwurf dar. Statt den gesamten Verkehrsfluß mit einem Allzweckmechanismus zu kontrollieren, wird für jede gewünschte Applikation spezialisierter Code eingesetzt. Dies ist gewiß hoher Aufwand, aber wahrscheinlich viel sicherer als jede Alternative. Man braucht sich nicht um Wechselwirkungen zwischen verschiedenen Filterregeln zu sorgen, noch um Lecks in den vorgeblich sicheren Diensten, die tausende interner Hosts außen anbieten. Es genügt, einige ausgewählte Programme unter die Lupe zu nehmen. Anwendungsschicht-Gateways bieten einen weiteren, für

gewisse Anwendungen recht wichtigen Vorteil: der gesamte ein- und abgehende Verkehr kann kontrolliert und protokolliert werden. Das SEAL-Paket von DEC nutzt dies aus. Abgehender FTP-Verkehr wird nur für befugte Personen zugelassen. Ziel ist es, den Diebstahl wertvoller Programme oder Daten der Firma zu verhindern. Dies ist allerdings von begrenztem Nutzen gegen interne, die gewünschte Dateien leicht auf Disketten oder Streamer kopieren könnten. Es ist aber auch nicht die Aufgabe einer Firewall, gegen Angriffe von innen zu schützen - sie schottet bloß nach außen ab.

3.8.1 E-mail

E-mail wird, unabhängig von der sonstige Firewall Technologie, häufig mittels Anwendungsschicht-Gateways realisiert. Mail-Gateways sind, unabhängig von allen Sicherheitsgedanken, an und für sich nützlich. Benutzer bleiben unter der gleichen Adresse erreichbar, unabhängig davon, auf welcher Maschine sie gerade arbeiten. Der Mail-Gateway kümmert sich auch um die korrekte Formatierung der Mail-Header, um Protokollierung und bietet einen zentralen Überwachungspunkt für das Mail-System. Gateways gelten in der Tat als so wichtig, daß im DNS zu ihrer Unterstützung eigens der MX-Eintrag vorgesehen ist. Keine andere Applikation hat einen definierten indieketen Zugangsmechanismus.

Vom Sicherheitsstandpunkt aus ist der Nutzen sogar noch größer. Interne Systemnamen, und damit eventuell wertvolle Details können verborgen werden. Protokollierung und Analyse des Verkehrsflusses und des Postinhaltes zum Auffinden von Informationslecks ist durchführbar.

Häufig werden Anwendungsschicht-Gateways gemeinsam mit Paketfiltern und Transportschicht Gateways eingesetzt. Ein clever programmiertes Gateway könnte den Datenfluß modifizieren und so Versuche, falsche `.rhosts` Dateien oder shells mit S-Bit zu installieren, vereiteln.

Die Selektionskriterien hängen von lokalen Bedürfnissen und Bräuchen ab. Anlagen mit vielen PC-Benutzen wollen eingehende Daten vielleicht auf Viren untersuchen.

Der Hauptnachteil der Anwendungsschicht-Gateways liegt darin, daß für die meisten angebotenen Dienste spezielle Benutzeroberflächen oder -programme bereitgestellt werden müssen. Die praktische Konsequenz ist, daß in der Regel nur die wichtigsten Dienste angeboten werden.

3.9 Transportschicht Gateways

arbeiten auf der Verbindungsschicht. Sie vermitteln als Relais TCP-Verbindungen. Eine externe Verbindung geht auf einem TCP-Port des Gateways ein, dieser kontaktiert dann ein internes Ziel. Während die Verbindung besteht, kopiert das Gateway die Daten zwischen den Schnittstellen um.

Manchmal wird die Verbindung automatisch hergestellt. Bei AT&T beispielsweise muß ein externer Host auf einen internen Drucker zugreifen. Er spricht den Druckdienst des Gateway an, der diese spezielle Verbindung an den Druckdienst der zuständigen internen Maschine durchreicht. Um sicherzustellen, daß nur dieser bestimmte externe Host auf den Druckdienst des Gateways zugreifen kann, benutzes sie Zugangskontrollen. Die Authentifizierung ist so stark, daß diese Verbindung selbst dann kein Schlupfloch bietet, wenn der externe Host gekapert wird.

In anderen Fällen muß den Vermittlungsdiensten das gewünschte Ziel mitgeteilt werden. Dazu wird ein kleines Protokoll zwischne Client und Gateway benötigt. Per Protokoll fordert der Client Ziel und Dienst an und empfängt gegebenenfalls Fehlermeldungen des Gateways. Eine solche Implementierung nennt man häufig *proxy*. Ist die Verbindungsanfrage erfolgreich, terminiert das Protokoll und der eigentliche Datentransfer beginnt. Diese Dienste setzten Modifikation im Client Programm oder seiner Bibliotheken voraus.

Im allgemeinen kontrollieren Relaisdienste den durchfließenden Datenstrom nicht, Größe und Ziel wird normalerweise protokolliert.

Für eingehende Logins müssen sich Benutzer mit Einmal-Paßwort-Geräten gegenüber einem Authentifikationsserver ausweisen, ehe sie Zugang zu internen Maschinen gestettet bekommen. Erhalten sie diese Erlaubnis, so schaltet sie der Gateway mittels einer vorauthentifzierten Verbindung, wie etwa rlogin, nach innen durch.

