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 wohlgesonnen 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 Inforamtion 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 sochen 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 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

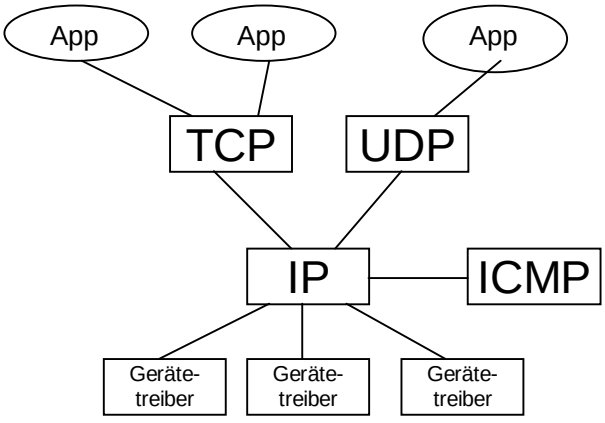
Nicht immer dient ein Angriff dem Erlangen neuer Informationen. Manche Leute stehen darauf, Autoreifen aufzuschlitzen oder Wände zu verunstalten. Vandalismus ist ein uraltes Phänomen. Die primitivste und einfachste Form im high-tech Zeitalter des Internets ist es, fremde Festplatten zum Überlaufen zu 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 ungesichertet 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 wegwerdenen, wenn er überlastet ist. Die Pakete können mehrfach ankommen oder in anderer Reihenfolge als sie gesendet wurden. All dies geschiet in der Regel stillschweigend: Es wird davon ausgegangen, daß höhere Protokollschichten (dh TCP) den Applikationen eine gesicherte Verbindung bereitstellen, indem sie solche Probleme erkennnen 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 spzielle 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 beitet 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 - Adress 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 umsetzten. 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 einen 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 derselbsen virtuellen Verbindung zu erkennen.

Jede TCP-Nachricht enthält den4-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 anbeiten, 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 behandlet 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 *priviligierten 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 Fehlerkorrekture 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 Quelladresses 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 abzubrechen.

Viele ICMP Nachrichten, die einen Host erreichen, sind nur für eine bestimmte Verbindung relevant oder durch ein bestimmtest Paket ausgelöst. So sollte eine Redirect- oder Destination Unreachable-Nachricht sich bloß auf eine bestimmte Verbindung beziehn. 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 Hostals Antwort auf ein Paket zum Host PING.CO.AT ein Destination Unreachable erhält, weil dieses eine 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, wir zB PING.CO.AT) Host-Namen in verwendbare IP-Adressen umsetzt und umgekehrt. Im Normalbetrieb senden Hosts UDP-Anfragen an DNS-Server. Dieses entworten 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 Namesnadreßraum von DNS ist baumförmig. Um den Betrieb zu erleichtern, können genze 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 Gegenszug 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, fehldende 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 Standad 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 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 sendenen 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 vorzutäuschen. 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 potentielles 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 sogaenannte *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 aufsetzten. Die meisten grundlegenden Charakteristika des Transportmechanismus scheinen durch. Ein Subsystem, welches RPC über UDP nutzt, muß sich also mit verlornenen, 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-Authtifikation, bestehend aus 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 *rpxinfo*-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 Vlient 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 erschwindelte 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 entwicklet, wird NFS haeute 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 Zugriffskontrollisten durchsetzten. 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 schoon 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 Dateitransferprotkolle

2.4.1 TFTP - Trivial File Transport Protocol

TFTP ist ein einfaches UDP-basiertes Dateitransferprotokol 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 Verzeichnisinhalt, 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 durchzusetzten 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
```