Datavetenskap

Christian Ohlsson & Emil Hevald

Brandväggar

Examensarbete

2000:30

Denna rapport är skriven som en del av det arbete som krävs för att erhålla en kandidatexamen i datavetenskap. Allt material i denna rapport, vilket inte är vårt eget, har blivit tydligt identifierat och inget material är inkluderat som tidigare använts för erhållande av annan examen.

Christian Ohlsson

Emil Hevald

Godkänd, 8 Juni 2000

Handledare: Niklas Nikitin

Examinator: Stefan Lindskog

Sammanfattning

Detta är ett examensarbete som handlar om brandväggar. Syftet med detta examensarbete är att studentdatorerna vid Karlstads universitet skall skyddas mot attacker från Internet med hjälp av en brandvägg. Det skall även finnas en installationsguide som hjälper en systemadministratör att installera och underhålla brandväggen.

Vi kommer att belysa vad en brandvägg är, vilka typer av brandväggar som finns att tillgå och slutligen vilken brandvägg som vi ansåg var den mest lämpade för syftet. Vi löste detta problem genom att söka på Internet efter information om vilka typer av brandväggar som fanns och sedan rådfråga andra personer som ställts inför liknande problem tidigare. Detta resulterade i att vi kom fram till sex stycken brandväggar som vi testade vidare, och slutligen kom vi fram till den lösning vi ansåg var bäst.

Firewalls

This is a bachelor's project about firewalls. The purpose of this project is to protect student-computers, at the Karlstad University, against attacks from the Internet by use of a firewall. The report will also contain an installation guide, which will help a system administrator install and maintain the firewall.

We will describe what a firewall is, which types of firewalls there are, and finally which firewall we considered most appropriate for our purpose. We obtained the information needed to accomplish this by searching the Internet and by consulting people with prior experience of similar tasks. Our research yielded six firewalls that were of interest to us. These were then tested futher, and we made the final choice of firewall.

Tack

Ett varmt tack till vår handledare Niklas Nikitin. Utan hans hjälp och hans goda råd skulle detta examensarbete ej kunna ha genomförts.

Innehåll

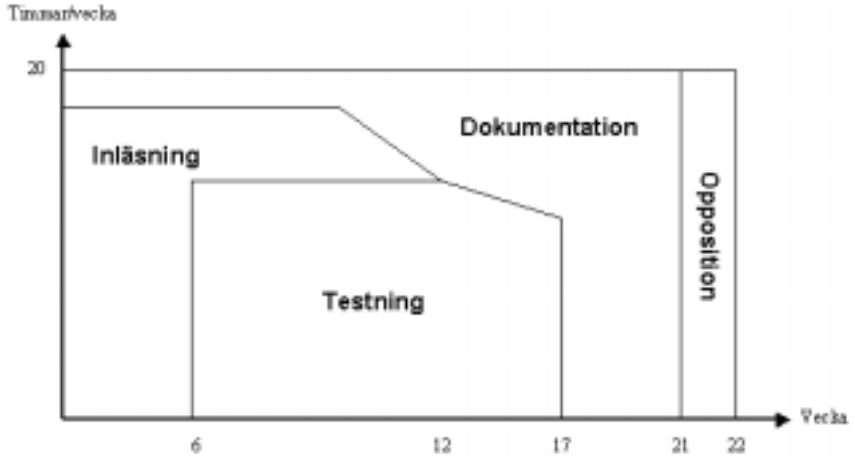
Figurer

1 Introduktion

Önskemålet från uppdragsgivaren, Instutionen för Informationsteknologi vid Karlstads universitet, var att ge studentdatorerna ett skydd mot angrepp från Internet. All trafik från studentdatorerna mot Internet skall vara tillåten, medan man från Internet endast kan komma in på studentnätet för att titta på hemsidor och göra filöverföringar. Man skall från studentdatorerna även kunna kolla mail, samt utföra en del andra tjänster. Detta var de grundlägande kraven på vårt arbete. Övriga krav diskuterades fram under arbetets gång med vår uppdragsgivare.

Ett problem av den här typen löser man med hjälp av en brandvägg. Brandväggens uppgift är att ta emot all trafik mot ett nät, som oftast är ett internt nätverk, och kontrollera vilken sorts tjänst som avsändaren avser att utnyttja. Om man tillåter avsändaren att använda denna tjänst kommer trafiken att släppas igenom, annars ej. Vi kommer att diskutera mer grundligt hur en brandvägg fungerar i kapitel 3.1.

Innan vi började arbetet gjorde vi en tidsplan som kan ses i figur 1.1. Detta var en grov uppskattning av hur vi skulle fördela den tid vi hade till förfogande.



Figur 1.1: Tidsplan

2 Ordlista

CPU	Central Processing Unit.
DHCP	Dynamic Host Configuration Protocol.
DMZ	DeMilitarized Zone.
DNS	Domain Name System.
FTP	File Transfer Protocol.
GNU	Gnu's Not Unix.
HTTP	Hyper Text Transfer Protocol.
ICMP	Internet Control Message Protocol.
IDE	Integrated Device Electronics.
IGMP	Internet Group Management Protocol.
IP	Internet Protocol.
ISDN	Integrated Services Digital Network.
ISP	Internet Service Provider.
JDK	Java Development Kit.
LAN	Local Area Network.
LRP	Linux Router Project.
NAT	Network Address Translator.
PERL	Practical Extraction and Report Language.
POP3	Post Office Protocol 3.
RIP	Routing Information Protocol.
SCSI	Small Computer Standard Interface.
SMTP	Simple Mail Tranfer Protocol.
TCP	Transport Controll Protocol.
UDP	User Data Protocol.
URL	Uniformed Resource Location.
WWW	World Wide Web.

3 Bakgrund

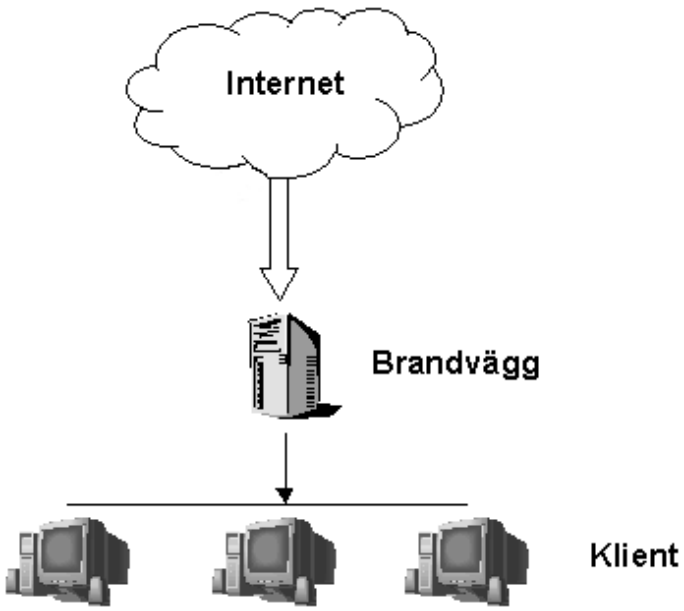
Innan detta examensarbete började hade ingen av oss någon direkt erfarenhet av vare sig brandväggar eller någon annan form av datasäkerhet. Därför började vi med att studera ämnet och lära oss mer om brandväggar. I detta kapitel kommer vi att presentera vad en brandvägg är, vilka olika sorters brandväggar som finns och vad dom kan skydda mot.

3.1 Allmänt om brandväggar

När man pratar om brandväggar är det framförallt tre typer av brandväggar man pratar om. Dessa är hårdvarulösningar, proxylösningar eller paketfilterlösningar. De två sistnämda kan fås i olika gratisvarianter från Internet, och körs oftast på något fritt operativsystem som t.ex. FreeBSD, NetBSD eller Linux. Så vad är nu en brandvägg, vilket man hör folk prata mer och mer om? Den frågan ska vi försöka svara på nu.

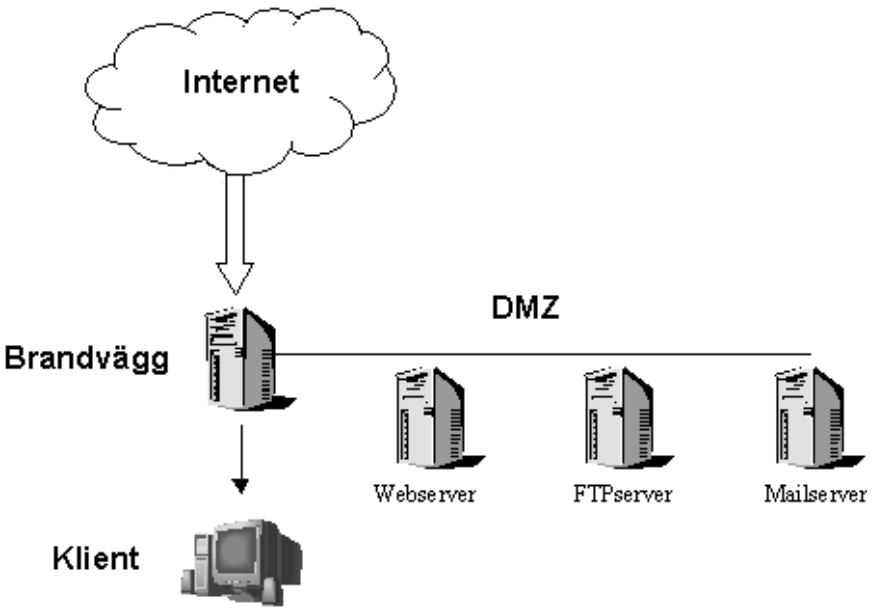
3.1.1 Vad är en brandvägg?

Brandväggar i det här fallet handlar om datasäkerhet. De flesta använder en brandvägg för att skydda sig mot obehörig access utifrån, med utifrån menas att människor uppkopplade mot Internet inte ska kunna få tillgång till data och tjänster på datorer hos de som har brandväggen. Det brandväggen gör är egentligen att antingen släppa igenom data eller stoppa den. All trafik från Internet måste gå igenom brandväggen för att komma fram till en klient på insidan av brandväggen. Brandväggens uppgift är då att försöka minimera den eventuella skada som kan uppstå på de datorer som finns på insidan genom att bara tillåta viss typ av trafik. En skematisk bild på hur den enklaste formen av en brandväggslösning kan se ut visas i figur 3.1 på nästa sida.



Figur 3.1: Brandvägg utan DMZ

Man kan även ha en så kallad tre-benad brandvägg. I ett sådant fall står de tre benen för Internet, lokalt nätverk och DMZ. Anledningen till att man vill ha det extra tredje benet, DMZ, är att man i denna zon kan ha tjänster som både människor på utsidan och insidan av brandväggen skall kunna komma åt. Dessa tjänster kan exempelvis vara web, mail och FTP. I DMZ tillåts endast trafik av den typ som behövs i DMZ. Om någon vill se på en websida dirigerar brandväggen om trafiken till webservern, och motsvarande för mail och FTP. På detta sätt kan man vara säker på att den typ av trafik man dirigerar om till DMZ aldrig kommer att trafikera någon annan dator i nätverket. Denna teknik kallas även portforwarding. Lättast blir det om man tänker på den trebenade brandväggen som en vägkorsning där brandväggen står som en polis i mitten och dirigerar om trafiken. Ett exempel på hur en tre-benad brandvägg ser ut visas i figur 3.2 på motstående sida.



Figur 3.2: Brandvägg med DMZ

3.1.2 Varför vill man ha en brandvägg?

På Internet, precis som över allt annars, finns det människor som är ute efter att förstöra för andra, medan man själv vill göra ett seriöst jobb över Internet. I detta fall kan en brandvägg vara en bra lösning, då den håller de som vill förstöra för en ute, samtidigt som du kan göra ditt jobb.

Det finns även många som använder sin brandvägg till andra tjänster än bara skydd. De kan till exempel ha publika tjänster på den, såsom företagsinformation, filer man kan ladda ner, uppgraderingar till deras produkter osv. Brandväggar kan även användas till 'masqurading'. Läs mer om 'masquerading' i kapitel 3.1.6.

3.1.3 Vad kan en brandvägg skydda mot?

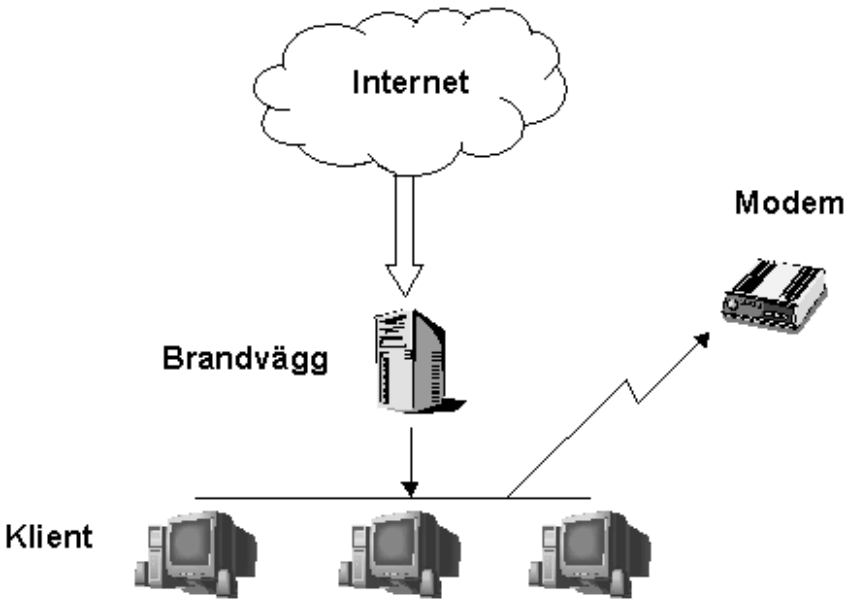
Vissa brandväggar tillåter bara e-mail att passera, i dessa fall skyddar den mot allt utom just attacker mot e-mailsystemet. Andra brandväggar släpper igenom allt utom tjänster

man vet det finns problem med. Mer sofistikerade brandväggar blockerar dock allt från utsidan, medan de på insidan kan göra vad de vill utåt. Brandväggen är också en bra punkt för en systemadministratör att kunna föra statistik och logga saker och ting då all data passerar den på sin väg mot det lokala nätet. I dessa loggar kan man se hur många intrångsförsök och så vidare man haft under en viss tid, och dessutom vem som försökte att bryta sig in. En brandvägg kan även skydda mot 'IP-spoofing'. Detta innebär att man kan utge sig för att vara någon annan än den man är. Med hjälp av 'IP-spoofing' kan man "lura" brandväggen att tro att man kommer inifrån det skyddade nätverket när man i själva verket kommer från Internet.

3.1.4 Vad skyddar inte en brandvägg mot?

Vi ska här börja med att säga att brandväggen inte kan skydda mot saker som inte går genom brandväggen. Många större företag som kopplar upp sig mot Internet bekymrar sig mycket över att data ska läcka ut den vägen. Men tyvärr kan data lika gärna läcka ut via en diskett eller annat datamedia. Dessutom så kanske det finns en modemsanslutning som människor skulle kunna utnyttja för att komma åt känslig information. Det är dumt att bygga en tjock ståldörr om man bor i en trähydda. Det finns organisationer som lägger ut stora pengar på en brandvägg och sedan har dom en modemanslutning bakom brandväggen, enligt figur 3.3 på nästa sida. Detta innebär att man inte behöver gå via brandväggen för att ta sig in i systemet, det räcker att logga in via modemet.

Brandväggen kan heller inte skydda dig mot förrädare som jobbar på företaget. Om någon anställd sparar ner känslig data på till exempel en diskett så kan denne lämna företaget med disketten utan att någon märker det, så det är många saker man bör tänka över. Ren och skär dumhet skyddar den inte heller mot. Om någon dum person kan lura en "vänlig" anställd att öppna upp en modemanslutning som han/hon kan utnyttja har vi ännu ett intrång utan att brandväggen kan skydda dig. Brandväggen kan oftast inte skydda



Figur 3.3: Säkerhetshål på grund av modemanslutning

mot trojaner[1] som körs på något protokoll som till exempel HTTP, SMTP eller liknande. En brandvägg kan inte användas som en ursäkt att inte installera uppgraderingar till de programvaror man använder, eller på annat sätt skära ner på säkerheten.

3.1.5 Skyddar en brandvägg mot virus?

Brandväggar skyddar väldigt dåligt mot virus. Det finns allt för många sätt att gömma ett virus i en körbar fil, och allt för många virustyper, för att en brandvägg ska kunna leta igenom alla. För att öka sitt skydd mot virusattacker bör varje lokal maskin ha anti-virusprogramvara installerat då detta skyddar mot virus som kommer från andra källor än Internet, exempelvis via diskett, modem, eller någon annan form av kommunikation. Det finns dock leverantörer av brandväggar som lovar viruskydd, men detta kommer bara att skydda mot virus som kommer in via brandväggen, om ens det.

[1]Ett program som utger sig för att utföra nytta, men innehåller en dold del som ställer till skada

3.1.6 Vad mer kan en brandvägg göra?

Ett annat användningsområde för brandväggen är att ge access mot Internet till hela sitt LAN även om man bara har fått en IP-adress från sin ISP. Då gör brandväggen något man kallar för *'masqurading'*, som är en egenskap Linux kan göra med hjälp av IPChains. Det finns under andra operativsystem också, dock under annat namn. 'Masqurading' går till som så att den ersätter avsändarens IP-adress i alla paket som kommer från insidan brandväggen med sin egen IP-adress och när paketet kommer tillbaka byter den tillbaka.

3.2 Operativsystem

Det finns ett flertal olika operativsystem som man kan använda för att köra en brandvägg på, bland andra Linux, Microsoft Windows, FreeBSD, Solaris, SunOS, MacOS med flera. Önskemålet från uppdragsgivarens sida var en brandväggslösning för Linux, men de kunde även tänka sig en brandvägg på en Microsoft Windows dator. Anledningen till att de andra operativsystemen inte var lika intressanta var att uppdragsgivaren har goda kunskaper om Linux och Microsoft Windows och ett val av något annat operativsystem skulle troligen ställa till med problem med installation, syntax och så vidare.

De operativsystem som nu återstår är alltså Linux och Windows. De fördelar som finns med att använda Windows är att brandväggsprogrammet torde vara grafiskt, vilket innebär att konfigurering torde vara enklare. Däremot brukar program till Windows vara dyra, och en brandvägg brukar till och med vara väldigt dyr. Linux däremot har ett flertal fördelar, själva operativsystemet är gratis, brandväggslösningen likaså, säkerhetsuppdateringar kommer ofta samt en annan mycket viktig sak: Linux är känt för att vara ett stabilt operativsystem. En brandväggsdator måste vara mycket stabil, en krasch är helt enkelt oacceptabel. Om brandväggsdatorn går ner kommer det att innebära att samtliga datorer på det skyddade nätverket bakom den, och i DMZ, inte kommer att ha någon kontakt med Internet.

3.3 Hårdvara

Hårdvara som krävs beror naturligtvis på hur många användare man har, och vad man valt för brandväggslösning. Kapaciteten på sin Internetuppkoppling har också viss betydelse då det blir mer för brandväggen att göra om man har en snabb uppkoppling. Generellt gäller att om man valt en paketfiltrerande lösning behöver man inte ha en så kraftfull dator. En proxylösning kräver lite bättre hårdvara. Det finns även hårdvarulösningar på brandväggar, dessa består ofta av någon form av optimerad hårdvara och något speciellt operativsystem som tillverkaren optimerat för sin hårdvara. Denna typ av brandvägg är oftast väldigt dyr, och var därför inte aktuell att titta närmare på i denna rapport.

3.4 Proxy-lösningar

En av de brandväggslösningar som finns att tillgå är en så kallad proxy-lösning. Vi kommer i det här kapitlet att beskriva vad en proxy-brandvägg är och vilka olika typer av proxy-brandväggar som finns att tillgå.

En proxy används oftast för att kontrollera eller övervaka utgående trafik. En del proxy sparar information i en 'cache'[2] vilket leder till att man använder mindre bandbredd, och att det går snabbare att hämta data för nästa användare som vill ha den. Att ha information sparad i en cache innebär att hemsidor man besökt tillfälligt sparas på proxyservern för snabbare access vid nästa besök, plus att det ger ett tydligt bevis på vilken information användare har hämtat. Proxylösningen kräver ofta lite kraftigare hårdvara än vad till exempel en paketfiltrerande lösning av brandvägg gör, och proxyservern stöder bara vissa protokoll, exempelvis HTTP, FTP, SMTP osv. Vill man köra något protokoll som inte proxyservern stödjer får man tänka om eller se om det finns en modul[3] som stödjer detta. De flesta proxylösningar är uppbyggda av moduler. Det finns framförallt två typer

[2]Sparar data på disk eller i minnet
[3]Tillägg till proxyprogrammet

av proxylösningar, och de är applikations-proxy och Socks-proxy.

3.4.1 Applikations-proxy

En applikations-proxy jobbar på det sättet att varje applikation som ska använda den måste lokalt ställa in proxyserverns IP-adress och port. När applikationen ska ut på Internet kommer den först koppla upp sig mot proxyservern och säga vad den vill ha tag i, sedan kopplar proxyservern upp sig dit och hämtar datan åt applikationen, och i vissa fall sparas den även i en lokal cache på proxyservern. Eftersom all kommunikation går via proxyservern kan man lätt logga all trafik. Om man har en web-proxy kan man se i loggarna varje URL som någon har besökt, och är det FTP-proxy kan man se vilka filer som laddats upp eller ner. En applikations-proxy kan även filtrera bort vissa olämpliga ord om man nu vill det, och scanna efter virus på det som går igenom den. En annan egenskap en applikations-proxy har är att man kan tvinga användaren som ska använda den att logga in mot proxyservern först. Detta sker innan en uppkoppling görs, för en webanvändare skulle det se ut som om varje hemsida var lösenordsskyddad.

3.4.2 Socks-proxy

En Socks-proxy kan liknas vid en kopplingscentral. Det den gör är att koppla om din uppkopplingsbegäran till rätt adress på utsidan. Man kör framför allt två versioner av Socks. Den senaste version, Socks V5 eller Socks5 som det skrivs ibland, stödjer flera olika sätt av autentisering[4], plus att den har nu stöd för både TCP och UDP. Förra version hade bara stöd för TCP och den hade heller ingen autentisering. Socks4 är på väg att ersättas av Socks5 då denna är bättre. Med denna typ av brandvägg kan man även sätta upp accessregler som baserar sig på användare, applikation, tid på dygnet, och så naturligtvis beroende av vart paketet ska, och var det kom från.

[4]Verifiering av användare

3.5 Paketfilter-lösningar

Paketfiltrering är ett sätt att titta i headern på de paket som kommer och avgöra om man vill släppa in dom eller inte. Ett exempel på hur ett paket ser ut kan du se i figur 3.4.

Figur 3.4: Ett paket med headerfält och datafält

Det finns tre sätt att hantera paketen på. Man kan välja mellan DENY, REJECT eller ACCEPT. DENY innebär att man kastar paketet som kommer in utan att meddela avsändaren. REJECT gör samma sak förutom att den även skickar tillbaka ett svar till avsändaren av paketet och talar om att det inte fick komma igenom. ACCEPT innebär att paketet accepteras och släpps igenom brandväggen. Utöver detta finns ofta en hel del specialsaker man kan göra, men detta är grunden i paketfiltrering.

3.5.1 Varför vill man använda paketfiltrering?

Det finns tre orsaker som motiverar användandet av paketfiltrering, och de är följande: kontroll, säkerhet och övervakning.

Kontroll: Innebär att man kan bestämma vad som ska få passera. Om man släpper ut sitt lokala nät på Internet kan man välja att filtrera bort vissa adresser om man vill. Om man till exempel vet att från spammer.net kommer enbart onödig trafik kan man blockera denna adress med en regel så har man löst det problemet.

Säkerhet: Om allt som står mellan ditt lokala nätverk och Internet är just en brandvägg som använder paketfiltrering kan det vara praktiskt att kunna stänga ute viss trafik man inte vill ha in. För att försäkra sig mot ”ping of death”[5]-attacker kan man helt

[5]Man skickar ett för stort IP-paket vilket medför att måldatorn krashar

enkelt blockera denna typ av trafik i brandväggen. Om man inte vill släppa ut någon information på Internet kan man välja att blockera alla paket som försöker göra en uppkoppling utifrån, men samtidigt släppa igenom allt som går inifrån och ut till Internet.

Övervakning: Man kan med hjälp av loggar se om någon har försökt att bryta sig in i ens system, eller få det rapporterat till sig omedelbart då intrångförsöket upptäckts. Man kan dessutom välja vad man vill ha loggat och hur detta skall rapporteras.

4 Analys

Vi ska i det här kapitlet belysa hur vi valt ut de brandväggar som testats, och beskriva egenskaper hos dessa brandväggar. Därefter kommer vi att gå igenom resultaten av en grovtestning, för att sedan komma till en mer detaljerad testning av en eller ett par brandväggar som vi tycker varit intressantast av de utvalda.

4.1 Utgallring av brandväggar

Vi började vårt arbete med att söka efter information om vilka brandväggar som var bra på Deja-news[7], som är en hemsida där man kan ställa en publik fråga och sedan vänta på svar. Man kan även söka bland tidigare frågor och svar, vilka kan innehålla den information man söker. Detta gav oss en mängd alternativ på olika lösningar och vi började att besöka de hemsidor som blev rekomenderade i de svar vi fick. Vi hade nu ett 20-tal olika lösningar att välja bland och det första urvalet gjorde vi genom att gå igenom dessa rekommenderade lösningar och lät kriterier som säkerhet, dokumentation, finesser, utvecklingmöjlighet, rekommendationer från andra användare och enkelhet att administrera vara stöttestenar.

Vi upptäckte att de flesta lösningar byggde på samma princip, nämligen paketfiltrering, men för att även ha fler förslag valde vi att även ta med en brandvägg byggd på en proxylösning. De lösningar som nu återstod var alla av typen paketfiltrerande, avsedda att användas på olika typer av UNIX-kompatibla datorer. Samtliga dessa brandväggar följer även licensavtal enligt GNU-konceptet, det vill säga att de är helt gratis att använda. En fördel med att använda brandväggen på en UNIX-kompatibel dator, exempelvis Linux, är att uppdateringar av säkerhetskritiska komponenter har en tendens att vara mer frekventa. Dessutom är ofta dessa typer av operativsystem mer pålitliga med avseende på robusthet än vissa andra.

4.2 Presentation av brandväggar

De brandväggar som vi ansåg tillhöra de bästa enligt kriterierna i kapitel 4.1 och har valt att fortsätta att titta lite närmare på är:
- pmFirewall[1]
- Mason Firewall[2]
- IP-Filter[3]
- Falcon Proxy[4]
- SINUS Firewall[5]
- CoyoteLinux Firewall[6]

4.2.1 pmFirewall

Detta är en paketfiltrerande brandvägg för Linux, som stödjer 'masqurading'. För att använda en Linuxdator som paketfiltrerande brandvägg används programmet IPChains, som finns inbyggt i Linux kärna[6]. Den är gjord så att en person med liten eller ingen vana av IPChains skall kunna få den att fungera. Den är transparent för användarna och har följande finesser:

- Autodetekterar IP-address och nätmask på varje interface.
  Detta innebär att den vid installationen automatiskt känner av den IP-adress och nätmask som systemet skall använda.

- Blockerar NetBIOS, NetBUS, Back Orifice och Samba.
  Detta är program som ofta används för att attackera en dator.

- Skyddar mot 'IP-spoofing'.

- Loggar alla DENIED paket.

[6]Hjärtat i Linux operativsystem

De paket som av någon anledning inte får färdas in i det skyddade nätverket kommer att loggas så att administratören av brandväggen lätt kan se vem som försökte göra vad.

- Masquradingstöd väljs under installationen
- Man kan lägga till egna regler i en fil.
  Trots att denna brandvägg skapar de regler man behöver kan det tänkas att man vid ett senare tillfälle vill lägga till eller ändra i de regler som skapades vid konfigurationen. Det finns då en fil dessa ändringar kan göras i.

Brandväggen pmFirewall kan köras på en mängd olika distributioner av Linux, däribland: RedHat, Mandrake, OpenLinux, Debian, SuSE och Slackware. Man kan även anmäla sig till en mailinglista för att få information om uppgraderingar och andra diskussioner om projektet.

4.2.2 Mason Firewall

Mason har ett lite annorlunda koncept, och det är därför vi valt att kika närmare på den. Det som är så speciellt med den är att den ska tydligen kunna lära sig saker själv! Grundstenen i denna brandvägg är IPChains. Inlärningen går till så att man sätter brandväggen i `ACCEPT`-mode, sedan gör man till exempel en telnetuppkoppling genom brandväggen, och när detta görs skapar Mason en regel som tillåter detta. Vill man blockera något sätter man brandväggen i `REJECT`-mode eller `DENY`-mode och gör det som inte ska gå igenom. Efter en omstart av Mason så ska reglerna finnas där. Detta är i alla fall grundtanken bakom Mason, som även har följande egenskaper:

- Man kan blanda IPChains och ipfwadm, som är föregångaren till IPChains.
- Den behöver inte köras som root, även om det innebär vissa fördelar.
- Man kan använda alla interface som klarar av TCP/IP.

- Den känner igen alla protokoll som finns i */etc/services*.
- Den hanterar automatiskt 'masqurading', och de regler som detta innebär.
- Alla regler självlärningsprocessen genererar har en kommentar så man ska veta vad den gör.
- Man kan ändra reglerna utan att starta om brandväggen. Detta är bra eftersom man inte skyddar sig under tiden brandväggen startas om.

4.2.3 IP Filter

Denna brandväggslösning har mest testats på NetBSD och FreeBSD, men det sägs dock på hemsidan att den även skall fungera under: Solaris, SunOS 4.1.1 - 4.1.4, BSD/OS-1.1 V4, IRIX 6.2 och Linux 2.0.31-35.

Denna brandvägg är som de flesta andra av typen paketfiltrerande. Den kan enligt bifogad dokumentation följande saker:

- Stoppa vilka paket man vill från att passera (självklart).
- Skydda mot 'IP-spoofing'. Läs mer om 'IP-spoofing' i kapitel 3.1.3
- Filtrera trafik från en viss host eller IP-block. IP-blocket behöver inte vara jämna /8, /16, /24 eller /32-block. Ett /8-block, som också kallas klass A nät, innebär att man har ett eget IP block som till exempel AAA.xxx.xxx.xxx där man kan bestämma xxx själv. Har man ett /16-block, eller klass B nät, har man AAA.BBB.xxx.xxx. Ett /24-block, eller klass C nät, så har man AAA.BBB.CCC.xxx. Slutligen har vi /32-block, eller en enskild host, alltså alla siffror är bestämda AAA.BBB.CCC.DDD.
- Filtrera bort vissa protokoll, till exempel ICMP.
- Skicka tillbaka ICMP error/TCP-reset på paket man filtrerat bort.

- Följa paketströmmar för att släppa igenom paket som tillhör en redan öppen förbindelse.
- Uppföra sig som en NAT.
- Omdirigera trafik, exempelvis skicka alla uppkopplingar inifrån till en viss IP-adress, till exempel en proxy.
- Inspektera pakethuvudet med externa program.
- Filtrera TCP/UDP paket inom ett visst port-intervall eller enskilda portar.
- Filtrera ICMP paket med avseende på vilken typ av ICMP-paket det är.
- Blockera TCP-paket med exempelvis SYN-flaggan satt. Detta för att förhindra alla former av uppkopplingar mot det lokala nätverket.
- Filtrera bort trasiga IP-paket.

4.2.4 Falcon Proxy

Detta är den enda proxylösningen vi valt att eventuellt testa. Den bygger på moduler som man använder då man vill ha stöd för något protokoll, till exempel HTTP, FTP, TELNET eller SMTP. Falcon proxy är open source och är skriven i Perl. Anledningen till att vi valde denna är att den andra proxylösningen vi tittade på, Squid, ej hade stöd för SMTP, POP3 och andra saker vi behöver. Falcon har några egenskaper vi vill belysa.

- Falcon är enligt manualen en lättviktare. De menar då att den inte har massor av parametrar och konfigurationsfiler man ska kunna utantill, utan kräva minimalt med manuell konfigurering.
- Den är modulär. Det vill säga att den är uppbyggd av moduler, så ska man ha en telnetproxy lägger man till en telnetmodul och så vidare.

Efter ett möte med uppdragsgivarna valdes denna proxylösning bort. Anledningen till detta var dels att den är är skriven i Perl, vilket innebär att brandväggsdatorn skulle ha fått vara en både kraftig och dyr maskin för att kunna klara av att dirigera trafik från såpass många datorer som det i detta fall skulle bli, och dels att om någon nödvändig modul till brandväggen saknades skulle denna typ av trafik ej kunnat användas.

4.2.5 Sinus Firewall

Detta är en paketfiltrerande IPChains lösning för Linux som är en vidareutveckling av sf-Firewall som funnits till Linux ganska länge. Den har ett gott stöd för övervakning av brandväggen och den konfigureras via en textbaserad prompt, men det finns en grafisk modul också. Vi valde denna brandvägg bland annat eftersom vi gärna ville testa en grafisk brandvägg också. Loggar kan mailas till administratören. På hemsidan finns mycket dokumentation att läsa, bland annat så finns det en testrapport som kan vara lärorik. Sinus klarar dock inte 'masqurading', men den har följande egenskaper:

- Filtrering av headrar i IP,TCP, UDP, ICMP och IGMP paket.
- Intelligent RIP och FTP support.
- Textbaserad konfiguration.
- Grafisk interface för konfigurering av flera brandväggar.
- Dynamiska regler med räknare och 'time-outs'.
- Stora loggningsmöjligheter, varningar, och intelligenta räknare.
- Skyddar mot 'IP-spoofing'.

4.2.6 CoyoteLinux Firewall

CoyoteLinux är en mycket annorluda lösning av brandvägg. I det här fallet installerar man programmet på sin dator och startar ett script. Detta script ber användaren att mata in de

regler man önskar för brandväggen, och dessa data sparas på en diskett, tillsammans med en Linux-kärna. När konfigureringen är klar kan man, om man vill, ta bort hårddiskar och andra lagringsmedia från brandväggsdatorn, skrivskydda disketten och starta upp datorn från disketten. Detta innebär att man får ett system som blir mycket svårt att attackera eftersom den som utför attacken inte får någonstans att lagra information på.

Denna brandvägg har sitt ursprung i LRP. Skillnaden mellan LRP och CoyoteLinux är sättet som man konfigurerar och underhåller brandväggen på. Den har för närvarande följande egenskaper.

- Använder sig av Linux kärnan 2.2.x.
- Stöd för DHCP och IPChains.
- Liten och kompakt, kör inte igång massor av onödiga tjänster som kan vara fallet med stora Linuxdistrubitioner.

CoyoteLinux finns dels som gratisprogram (freeware) och dels som kommersiellt program. Skillnaden mellan gratisdistributionen och den kommersiella är sättet som denna diskett skapas på. I den kommersiella versionen leds man igenom en grafisk Wizard under Microsoft Windows som skapar disketten, medan man i gratisdistributionen svarar på frågor i ett textbaserat script under Linux. I övrigt finns ingen skillnad mellan distributionerna.

Den kommersiella versionen på CD kostar 50 dollar. Det finns även en mailinglista för CoyoteLinux som man kan anmäla sig till på deras hemsida.

4.3 Allmänt om testning

Testningen av brandväggarna var svårare än vi från början trodde. Svårigheten låg i att veta hur vi skulle skulle testa dom, och om vi testat dom tillräckligt bra. När det kommer

till ämnet datasäkerhet kan man aldrig vara helt säker. Hackers[7] kommer alltid att hitta nya sätt att bryta sig in i system på. Och som sagt, hur ska man testa? Det är här vårt största problem ligger. Ingen av oss har någon erfarenhet av detta så vi har sökt information på Internet, och de vanligaste rekommendationerna är att man kör program som NMap och SAINT, vilka förklaras i appendix B1.

Dock anser vi inte att detta verkar vara tillräckligt. Dessa program scannar en dator och ser vilka portar som är öppna[8]. Därefter testar programmen kända säkerhetshål och så vidare. Målet med brandväggen är att den inte ska ha några onödiga portar öppna då detta försvårar att någon kan bryta sig in i den. Dessa program testar bara den dator som har brandväggsprogrammet installerat, de försöker inte penetrera den och komma åt de datorer som ligger innanför. Visserligen är det i vårt fall omöjligt att komma åt datorerna innanför brandväggen, då de använder interna IP-adresser. I och med detta är det ingen router[9] på Internet som routar denna typ av IP-adresser. Det enda sättet att komma åt datorerna innanför brandväggen är att först hacka brandväggen och därefter hacka sig vidare till de interna datorerna.

4.4 Grovtestning av brandväggar

Det första testet av dessa brandväggar är ett grovtest vars syfte är att snabbt gallra ut vissa brandväggar som vi inte anser vara tillräckligt bra för att testa vidare. Anledningen till detta är att vi inte i onödan skall hårdtesta de brandväggslösningar som vi redan vid första anblicken anser generera otillräckligt eller överflödigt med regler, samt att gallra bort brandväggar som på annat sätt är sådana att vi inte anser att de är bra.

[7]Någon som vill bryta sig in i ett system
[8]Dörrar in mot en dator som intrång kan ske igenom
[9]Dator på Internet vars uppgift är att dirigera om trafik

4.4.1 pmFirewall

Denna brandvägg är enkel att installera, man startar ett script och svarar helt enkelt ja eller nej på ett antal frågor. När detta är klart startar man upp brandväggen. Datorn innanför kom ut på Internet mycket snabbt. Dock har denna brandvägg två stora brister. De regler som skapas är inte bra. Den första regeln som pmFirewall sätter upp tillåter all trafik mot det interna nätverket, och efterssom detta är tillåtet, kontrolleras inga andra regler. Den andra stora nackdelen är att pmFirewall ej stödjer tre nätverkskort. Detta innebär att vi ej kan använda oss av en DMZ, vilket var ett krav från uppdragsgivarens sida.

4.4.2 Mason Firewall

Vid första testet av Mason genererade den hur mycket regler som helst, det blev till slut helt omöjliga att kontrollera. Nu när vi testat den en andra gång genererades inte några regler alls, så vi är föga imponerade hittils. Det var dock inga problem att få 'masqurading' att fungera, och de är ju alltid ett steg i rätt riktning. Datorerna på insidan av brandväggen kom ut på Internet fort, men självlärningen är det som sagt sämre med, om man nu behöver så mycket regler när man har interna IP-adresser på sitt LAN. Att lägga till regler i Mason är dock ganska lätt, om man kan IPChains-syntaxen, man lägger bara till det man vill ha i en speciell fil så är det klart sen. Eftersom Mason genererade mycket oregelbundna regler valdes även denna brandväggslösning bort.

4.4.3 IP Filter

IP Filter kunde ej installeras på den dator vi ska köra brandväggen på. Enligt dokumentationen kan man köra denna brandvägg under Linux 2.0.31, och vi hade hoppats på att den även skulle vara kompatibel med nyare versioner av Linux, men så var ej fallet. Därför kommer vi ej att testa denna brandvägg vidare då vi har beslutat oss för att använda Linux 2.2 som operativsystem för brandväggen.

4.4.4 Sinus Firewall

Det visade sig att denna brandvägg var uppbyggd av en server, som är själva brandväggen, och en klient som man kan konfigurera servern med. Klienten var dock skriven i Java och den ville inte kompilera på den JDK vi installerade. I manualen står det att Sinus Firewall kräver JDK 1.1.6 och Swing, men eftersom man egentligen inte behöver ha klientdelen har vi skippat den tills vidare. Vid en första anblick på reglerna ser syntaxen ut att vara ganska lätt. Man skriver mer som vanlig text vad regeln ska göra istället för IPChains-syntaxen med massor av parametrar och så vidare, vilket gör IPChains ganska svårläst för ett otränat öga. Kompileringen gick fint av serverdelen, men som sades ovan fick vi inte klientdelen att kompilera som den skulle. Det som gör att serverdelen gick igenom utan problem är att denna är skriven i C och inte i Java som klientdelen är.

4.4.5 CoyoteLinux Firewall

Efter att ha kört scriptet som skapar en bootbar diskett startades datorn upp ifrån denna. Programmet har en inbyggd konfigureringsmeny där inställningar kan göras om man i efterhand vill ändra några inställningar man gjorde vid installationen, eller om situationen förändrats och man behöver ändra något. För att denna brandväggslösning skall fungera måste paketet `mtools` finnas installerat på datorn, eftersom detta program krävs för att skapa disketten. Man behöver dessutom lägga till stöd för `msdos` i kärnan.

CoyoteLinux fungerade bra, datorn innanför brandväggen kom omedelbart upp på nätverket, men däremot stödjer inte CoyoteLinux tre eller fler nätverkskort, vilken i praktiken innebär att vi ej kan använda oss av en DMZ. Möjligheten att lägga till egna regler är dessutom krånglig och det finns inget enkelt sätt att göra detta. Man kan dock avsluta programmet och skriva egna regler med hjälp av IPChains.

4.5 Testresultat

Ursprungligen hade vi tänkt att efter att ha grovtestat de olika brandväggslösningarna skulle vi genomföra ett hårdtest av dessa för att verkligen se hur bra de var. Nu blev detta inte fallet då ingen av de brandväggar vi valt hade det stöd vi krävde av dom. De flesta gick att installera och köra utan problem, men däremot var det nästan ingen av de brandväggar vi valde att grovtesta som hade stöd för tre eller fler nätverkskort, det vill säga att till exempel använda en DMZ. När vi insåg detta provade vi med att göra en egen lösning som skulle klara just de saker vi krävde av brandväggen.

5 Egen lösning

Det finns flera olika sätt att skapa en egen brandvägg på. De flesta använder någon typ av scriptspråk, exempelvis Perl, som ställer frågor angående vilken typ av trafik man vill släppa igenom. Denna metod använde även vi. Vi började med att skriva ett litet script som skapade ett fåtal regler för att testa oss fram. Dessa regler sparades i en fil och det bör nämnas att denna fil är en vanlig textfil innehållade kommandon som utförs i tur och ordning då filen exekveras. Ett exempel på en sådan regel är: *ipchains -A input -p tcp -s $source_address/24 -d 0/0 21 -j ACCEPT*. Denna regel tillåter TCP-trafik på port 21 från det interna nätverket. Vi kommer inte att förklara IPChains-syntaxen mer ingående, detta illustrerar bara hur en IPChains-regel kan se ut. Vi har även använt oss av 'portforwarding', och en sådan regel kan se ut så här: *ipmasqadm portfw -a -P tcp -L $source_address 53 -R $destination_address 53*. Denna regel säger åt Linux att all trafik som kommer på port 53 skall skickas till datorn med IP-adress *destination_address*.

För att automatisera och göra vår brandvägg mer generell valde vi att skapa ett Perl-script som ställer frågor om vilken trafik som skall tillåtas. Nedan följer ett urdrag ur vår brandväggslösning, som vi valt att ge namnet *cowWall*. Urdraget beskriver den del som frågar användaren om denne har en webserver, och i sådana fall vilken IP-adress webservern har. Om en sådan server finns kommer en portforwarding-regel att skapas.

```
sub forward_ask_webserver() {
  print "Do you have a WWW-server        [y/n]? ";
  if($svar eq 'y') {
    print "Enter IP-address:              [aaa.bbb.ccc.ddd]: ";
    $web_ip = <STDIN>;
    printf UT "\n# Forwarding Web-traffic to $web_ip\n".
      "ipmasqadm portfw -a -P tcp -L $source_address 80 -R $www_ip 80\n".
```

```
      "ipmasqadm portfw -a -P udp -L $source_address 80 -R $www_ip 80\n";
  }
}
```

Denna fråga genererar två stycken portforwarding-regler, som tillsammans med alla andra regler skrivs till en fil. Filen körs och man har en fungerande brandvägg. En stor fördel med att använda denna typ av brandvägg är att administratören har full kontroll på vilken trafik som tillåts i nätverket, då de övriga lösningarna genererade regler som ej alltid var helt tillfredsställande.

Denna brandväggslösning har följande karakteristik:

- Man får en god kontroll över de regler som skapas.
- Reglarna som skapas är direkt anpassade efter den miljön som brandväggen skall användas i.
- Skyddar mot 'IP-spoofing'.
- Loggar alla paket som ej tillåts.
- Möjlighet att filtrera bort ICMP-paket.
- Blockerar NetBus och Back Orifice.
- Fullt stöd för 'masquerading'.
  Detta gör att direktaddressering av datorer på insidan brandväggen ej blir möjlig då de använder interna IP-addresser.

Om detta är den bästa lösningen vet vi inte, men den är i alla fall optimerad efter vårt önskemål. Det är möjligt att det finns någon liknande brandvägg på Internet, men att hitta en sådan är inte så lätt som man kan tro då det finns en mängd brandväggar och

det är omöjligt att testa alla. Nu menar vi inte att detta inte är en bra brandvägg. Denna brandvägg är mycket flexibel som till skillnad från många andra inte genererar regler "i det dolda" som en administratör inte har koll på. Alla regler som skapas görs på order av administratören, det enda övriga som genereras är blockerande regler. Detta ger en kontroll av reglerna som skapas som inte återfinns i någon av de andra brandväggarna vi testade.

6 Slutsatser

Vad kan vi nu dra för slutsatser av detta arbete? Det första man märkte var att det finns verkligen hur många brandväggar som helst där ute, många av dom var dock inte aktuella för oss. Dels för att vi bara kikade på gratislösningar, och sedan verkade en paketfiltrerande lösning bäst för oss. Detta gjorde att en hel del andra brandväggar gick bort, vi testade inte ens den proxylösning vi hade valt ut för test då arbetsgivaren strök denna direkt. Att hitta en brandvägg som är värd att testa kan ta ganska lång tid, och vilken man än väljer finns det troligen en som passar en bättre någon annanstans på Internet. Att nästan ingen av de brandväggar vi installerat och tittat närmare på har stöd för fler än två nätverkskort är lite underligt då en systemadministratör borde vilja ha en DMZ till sin brandvägg, i alla fall de som erbjuder någon tjänst utåt som till exempel WWW, FTP eller DNS.

En annan sak som kan vara bra att tänka på är vad man vill att brandväggen ska klara av innan man ger sig ut och letar. Detta är inte alltid helt självklart och vi spenderade en del tid på att diskutera vad vi egentligen ville släppa ut från DMZ'n och vad vi skulle tillåta mellan DMZ och vårt LAN. Då vi gör jobbet på ett universitet vet vi att man inte kan lita på användare som sitter på LAN, och bör därför ha vissa restriktioner mellan LAN och DMZ.

Eftersom vi valde att göra en egen brandvägg medförde detta att vi fick en större inblick i datasäkerhet. Dels blev vi tvugna att lära oss hur IPChains fungerar, och dels många av de olika typer av trafik som kan komma. Utöver detta lärde vi oss även hur ett brandväggs script fungerar, samt att vi fick lära oss lite bättre att programmera i Perl.

Dessvärre närmade vi oss 'deadline' för projektet. Om vi hade haft mer tid på oss skulle vi gärna vilja förbättra vår brandvägg cowWall. Exempel på förbättringar är:

- Automatisk detektering av IP-adresser och nätmask.
- Förbättra GUI, kanske även grafiskt.
- Öka läsbarheten i loggarna.
- Generering av brandväggs-scriptet via WWW.

Trots att vi har största förtroende för cowWall är den fortfarande i sin testfas, och vi vet ännu inte exakt hur den kommer att fungera då den körs skarpt, och levererar trafik åt cirka 200 datorer. Dock är det vår övertygelse att cowWall kommer att lösa sin uppgift mycket bra.

Referenser

[1] *pmFirewall* `http://www.pointman.org/PMFirewall/` Maj 2000

[2] *Mason Firewall* `http://users.dhp.com/ whisper/mason/` Maj 2000

[3] *IP-Filter* `http://coombs.anu.edu.au/ avalon/` Maj 2000

[4] *Falcon Proxy* `http://falcon.naw.de/` Maj 2000

[5] *Sinus Firewall* `http://www.ifi.unizh.ch/ikm/SINUS/firewall/` Maj 2000

[6] *CoyoteLinux* `http://www.coyotelinux.com/` Maj 2000

[7] *Deja News* `http://www.deja.com/` Maj 2000

[8] *Saint* `http://wwdsilx.wwdsi.com/saint` Maj 2000

[9] *NMap* `http://www.insecure.org/nmap` Maj 2000

[10] *Strobe* `http://www.rootshell.com` Maj 2000

[11] *Shields-Up!* `https://grc.com/default.htm` Maj 2000

A Installationsguide

Denna installationsguide ska steg för steg visa hur man installerar brandväggen på ett RedHat-system. Den konfiguration vi haft är som följer.

CPU	PII@233MHz
Minne	160MB Ram
HD	Adaptec 7880 SCSI-kontroller med 2GB SCSI hårddisk
Nätverk	Två stycken 3Com 3b905B och ett 3Com 3b509B

Den del som beskriver hur man konfigurerar kärnan kommer grundas på denna hårdvara. Vad gäller att säkra operativsystemet är detta grundat på RedHat 6.1 och dess konfigurationsfiler. Detta kan vara bra att ha i åtanke om inte allt verkar stämma på ditt system.

A.1 Mjukvarukrav

Vad gäller mjukvara ska brandväggen köras på Linux. Kärnan ska vara av version 2.2 eller senare, dock är all testning gjord på version 2.2.14 så denna är att rekommendera, vidare behövs också programmet *ipmasqadm*, som krävs för portforwarding, och sist men inte minst stöd så man kan köra Perl-script då scriptet som genererar brandväggen är skrivet i PERL.

A.2 Linuxinstallation

Installera RedHat Linux så det fungerar som det ska. Glöm inte att kolla så alla nätverkskort fungerar och att de har rätt IP-adresser. Vi kommer att gå igenom hur man säkrar systemet senare. Det kan dock vara bra att skippa allt som har med X[10] att göra då detta ändå inte bör köras på brandväggen, och tar upp onödigt med plats på hårddisken.

[10]Fönsterhanteren i Linux

Installera heller inte saker ni inte kommer använda då en eventuell angripare kan ha användning för detta. Installera så lite som möjligt!

A.3 Konfigurering av kärnan

Nu ska vi konfigurera kärnan. Detta gör vi för att det är vissa saker vi måste ha stöd för, och vi vill ha en optimerad kärna som inte innehåller onödiga saker som tar upp resurser och som kan innehålla säkerhetshål. Vi kommer visa hur vi har konfigurerat vår kärna, och det som kan skilja är stöd för nätverkskort. Detta för att ni kanske har något annat märke eller modell på de nätverkskort ni använder, men det borde ge en fingervisning om hur man konfigurerar kärnan för rätt nätverkskort i alla fall.

Det första man ska göra är att hämta hem den kärnan man vill ha, detta görs lämpligen från *ftp://ftp.sunet.se/pub/Linux/kernels/* eller från ett annat pålitligt ställe. Ta hem den senaste 2.2 kärnan, dock bör du minst ta 2.2.14, som när denna rapport skrevs är den senaste kärnan. Om du har en katalog som heter */usr/src/linux* kan det vara bra att döpa om denna till */usr/src/linux.old* eller liknande så den finns kvar om det blir strul. Packa sedan upp kärnan så den ligger i */usr/src/*, den kommer då lägga sig som */usr/src/linux*. Gå sedan till den katalogen och kör *make menuconfig*, detta kommer att ge dig en meny där man kan göra själva konfigureringen.

Vi ska nu gå igenom hur vi har konfigurerat vår kärna. Vi kommer även tala om de saker man **måste** ha med.

- *Code maturity level options* här lägger vi till *Prompt for development and/or incomplete code/drivers*, detta är en sak ni måste ha med.

- *Processor type and features* här är det bara att fylla i vad ni har. Detta spelar inte in på själva brandväggen.

- *Loadable module support* har vi inte med då vi inte har några kärnmoduler.
- *General setup* kan lämnas som den är, om ni inte har något eget ni vill ändra. Den har inget med brandväggen att göra.
- *Plug and Play* har vi inget stöd för.
- *Block devices* har vi bara stöd för *Normal PC floppy disk support* i då vårat system har SCSI.
- *Networking options* är den stora biten. Här ska vi lägga till saker så vi kan använda IPChains och portforwarding. Vid problem finns alltid *Help* att tillgå. Detta är saker du måste lägga till.
  Packet socket.
  Kernel/User netlink socket.
  Network firewalls.
  Unix domain sockets.
  TCP/IP networking.
  IP: firewalling.
  IP: masquerading.
  IP: masquerading special modules support. Krävs för att vi ska kunna välja portforwarding.
  IP: ipportfw masq support (EXPERIMENTAL). Detta måste vi ha för att portforwarding ska fungera.
  IP: TCP syncookie support (not enabled per default). Skydd mot SYN-flooding attacker.
  IP: Allow large windows (not recommended if <16Mb of memory).

  Nu är den tunga biten klar. Det som följer är sånt som är speciellt för den hårdvara man har. Så här kommer det troligen skilja sig en del.

- *Telephony Support* har vi inte med.
- *SCSI support* har vi med eftersom vi har en SCSI-kontroller, så har ni de gå in och välj rätt modell.
- *Network device support*, här talar man om vad man har för nätverkskort. Vi har valt våra 3Com-kort plus att vi har med *Network device support*.
- *Amateur Radio support* har vi inte med, av naturliga skäl.
- *IrDA subsystem support* har vi inte med.
- *ISDN subsystem* har vi inte med, vi har en fast uppkoppling mot Internet.
- *Old CD-ROM drivers (not SCSI, not IDE)* har vi inte med eftersom vi inte har någon sådan hårdvara.
- *Character devices*, här har vi med *Virtual terminal*, för att man ska kunna logga in med flera konsoler samtidigt på brandväggen (Alt+F<siffra>), och *Support for console on virtual terminal* som har med vart meddelande från kärnan hamnar. Till sist har vi stöd för våran mus.
- *Filesystems*, här har vi valt att ta med */proc filesystem support*, *Second extended fs support* och *ISO 9660 CDROM filesystem support* då vi har en CDROM på brandväggen som använts då vi installerat RedHat från CD.
- *Console drivers* har vi bara med *VGA text console* då vi inte behöver något annat.
- *Sound* har vi inte.
- *Kernel Hacking* behöver vi inte heller.

Nu när vi konfigurerat kärnan skall den kompileras, men först en sak om konfigureringen. När man konfigurerar kärnan ska man bara ta med sånt man vet att man behöver, och inte

sånt som kan vara roligt eller bra att ha. Detta för att minimera risken för säkerhetshål. Detta gäller dock brandväggsdatorer eller datorer som måste vara säkra.
Nu tillbaka till kompileringen. Kör bara *make dep&&make clean&&make zImage&& make modules&&make modules_install* och vänta, detta kan nämligen ta en bra stund beroende på vad man har för hårdvara. Om den skulle klaga på att kärnan blir för stor byt ut *zImage* mot *bzImage* så packas kärnan hårdare.

Make modules && make modules_install är med så den installerar eventuella moduler. Om inga moduler är valda kommer kompileringen att fungera ändå så det kan få vara kvar. Den kompilerade kärnan hamnar i */usr/src/linux/arch/i386/boot/* och heter *zImage* eller *bzImage*. Kopiera den till */boot/vmlinuz-<version>* och editera */etc/lilo.conf*, om ni har en sådan. Ett tips är att spara den gamla kärnan så man kan boota den om något går fel. Starta nu om datorn och den nya kärnan kommer att användas.

A.4 Konfigurering av operativsystem

Tänk på att detta gäller RedHat 6.1, så om ni inte använder det kanske inte filerna som nämns ligger på samma ställe. Syftet med att konfigurera operativsystemet är att se till att man täpper till alla tänkbara säkerhetshål som finns i operativsystemet. Det går huvudsakligen ut på att se till att det inte finns någon onödig service som körs på brandväggen. Det första vi ska göra är att se till att något sådant inte startas vid en eventuell omstart. Detta kommer vi göra genom att ta bort de filer som gör att de startas. Först börjar vi med att editera */etc/inetd.conf*. Detta är den fil som startar tjänster som finns på speciella portar, till exempel telnet, FTP och så vidare. Man kan i denna fil avmarkera allt genom att sätta '#' framför de rader som inte har det. Detta för att vi inte vill att någon ska kunna komma in på brandväggsdatorn om de inte har fysisk tillgång till den. En avmakering av raden som börjar med *telnetd* innebär att vi stänger ute möjligheten att upprätta en telnetuppkoppling mot brandväggen.

När detta är klart ska vi titta lite närmare på */etc/rc.d/*, som är den katalog där alla startfiler ligger. Vilken katalog man skall ta bort saker i kollar man genom att köra kommandot *runlevel* som då returnerar ett heltal som talar om vilken runlevel man kör för tillfället. I vårt fall är det runlevel 3, så vi kommer fixa till katalogen */etc/rc.d/rc3.d/*. I denna katalog lämnas bara ett minimalt antal startfiler kvar. Ni kan här ta bort allt utom följande filer: *K55routed, S10network, S20random, S30syslog, S40cron, S50inet, S75keytable, S99local.* Efter att fileran har tagits bort kan ni starta om datorn så att ändringarna tar effekt. Man kan även köra varje fil med parametern *stop* innan man tar bort den, om detta görs slipper man starta om datorn vilket kan vara bra ibland.

För att ta reda på om brandväggen är öppen, det vill säga att den lyssnar på någon port, kör man lämpligen kommandot *netstat -l*. Här bör det nu inte finnas nån rad som det står *LISTEN* efter. *LISTEN* talar om att datorn lyssnar på den porten, det vill säga att man har ett program som inväntar en uppkoppling där. Om ingen sådan rad finns bör datorn vara ganska säker mot intrång utifrån, då ingen port är öppen.

A.5 Brandväggs installation

Starta brandväggsscriptet genom att skriva <sökväg>/cowWall. En meny presenteras där du kan välja mellan att konfigurera en ny brandvägg, lägga till eller ta bort regler, eller lista de regler som redan finns. Första gången detta görs väljer du att konfigurera en ny brandvägg. Efter detta uppmanas du att mata in de nätverksadresser du vill ha. Först skall du mata in den externa IP-adressen, därefter nätverksadresserna dels till ditt interna LAN, och dels till din DMZ. Om dessa fält lämnas tomma antas default gälla. Kontrollera att du skrivit rätt, och ange i sådana fall 'y'.

Portforwarding

Du skall i denna del av installationen mata in de servrar som finns på insidan av brandväggen. Brandväggen cowWall kommer att fråga dig vilka servrar du har och be dig ange dess IP-adresser. Brandväggsprogrammet kommer att se till att trafik på dessa portar omdirigeras till rätt server. Om du vill att man skall kunna till exempel "pinga" brandväggen skall du tillåta ICMP-trafik, annars inte. Brandväggsscriptet frågar efter ett visst antal olika servrar, men om du har någon server som inte efterfrågas kan du addera denna genom att ange denna servers IP-adress samt vilken port som skall omdirigeras till den.

Regler mellan LAN och DMZ

Du skall nu ange vilka regler som skall gälla mellan det interna nätverket och DMZ. Tänk på att inte släppa igenom alltför mycket trafik här, efterssom det troligen finns minst lika många "hackers" på det interna nätverket som det finns utanför! Om du vill tillåta någon trafik som inte räknas upp anger du bara den port du vill släppa igenom.

Du har nu angivit allt som behövs och du kan starta brandväggen genom att skriva ./run_me, som är filen innehållande de regler du just angivet.

För att brandväggsdatorn omedelbart efter en omstart skall kunna erbjuda ett skydd krävs även att den automatiskt startas upp vid en omstart. För att göra detta ställ dig i katalogen */etc/rc.d/rc3.d/* och skriv: `ln -s <sökväg>/run_me S55cowWall`.

Övriga funktioner

Brandväggen cowWall har även ett par andra funktioner som kan vara bra att känna till. Du kan efter det att brandväggen blivit installerad lägga till regler i efterhand genom att starta cowWall och sedan välja 'Add additional firewall rules'. Du kommer då att få en ny meny presenterad för dig där du kan välja var i nätet du vill lägga till regler, in mot brandväggen eller mellan LAN och DMZ, samt om du vill tillåta extra portar eller om du

vill stänga ner några portar. Allt du behöver göra är att ange vilken port du vill modifiera och ändringen sker omedelbart.

A.6 Funktionalitetstest

Att testa funktionaliteten på brandväggen är ganska så enkelt. Vi antar dock att själva datorn är uppe, och att nätet fungerar på den. Detta funktionalitetstest går mer ut på att testa brandväggen än att testa operativsystemet. Efter det att ni installerat brandväggen är det bara att testa om ni kan göra de saker ni valt att man ska kunna göra. Om detta fungerar kan man också kontrollera att den stänger ute sånt som man inte vill ha, det vill säga allt utom de man valt att tillåta. Tänk på att se efter så att trafik fungerar både mellan LAN och DMZ, och mellan LAN och ut till Internet.

A.7 Underhåll

Underhåll på en brandvägg är viktigt, speciellt när det kommer till att säkra brandväggsdatorn. Se till att uppdatera kärnan om det släpps nya, och installera eventuella patchar som släpps till program du använder på datorn, då dessa patchar ofta täpper till säkerhetshål som hittats i det aktuella programmet. Något som kan vara bra att ha i åtanke är att man bör köra stabila versioner av de program man använder på brandväggsdatorn så långt som möjligt då dessa är mycket mer testade än betaversioner. Stabila versioner har troligen färre säkerhetshål. Detta inkluderar även kärnan man använder. Kör nya stabila versioner av den.

B Mall för testrapport

Testningen av alla utvalda brandväggar har gjorts enligt följande steg:

- Gör en bedömning av hur lätt den var att installera.
  Vi jämför de olika brandväggarna med varandra för att se hur lätta de är att installera. Syftet med detta är att en slutanvändare av brandväggen ej skall anse att installationsprocessen är krånglig. Detta kommer i sådana fall troligtvis att leda till att brandväggen inte kommer att användas. Dessutom, om två brandväggar har liknande funktioner kan installationsförfarandet avgöra.

- Kör en portscan med programmet Strobe och se vilka portar som är öppna.
  Vi kontrollerar hur säker datorn är, med avseende på vilka portar som är öppna. Syftet med detta är att se vilka tjänster en person som skall bryta sig in i datorn kan använda sig av.

- Kör SAINT, som är en variant av Satan.
  SAINT scannar portar ytterligare en gång och testar dessutom kända säkerhetshål på själva datorn, hur säker den vad gäller försök till intrång osv.

- Kör ShieldsUp![11]
  ShieldsUP! erbjuder en liknande tjänst som SAINT, men är en webbaserad version av programmet NMap.

- Försök få en överblick över reglerna brandväggs-scriptet skapat. Detta för att avgöra om den skapat onödiga regler.

- Testa hur lätt det är att blockera en viss typ av trafik, till exempel telnet-sessioner mot datorn.

- Testa hur lätt det är att tillåta en viss typ av trafik, till exempel telnet-sessioner mot datorn. Syftet med de två sistnämnda testerna är att avgöra vilken som är lättast att underhålla, vilket är ganska viktigt när det gäller brandväggar.

B.1 Testverktyg

Här kommer vi att beskriva lite om de testverktyg vi använt oss av för att kontrollera hur säkra de olika brandväggslösningarna är.

SAINT

SAINT[8], Security Administrator's Integrated Network Tool, bygger på programmet NMap, samt att det innehåller en del andra egenskaper. Man kan vid en scan ställa in hur kraftigt den skall scanna, det vill säga vilka typer av funktioner den skall använda.

NMap

NMap[9] är gjort för att scanna större nätverk med, men det fungerar även bra för enskilda datorer. Den är skapad enligt filosofin TMTOWTDI (There's More Than One Way To Do It). Detta är visserligen Perl's slogan, men enligt programmeraren av NMap gäller det också scanners. Det som skiljer NMap från andra port-scanners är att den innehåller flera olika funktioner. Ibland vill man kanske scanna något snabbt, ibland kanske man inte vill bli upptäckt (bra för att testa loggning på till exempel en brandvägg), och ibland kanske man vill specificera olika protokoll. För att göra detta behövs ofta flera olika scanner-program, men inte i fallet NMap, som klarar av att utföra flera olika typer av scans. Vi har inte tagit upp allt NMap kan göra, utan bara de vi tycker verkar viktigt för förståelsen av vad NMap är för typ av program.

Strobe

Det Strobe[10] gör är att den försöker koppla upp sig till alla portar och ser vilka som svarar. På så sätt kan man få reda på vilka sorters tjänster det är tillåtet att göra på datorn i fråga. Denna information kan användas av en person som vill bryta sig in i systemet för att denne skall veta vilken typ av attacker denne kan göra mot datorn.