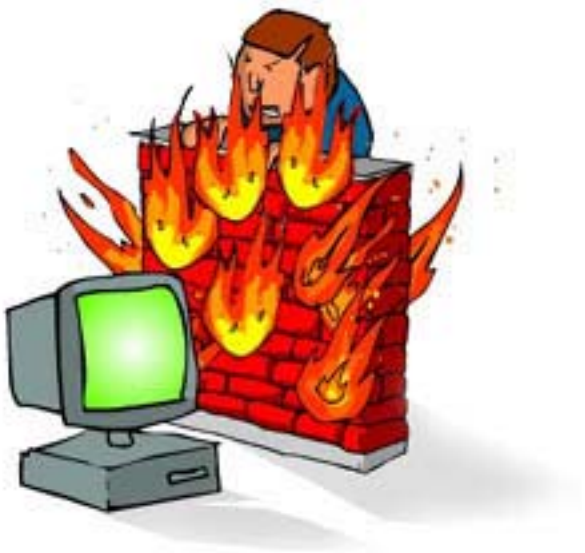
Opsætning af Firewall

- Konfiguration af IP-tables i Debian



Udarbejdet af Kenneth Dalbjerg og Ole Rosengreen ITE 2.1

Erhvervsakademi
Midtjylland

Lillelundvej 29 ▪ DK-7400 Herning Telefon +45 97 12 52 00

Titel: Netværk
Tema: Firewall
Projektperiode: uge 44 - 46 2003

IT projekt-

ITE 3. Semester

- Opsætning af Firewall

Vejleder: Benny Dyhr Thomsen

Side antal: 32 sider
Afsluttet: 13 November 2003

Udarbejdet af: Kenneth Dalbjerg & Ole Brix Rosengreen

Forord

Som udgangspunkt i vores projekt, går vi ud fra at brugeren har en Debian-box og de dertil-krævede ressourcer. Derudover bør man som bruger have en basal viden omkring navigering og brug af Linux, fordi det følgende materiale ellers kan virke lidt komplekst.

Vi har i samarbejde med de øvrige studerende i ITE21 på Erhvervs Akademi Midtjylland, udarbejdet en case som skal gøre det lidt mere realistisk at opsætte rettigheder på Firewall'en. Derved har vi nogle faste retningslinier som afspejler en realistisk opsætning af en Firewall i en fiktiv virksomhed i dagens Danmark. Casen kan ses i bilag 1 på side 26.

Ellers er den følgende dokumentation udformet som en guide, hvor at de aspekter vi mener kræver en forklaring, har fået det.

Kenneth Dalbjerg

Ole B. Rosengreen

Formål

Formålet med dette projekt er først og fremmest at skabe et bedre kendskab til teorien omkring opsætningen af en Firewall vha. *Iptables*, samtidig med at det udføres i praksis.
Herunder kan f.eks. nævnes forwarding, åbning og lukning af porte. Adgang for forskellige IP'er og MAC-addresser skal også tages i betragtning sammen med "åbningstiden" på de forskellige porte. Vi kan jo f.eks. ikke have at virksomheds-nettet bliver belastet af ekstern FTP-traffik i arbejdstiden.
Det kan jo eksempelvis løses ved at lukke for FTP'en i arbejdstiden og så åbne efter fyraften, men det vender vi tilbage til senere.

Derudover skal det skabe et bedre overblik over hvilke rettigheder, de forskellige zoner i en virksomhed skal tildeles, for at skabe et sikkert netværk og stadig bibeholde funktionaliteten af de forskellige webbaserede applikationer.

Ofte opsætter man en Firewall sammen med en antivirus-applikation for at øge sikkerheden, men vi har valgt kun at fokusere på Firewall opsætningen. Derudover findes der groft sagt to løsninger indenfor Firewalls, nemlig hardware og software. Vi kunne desværre ikke fremskaffe en hardwarebaseret Firewall så vi arbejder kun med software, mere præcist *Iptables* på en Debian-box.

Indholdsfortegnelse

Indledning

Det samfund vi i dag lever i, er meget præget af sikkerhed og man er efterhånden nødt til at beskytte sig mod alt fra virus og Crackere, til de ansatte i sin egen virksomhed. Et antivirusprogram kan med daglige opdateringer opsnappe de fleste viraer, men kan ikke beskytte mod unødig indtrængen fra Crackere eller nysgerrige ansatte. Derfor er det nødvendig at opsætte en Firewall, til at lave de generelle restriktioner og rettigheder.
Ved at åbne og lukke for diverse porte på Firewall'en, kan man begrænse unødig og uønsket trafik, men det udelukker jo ikke indtrængen af viraer. Derfor opsætter man ofte en Firewall og et antivirusprogram til at virke som én enhed.
Inden vi begriber os ud i den helt store opsætning af Firewall'en, så er der lige nogle generelle begreber som skal introduceres.

Softwarebaserede Firewalls findes I mange afskygninger både til Windows, Linux og andre styresystemer. Vi har jo, som før nævnt, valgt at tage udgangspunkt I Linux-distributionen Debian. Der findes mange webbaserede applikationer til opsætning af Firewall i Linux, her kan blandt andet nævnes "Smoothwall" og "Astaro". De kan selvfølgelig også konfigureres direkte på serveren, men de ligger op til den grafiske webbaserede brugerflade. De er begge baseret på det integrerede program *Iptables* som er en fornyelse af det "gamle" *Ipchains.*

Når man skal sætte et netværk op i en virksomhed skal man fastsætte nogle faste retningslinier indenfor sikkerhed, restriktioner og rettigheder. Forskellige virksomheder har forskellige retningslinier, men man kan generelt dele et firmanetværk op i to grupper: Det private netværk og DMZ.

DMZ er en forkortelse af DeMilitarized Zone, ingenmandsland. Betegnelsen dækker normalt over det landområde der ligger mellem to landes grænser.
Indenfor netværk bruges begrebet altså i Firewall sammenhæng. Typisk ønsker man ikke at resten af Internettet kan se noget som helst af det man foretager sig på det indre netværk, men dette harmonerer jo ikke med det stigende behov for at kommunikere med omverdenen.
De fleste organisationer på nettet har et vist behov for at servicere www, DNS, FTP, E-mail og lignende tjenester
DMZ (De-Militær-Zone) Er altså en afskåret del af det samlede netværk, hvor diverse servere står og har separate rettigheder i forhold til f.eks. Internettet. Det private netværk indeholder alle de PC'ere som anvendes af de ansatte. De har normalt ikke adgang til DMZ, ud over de services som kører derpå f.eks. FTP og WEB.

For at skabe et bedre overblik over netværks-topologien har vi lavet en grafisk oversigt som kan ses i [1]bilag 2. Oversigten viser de forskellige rettigheder for de forskellige net, hvor grøn er de porte som er åbne og rød er lukket, jævnfør casen. Den blå farve er en lille feature som vi selv har tilføjet til opgaven, fordi vi mener at det har en vis relevans for netværkets belastning. Featuren består af funktionen "Time" som gør det muligt at bestemme "åbningstider" for diverse porte og forbindelser. Vi har så valgt at der ikke skal være åbent for firma-FTP'en udefra, i arbejdstiden, for at mindske trafikken på serveren.

[1] Se side 28

Teori omkring IP-tables.

Dette relativt korte teorikapitel er opdelt i to afsnit, nemlig begreber og parametre. Eftersom iptables ikke er et decideret program, men nærmere en funktion i en større helhed, er der ikke den helt store teori bag. Men alligevel er en forklaring af de forskellige egenskaber meget berettiget, da det kan være lidt svært at gennemskue sammenhængen for det utrænede øje.

Begreber:

Hvis man vælger at bruge iptables til sin Firewall, er der nogen vigtige begreber som det er vigtigt at kunne holde styr på. Begreberne er inden for 3 kategorier: kæder, tabeller og polices.

Kæder:

Som standard har man 3 kæder, INPUT; OUTPUT & FORWARD. Disse bruges til at inddele trafik efter om de kommer til maskinen, går gennem maskinen eller kommer ud af maskinen.

INPUT: Er trafik som kommer ind til maskinen.
OUTPUT: Er trafik som kommer ud af maskinen.
FORWARD: Er trafik som gå gennem serveren og videre.

Tabeller:

Vi bruger tabeller til at lave NAT med, det er nemlig i vores NAT-tabel at vi skal angive vores destinations- eller source NAT (DNAT / SNAT).
SNAT bruges til at dele Internettet med, hvor man kun har én offentlig IP-adresse, her vil NAT kunne bruges til at oversætter IP pakkerne, så de ser ud til at komme fra den offentlige IP-adresse. Det er her NAT-tabellen kommer ind i billedet, for det er her serveren gemmer hvilke klienter som har forespørgsler i gang på nettet.
DNAT bruges til at Forwarde IP'er den anden vej, altså fra intranettet ud til f.eks. Internettet.

Polices:

Det er her man bestemmer om pakkerne skal droppes, eller om de godt må komme igennem, der er 3 ting her man kan bruge, dette er ACCEPT, DROP & REJECT.

ACCEPT: Her må pakkerne komme igennem
DROP: Her bliver pakkerne smidt væk
REJECT: Her bliver pakkerne smidt væk, men der sendes også et svar retur med at pakkerne, ikke måtte komme igennem.

Man bruger normalt DROP, i stedet for REJECT, da man ikke ønsker at man på Internettet modtage svar fra de ting som ikke kommer igennem, da dette kan forbindes med en sikkerhedsbrist.

Parametre:

Her kommer lidt omkring de forskellige parameter, man kan bruge. Alle i forbindelse med iptables til brug af en Firewall. Der findes rigtigt mange flere, men vi se kun på dem som vi har brugt.

-A	Dette står for append, og denne bruger man til at tilføje noget til enden af sin iptables tabel.
-I	Dette står for insert, og denne bruger man til at tilføje noget til starten af sin iptables tabel.
-s	Source adressen. Bruges til at checke afsender adressen. Dette kan bl.a. bruges hvis man ønsker at en bestemt maskines IP adresse ikke skal have adgang til maskinen. Hvis source adressen angives til FORWARD kæden kan man blokere internet adgangen for en bruger, når ens egen maskine fungerer som gateway / Firewall. Man kan i stedet for at angive en IP adresse, kigge på hele subnets ved at skrive følgende: 172.16.0.0/16. Her fungere vores / som adskillelsen imellem IP og subnet. Tallet efter /'et angiver hvor mange subnet bits vi ønsker at checke på. I dette tilfælde er det 16, som giver en subnet maske på 255.255.0.0.
-d	Modsvarer –s, her kigges der på hvor pakkerne gå hen. Altså destination.
-P	Kæde standard action - Sætter standard action på kæden (normalt ACCEPT)
-t	Angiver hvilken tabel man ønsker at bruge. F.eks. nat, mangle og filter som er standard hvis der ikke er angivet andet.
-p	protokol, hvilket protokol bruger man, dette kan både være som et protokol nummer eller ved et alm. Navn, eks: tcp, udp og som nummer kan det være 47. (47 er Ipv6.)
--sport	Source port, hvis man ønsker at kigge på den port hvor trafikken kommer fra. For at bruge denne option, skal man også angive hvilket hvilket protokol man bruger med -p
--dport	Destinations port, hvis man ønsker at kigge på den port man sender trafik til. For at bruge denne option, skal man også angive hvilket hvilket protokol man bruger med -p
--timestart	Bruges sammen med --timestop, til at angive en tid, hvor man eks, kan åbne op for trafikken. For at kunne bruge –timestart skal man have patch-o-matic i sin kernel. [2]
--timestop	Bruges sammen med --timestart, til at angive en tid, hvor man eks, kan åbne op for trafikken.[3]
--days	Bruges til at angive hvilket dage, det --timestop & --timestart skal gælde for[3].
-j	Dette bestemmer hvad der skal ske med pakkerne som matcher de kriterier man har sat op. F.eks. ACCEPT, DROP og REJECT.
--to-destination	Bruges i forbindelse med -j DNAT, hvor man så kan sige at en port skal videresendes til en anden maskine bag Firewallen.
-D	Bruges til at slette en regel.
-m	Load module, bruges til at loade udvidelse moduler til iptables. Som f.eks. state, time osv.
-L	Viser alle de regler der er sat op.

[2]Se side 9 for installation af patch-o-matic

-L -t nat	Viser de NAT regler der er sat.
--mac-source	Bruges til angive en mac adresse i stedet for en ip adresse. Dette er lidt mere sikkert da det er noget svære at ændre sin mac adresse, end det er at ændre sin ip adresse. Dette kan kun bruges i forbindelse med PREROUTING, FORWARD & INPUT.
--icmp-type	Bruges til at angive hvilket typer ICMP pakker som må komme igennem. Du kan se hvilke typer der findes ved at skrive iptables iptables -p icmp -h

Vi vil nu prøve at sætte nogle nemme regler op, for at komme i gang med iptables,

```
iptables -A INPUT -j DROP
```

Her smider vi en regle ind i bunden af vores tabel med -A, vi tager alt som går ind til Firewallen med INPUT og vi smide dem væk med -j DROP.

```
iptables -A INPUT -s 10.0.0.1 -j DROP
```

Her gør vi det samme, men dog kun med de pakker som kommer fra 10.0.0.1.

```
iptables -A FORWARD -s 10.0.0.1 -d 212.54.64.170 -j DROP
```

Her sætter vi en regel ind i bunden af FORWARD kæden, som tager pakker som kommer fra 10.0.0.1 og skal til 212.54.64.170, og smider dem væk.

```
iptables -I INPUT –p icmp --icmp-type echo-reply -s 10.0.0.1 -d 192.168.1.16 -j DROP
```

Her sætter vi en regel ind i bunden af INPUT kæden, hvor vi kigger nærmer på ICMP, med -p ICMP og tager kun udgangspunkt i at det er pakker med ICMP, altså pakker med echo-reply i, som kommer fra 10.0.0.1 og skal sendes til 192.168.1.16, disse pakker smider vi væk.

Downloading af kernen og patch

Disse filer kan hentes direkte ned til serveren med **wget,** den bruges på denne måde

```
host:~# wget url
```

Følgende filer skal man have hentet og lagt ind i /usr/src/.

```
host:~# cd /usr/src
host:~# wget http://www.kernel.org/pub/linux/kernel/v2.4/linux-2.4.22.tar.bz2
host:~# wget http://www.netfilter.org/files/patch-o-matic-20030912.tar.bz2
```

Kompilering/patching af kernen

Vi starter med at pakke kernen ud, hvorefter vi laver et ”symlink” (en såkaldt genvej) til mappen ”linux” og det gøres på følgende måde:

```
host:~# tar jxfv linux-2.4.22.tar.gz
```

Så skal der laves et symlink som hedder /usr/src/linux der peger på /usr/src/linux-2.4.22/

```
host:~# ln -s /usr/src/linux-2.4.22 /usr/src/linux
```

Pak derefter **patch-o-matic-20030912.tar.bz2** ud, dette gøres med:

```
host:~# tar jxfv /usr/src/patch-o-matic-20030912.tar.bz2
```

Så skifter vi til biblioteket /usr/src/patch-o-matic

```
host:~# cd /usr/src/patch-o-matic
```

Og kører filen *runme* med ”ekstra” som parameter.

```
host:~#./runme eksta
```

Her spørger den så om kerne-biblioteket der som standard er /usr/src/linux og så trykker vi bare **Enter.** Herefter bliver man spurgt om man vil inkludere en masse patches og her vælger man (N) altså nej indtil man støder på en patch som hedder *time.patch* og trykker (y). Herefter trykker man **Enter** og fortsætter med at sige (N) til de sidste patches.
Grunden til at vi ikke siger ja til alle patchene er at vi ikke har haft tid til at teste om kernen er funktionsdygtig med alle patchene installeret. Nogle af patchene overskriver nemlig hinanden og kan derfor skabe konflikter.

Så! Nu skulle kernel-materialet være pakket ud og klar til anvendelse. Vi går ind i mappen /usr/src/linux.

```
host:~# cd /usr/src/linux
```

Nu skulle kernen være klar til at blive kompileret og det gøres med følgende kommando:

```
host:~# cd /usr/src/linux/
host:~# make menuconfig
```

Herefter skal du vælge de moduler du skal bruge på dit system, for at kunne bruge vores opsætning. Gå ind under:

Networking options -->

Og markér:

Network Packet Filtering (replaces ipchains)

Vælg herefter:

IP: Netfilter Configuration --->

Og markér:

IP tables support (required for filtering/masq/NAT) (New)

Herunder markeres alt.

Når man er færdig med at vælge indholdet i sin kerne afslutter man menuen og skriver følgende:

```
host:~# make-kpkg --revision=mitimage.1.0 kernel_image
```

Dette kan godt tage et stykke tid, så hvis man skal på tønden er det et godt tidspunkt nu.
Efter den er færdig med at kompilere skal du installere kernen.

```
host:~# cd ..
host:~# dpkg -i kernel-image-2.4.22_mitimage.1.0_i386.deb
```

Herefter genstarter du maskinen.

Generel opsætning/initialisering af Firewall'en

Under den generelle opsætningen af Firewall'en opsætter vi den sådan at man ikke kan forwarde pakker. Derved kommer der ikke konflikter med anden forwarding når man kører det, eller en hacker kan ikke hacke sig ind lige bestemt på det tidspunkt, hvor man kører det og dermed få nogen pakker igennem som vi senere spærre for.

Dette gør man ved at ændre i proc, som er en masse indstillinger for dit hardware i ens Linux box. Alle disse indstillinger er "små filer", disse filer indeholder alt lige fra CPU størrelse, til hvor meget trafik der er gået gennem netkortet. Og for at ændre i disse filer bruger man echo.

For at slår forwarding fra, skal man skrive:

```
echo 0 > /proc/sys/net/ipv4/ip_forward
```

Vi sætter ligeledes nogen variabler op, således at man ikke skal huske på de lange net. Man bare kan skrive $LAN_IP_NET i stedet for at skrive 10.0.0.0/24. Det andet siger også lidt mere end 10.0.0.0/24.

```
LAN_IP_NET='10.0.0.0/24'
DMZ_IP_NET='10.0.1.0/24'
WAN_IP='192.168.1.16'
```

Vi skal nu have nulstillet vores kæder, så de er friske og parate til at få noget nyt input. Derved kan man køre scriptet flere gange uden at tabellerne bliver gentaget hver gang.
Til dette bruger man -F som nulstiller kæderne, man kan dog ikke bare skrive iptables -F, dette skal man i nogen sammenhæng faktisk passe på med, i sær hvis man sidder over SSH, da den kobler alt netværket fra.
Man skal nemlig have aktiveret et par ting bagefter igen for at det virker, derfor skal det altid køres i forbindelse med Firewall-scriptet.

Vi nulstiller her alle standard tabellerne og da vi ikke selv opretter nogen i vores Firewall script skal disse derfor ikke nulstilles.

```
iptables -t nat -F POSTROUTING
iptables -t nat -F PREROUTING
iptables -t nat -F OUTPUT
iptables -F
```

Vi lukker derefter af for alt, således at vi skal åbne for det som vi skal have adgang til senere i Firewall'en. Dog må Firewall'en alt, så denne lukker vi ikke for.

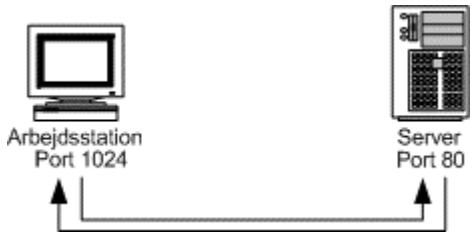
```
iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -P OUTPUT ACCEPT
```

Masquerading

Få at man kan få maskiner som sidder bag Firewall'en til at kunne køre videre ud på nettet, skal man enten gøre brug af masquerading, eller route IP'en videre. Vi benytter i vores Firewall masquerading.

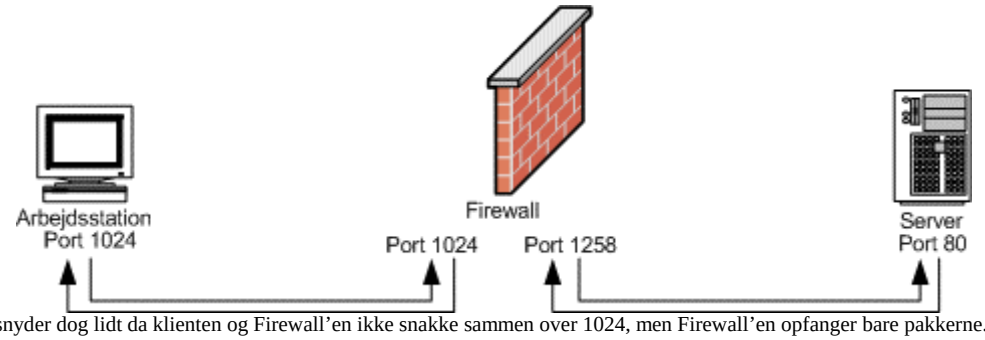
For at kunne forstår masquerade, skal man vide en smule omkring TCP/IP trafik. Det man i korte træk skal vide er følgende: Når f.eks. en webbrowser vil kontakte en webserver, så gør browseren det at den åbner en port som den lytter på, på maskinen, denne port er 1024 eller derover. Herefter vil den så forvente et svar retur på denne port. Den sender pakkerne, til serveren og vedhæfter porten.

Så en normal TCP/IP trafik vil se således ud:



Men når man har sat en Firewall, ind i mellem, så bliver Firewall'en nød til at ændre på disse porte, på grund af at klienterne ikke snakke sammen, om hvilket porte de bruger. Og således kan 2 maskiner derfor komme til at vælge den samme, port og så vil der ske problemer.

Derfor vil billedet være således



Billedet snyder dog lidt da klienten og Firewall'en ikke snakke sammen over 1024, men Firewall'en opfanger bare pakkerne.

Man opretter masquerade i nat "databasen", denne har 3 tabeller, og man bruger den som hedder POSTROUTING.
Derfor skriver man følgende:

```
iptables -t nat -A POSTROUTING -s $LAN_IP_NET -j MASQUERADE
iptables -t nat -A POSTROUTING -s $DMZ_IP_NET -j MASQUERADE
iptables -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT
```

Stateless

Vi har i vores Firewall lukket for næsten alle porte som kommer ind til Firewall'en og pakkerne vil derfor ikke kunne komme ind på de porte som klienterne i samarbejde med Firewall'en har åbnet op for. Derfor skal man bruge noget som hedder stateless masquerade. Dette gør det at den holder øje med hvilke pakker som går ud af Firewall'en, og hvilke som kommer retur. Og således åbner den op for de pakker som kommer retur fra en server af, som en klient har bedt om.

```
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
```

Opstart af Firewall'en

En Linux box vil altid starte netværket op som noget af det første i opstarten, dette bevirker også at Firewall'en vil være sårbar i nogle få minutter i opstarten og ligeledes når man lukker Firewall'en ned.

Derfor er vigtigt at et Firewall-script bliver startet rigtigt op. Dette gør man ved at fjerne "auto eth[enhed]" i sin **/etc/network/interface,** ud for hver af disse netværks enheder, man har i sin Firewall.

Og derefter laver man et script i **/etc/init.d**, denne kunne hedde "Firewall", indholdet af denne fil kunne f.eks. se således ud:

```
#! /bin/sh

case "$1" in
start|restart|force-reload)
        /usr/local/bin/firewall.sh
        ifup eth0
        ifup eth1
        ifup eth2
  ;;
firewall-stop)
        /usr/local/bin/firewall-stop.sh
  ;;

stop)
        ifdown eth0
        ifdown eth1
        ifdown eth2
        /usr/local/bin/firewall-stop.sh
  ;;
*)
  echo "Usage: /etc/init.d/ntpdate {firewall-stop|start|stop|restart|force-reloa
d}"
  exit 1
esac

exit 0
ildmur:/etc/init.d
```

Dette script gemmer man, og skriver:

```
host:# chmod 777 /etc/init.d/firewall
host:# update-rc.d firewall defaults
```

Og så skulle scriptet gerne starte op rigtigt hver gang Linux bootes, således at netværks-enhederne først bliver startet op efter Firewall'en er startet og at netværks enhederne bliver lukket ned før Firewall'en gør det.

Administrator rettigheder

For at gøre det nemmere for os, som administratorer, er alt trafik via SSH fra vores MAC-adresser til Firewall'en, tilladt. Dvs. at vi har adgang til Firewall'en lige meget hvor vi sidder på LAN. Vi kan også komme på via 192.168.1.16, altså Firewall'ens eksterne IP, men ikke hvis vi sidder på den anden side af routeren. Dette kan godt antages som en sikkerhedslempelse, men er desværre nødvendig hvis man skal have mulighed for at administrere Firewall'en fra skolens net.

```
iptables -A INPUT -p tcp --dport 22 -m mac --mac-source 00:0C:6E:21:D8:0A -j ACCEPT
iptables -A INPUT -p tcp --dport 22 -m mac --mac-source 00:00:E2:85:6C:E2 -j ACCEPT
```

Normalt vil man nok have konfigureret Firewall'en således det kun er LAN siden af Firewall'en, og muligvis også DMZ siden, som må administrere dette. Grunden der til er at det giver et sikkerhedsbrud at lade folk se at der er åben for port 22, på Firewall'en hvor man så derfra kan få hul til det private netværk.

Hvis man vil have adgang til Firewall'en fra Internettet skal man forwarde IP'en gennem routeren.

Vi gør i scriptet oven over brug af et modul ved navn mac, dette modul skal selvfølgeligt være kompileret med i kernen. Men det er nu ganske simpelt, efter det er loadet kan man skrive --mac-source efterfulgt af mac adressen. Ud fra dette kan man så tillade trafik eller ej, til bestemte porte.

Vi gør brug af MAC adresser, da disse er noget sværere at ændre end en IP adresse på et lokalt netværk. Dette virker kun hvis Firewall'en har direkte forbindelse til den maskine som man vil har accepteret eller blokeret. Da MAC-adressen vil blive ændret hver gang den kommer igennem en router, eller en anden enhed som ændrer på pakkerne, på samme måde som en router.

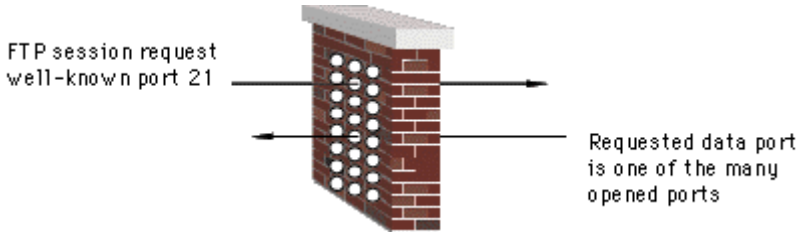
Privat netværk

Det private netværk har lukket alt trafik ud som standard og derfor skal man åbne op enkelt vis for de porte / programmer, som man skal bruge.

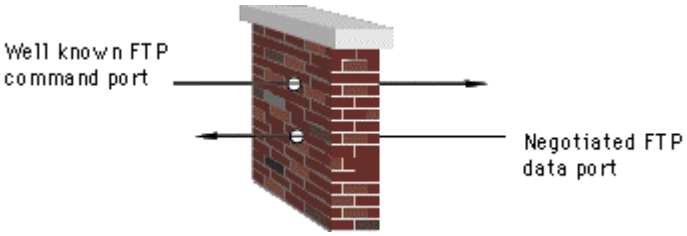
De programmer vi har brug for i vores case[3] er gennemgået herunder:
NB: Man kan se en grafisk oversigt over netværket i bilag 2.

FTP

FTP står for File Transfer Protocol, denne protocol bruger tcp og bruges til at overføre filer mellem 2 maskiner. FTP har 2 typer, aktiv og passiv FTP.
Ved passiv FTP tilslutter klienterne sig på port 21 og beder om at bruge en alternativ port, til at overføre.



Men når man bruger Firewall, så har man som regel ikke lige overført alle porte til sin ftp server, men kun port 21 og port 20, derfor bruger man noget der hedder aktiv ftp her. (Vi bruger aktiv ftp)



Vi sætter det op på følgende måde:

```
iptables -A FORWARD -p tcp -s $LAN_IP_NET -m multiport --dport 20,21 -j ACCEPT
```

[3] Se Bilag 1 side 26

SSH

SSH står for Secure Shell, og bruges normalt til at fjernstyre Linux Box hvor der kræves kryptering. SHH bruger TCP protokollen og kører på port 22.

Vi sætter det op på følgende måde:

```
iptables -A FORWARD -p tcp -s $LAN_IP_NET -d ! $DMZ_IP_NET --dport 22 -j ACCEPT
```

Protokollen TCP bliver accepteret fra sourcen $LAN_IP_NET , igennem Firewall'en og til port 22 på en given maskine på Internettet. Men ikke til $DMZ_IP_NET. Dvs. at det ikke er muligt at kontakte DMZ'en fra det private net.

SMTP

SMTP står for Simple Main Transfer Protocol og den bruger TCP protokollen via port 25. Når du sender mail bruger du SMTP, ligeledes når mail-serveren sender og modtager mail.

Vi sætter det op på følgende måde:

```
iptables -A FORWARD -p tcp -s $LAN_IP_NET -d ! $DMZ_IP_NET  --dport 25 -j ACCEPT
```

DNS

DNS står for Domain Name System og bruges til at oversætte navne til IP adresser. Når vi snakker om DNS opslag, altså hvor en klient spørger en DNS-server hvilket IP der er bag et domæne, bruger den UDP trafik, men hvis 2 DNS-servere skal snakke sammen og udveksle Domæne opsætninger, så bruger de TCP trafik, det vil dog begge steder være port 53 de bruger.

Vi sætter det op på følgende måde:

```
iptables -A FORWARD -p udp -s $LAN_IP_NET -d ! $DMZ_IP_NET  --dport 53 -j ACCEPT
```

HTTP

Http står for Hypertext Tranfer Protocol og er en applikationsprotokol, der indeholder et regelsæt for udveksling af filer over Internettet, denne protokol benytter TCP og port 80.
Her må det private net godt have lov til at kontakte webserveren i DMZ'en.
Vi sætter det op på følgende måde:

```
iptables -A FORWARD -p tcp -s $LAN_IP_NET --dport 80 -j ACCEPT
```

POP3

POP3 står for Post Office Protocol 3, denne protokol bruger du når du henter mail ned fra din udbyder. (Der kan i nogen tilfælde være brugt noget som hedder imap). POP3, bruger TCP og kører over port 110.

Vi sætter det op på følgende måde:

```
iptables -A FORWARD -p tcp -s $LAN_IP_NET -d ! $DMZ_IP_NET  --dport 110 -j ACCEPT
```

Samba

Samba kommer fra Linux verden, og er et navn som er kommet ud fra SMB (Server Message Block). Samba benyttes til Microsoft netværk, hvor man kan dele filer, og sende beskeder. Den benytter et par porte, nemlig 137,138,139, men dog er alle sammen TCP trafik.

Vi sætter det op på følgende måde:

```
iptables -A FORWARD -p tcp -s $LAN_IP_NET -d ! $DMZ_IP_NET -m multiport --dport
137,138,139 -j ACCEPT
```

HTTPS

Er http via SSL, det står for Hypertext Tranfers Protocol secure, og benyttes blandt andet af banker, til at sikre sig. Da alm. HTTP, ikke er krypteret vil alt udveksling af kodeord foregår ukrypteret, og en hacker kan derfor sniffer sig til pakkerne.

Vi sætter det op på følgende måde:

```
iptables -A FORWARD -p tcp -s $LAN_IP_NET -d ! $DMZ_IP_NET  --dport 443 -j ACCEPT
```

MSN Messenger

Nå man kommer over port 1024, så er det lidt mere forskelligt, hvilket porte de forskellige programmer bruger, dette hænger sammen med at der ikke rigtigt er nogen standard heroppe, og derfor kan man også vælge hvilke porte man vil bruge. Dog er det en god skik, ikke at bruge en port som en kendt protokol bruger.
MSN Messenger bruges til at sende beskeder og filer til hinanden med, denne bruger TCP og kører på port 1863

Vi sætter det op på følgende måde:

```
iptables -A FORWARD -p tcp -s $LAN_IP_NET -d ! $DMZ_IP_NET  --dport 1863 -j ACCEPT
```

ICQ

ICQ bruger i modsat messenger nogle flere porte, men vi har valgt kun at åbne op for port 5190, som er ICQ hovedport dette bevirker at der er nogen services i ICQ, som man ikke kan bruge, men hoved formålet, med programmeret er at man kan chatte. Denne kører også over TCP protokollen.

Vi sætter det op på følgende måde:

```
iptables -A FORWARD -p tcp -s $LAN_IP_NET -d ! $DMZ_IP_NET  --dport 5190 -j ACCEPT
```

IRC

Står for Internet Relay Chat og er ligeledes som MSN og ICQ et chat program, her kan man også dele filer med andre, men i stedet for MSN og ICQ, som begge mest bliver brugt til at chatte med en person, chatter man her i store rum, hvor der kan være mellem 3 og flere 1000 mennesker.
IRC kører som standard på 6667, men der mange som hoster flere servere og kun vil bruge en IP, så bliver 6668 og 6669 også brugt til dette.

Vi sætter det op på følgende måde:

```
iptables -A FORWARD -p tcp -s $LAN_IP_NET -d ! $DMZ_IP_NET  -m multiport --dport 6667,6668,6669 -j ACCEPT
```

Skype

Skype, er ligeledes et chat program, bare her skriver man ikke til hinanden her ringer man op og snakker så med hinanden, dette kan spare virksomhederne rigtigt mange penge. Det er gratis at ringe mellem hinanden og lyd kvaliteten, er i hvert fald når man "ringer" inden for landets grænser, bedre end normalt og da forbindelserne til udlandet er blevet noget bedre, så burde lydkvaliteten til udlandet også være ganske god.
Skype benytter port 33033, og bruger TCP trafik.

Vi sætter det op på følgende måde:

```
iptables -A FORWARD -p tcp -s $LAN_IP_NET -d ! $DMZ_IP_NET  --dport 33033 -j ACCEPT
```

ICMP

Vi har valgt at åbne op for alt icmp trafik, man kan dog godt bare åbne op for f.eks. ping, eller bare det modsatte pong (Ping er det som maskinen sender og et pong er det som der kommer retur), altså kan man lave sådan at man kan pinge ud på Internettet, men Internettet kan ikke pinge dig.

Vi sætter det op på følgende måde:

```
iptables -A FORWARD -p icmp -s $LAN_IP_NET -d ! $DMZ_IP_NET -j ACCEPT
```

De-Militær Zone

Rettighederne for den De-Militære Zone kan ses som tekst i bilag 1 og grafisk i bilag 2. Som man kan se på Bilag 2, er ikke muligt at skabe en forbindelse fra DMZ til det private net og det skyldes at DMZ har nogle åbne porte på Internettet der opfattes som en sikkerhedsbrist. Derimod kan det private net godt bruge de ydelser som er i DMZ'en.

Fra DMZ til Firewall'en er det kun muligt at skabe en SSH-forbindelse for ikke at skabe for meget usikkerhed på vores lille netværk. SSH'en skal bruges hvis Administratoren sidder ved en server i DMZ'en og skal have forbindelse til Firewall'en, for at lave eventuelle ændringer. Den anden vej er der selvfølgelig fuld adgang, altså fra Firewall'en til DMZ'en.

De ydelser som er på DMZ'en skal også være tilgængelige på Internettet, derfor er der åbent for web og FTP. Vi har så været så frie at sætte åbningstider på FTP-serveren for at dæmpe belastningen af netværket i arbejdstiden. Tiderne er som følger:

Åbningstider:	
Mandag – Fredag	16.00 – 08.00
Lørdag – Søndag	Åbent

Derudover skal DMZ'en have lov til følgende ydelser på Internettet: FTP, SSH, web, DNS, HTTPS, ICMP.

Opsætningen af rettighederne sker som vist herunder:

Først skal alt kommunikation til det private net spærres. Dog bliver der ikke helt spærret da vi kører med "stateless"[4] Firewall, det er stadig muligt at sende reply tilbage. Altså hvis en bruger fra det private net ønsker at komme på FTP'en i DMZ'en, så må DMZ'en gerne sende reply tilbage. $DMZ_IP_NET og $LAN_IP_NET kan ses på side 12.

```
iptables -A FORWARD -s $DMZ_IP_NET -d $LAN_IP_NET -j DROP
```

Kommunikationen fra det private net til DMZ'en er spærret som "standard", dvs. at det har vi spærret i starten af vores script[5]. Derefter åbner vi stille og roligt for de porte som vi skal bruge og udelukker derved alle andre porte.

Hvis administratoren sidder fysisk i DMZ'en skal han have mulighed for SSH til Firewall'en, derfor åbner vi for SSH fra DMZ:

```
iptables -A INPUT -p tcp --dport 22 -s $DMZ_IP_NET -d $DMZ_IP -j ACCEPT
```

Firewall'en har selvfølgelig fuld adgang den anden vej til DMZ'en og det er defineret i starten af scriptet[6].

[4] Se side 13
[5] Se bilag 3 side 30

DMZ → Internettet

```
iptables -A FORWARD -p tcp -s $DMZ_IP_NET -m multiport --dport 20,21 -j ACCEPT
iptables -A FORWARD -p tcp -s $DMZ_IP_NET --dport 22 -j ACCEPT
iptables -A FORWARD -p udp -s $DMZ_IP_NET --dport 53 -j ACCEPT
iptables -A FORWARD -p tcp -s $DMZ_IP_NET --dport 80 -j ACCEPT
iptables -A FORWARD -p tcp -s $DMZ_IP_NET --dport 443 -j ACCEPT
iptables -A FORWARD -p icmp -s $DMZ_IP_NET -j ACCEPT
```

Den første linie forwarder port 20 og 21 gennem Firewall'en via TCP protokollen og det er FTP'en der opererer gennem disse porte. –m multiport gør det muligt at sætte flere --dport's. Den næste er SSH på port 22. Herefter kommer DNS, WEB, HTTPS og til sidst ICMP.

Internettet → DMZ

Der skal være mulighed for at kontakte WEB-serveren fra Internettet og det gøres på følgende måde.

```
iptables -A FORWARD -p tcp -d 10.0.1.2 --dport 80 -j ACCEPT
iptables -t nat -I PREROUTING -p tcp -d $WAN_IP --dport 80 -j DNAT --to-destination -- 10.0.1.2:80
```

I den første linie accepterer vi trafik til 10.0.1.2 på port 80. Derefter forwarder vi trafikken fra WAN_IP (192.168.1.16) til webserveren 10.0.1.2 på port 80. Så nu skulle det være muligt at kontakte webserveren fra Internettet.

Så skal vi bare have sat rettigheder op for FTP-serveren og det gøres på næsten samme måde. Det er her vi har udvidet vores case lidt, ved at tilføje åbningstider. Først skal vi have accepteret trafik på FTP'ens 2 porte, 20 og 21. Det gøres ligesom beskrevet ved WEB ovenover, men for at åbne i et bestemt tidsrum er man nødt til at gøre brug af modulet **time** som vi har tilføjet til kernen med Patch-o-matic. Dernæst angiver man et starttidspunkt og et sluttidspunkt, samt hvilke dage det skal gælde.

```
iptables -A FORWARD -p tcp -d 10.0.1.2 -m multiport --dport 20,21 -m time --timestart 16:00 --timestop
→ 00:00 --days Mon,Tue,Wed,Thu,Fri -j ACCEPT

iptables -A FORWARD -p tcp -d 10.0.1.2 -m multiport --dport 20,21 -m time  --timestart 00:00 --timestop
→ 08:00 --days Mon,Tue,Wed,Thu,Fri -j ACCEPT

iptables -A FORWARD -p tcp -d 10.0.1.2 -m multiport --dport 20,21 -m time --timestart 00:00 --timestop
→23:59 --days Sat,Sun -j ACCEPT

iptables -A FORWARD -p tcp -d 10.0.1.2 -m multiport --dport 20,21 -m time --timestart 23:59 --timestop
→ 00:00 --days Sat,Sun -j ACCEPT
```

"→" betyder at linien fortsætter fra forrige linie og skal ikke medtages i den samlede kommando.

Herefter skal vi lige forwarde de to porte til FTP'en, ellers vil det jo ikke være muligt at kontakte FTP'en fra Internettet. Når en WAN_IP så forespørger på port 20 og 21 vil det blive forwardet til 10.0.1.2.

```
iptables -t nat -I PREROUTING -p tcp -d $WAN_IP --dport 20 -j DNAT --to-destination 10.0.1.2:20
iptables -t nat -I PREROUTING -p tcp -d $WAN_IP --dport 21 -j DNAT --to-destination 10.0.1.2:21
```

Trusler

Der vil altid være et par trusler i et computer system og hvis man skulle pege på nogen her, vil det være at man intern kunne bruge et hackerprogram. Ved at sætte den op til at bruge en af de porte som vi har åbnet op for, kan en såkaldt trojansk hest få forbindelse til Internettet fra vores net. Dette kan man dog godt opfange ved at kigge nærmere i pakkerne og grabbe noget af de informationer som står i den og så kunne acceptere dem på den måde.

Ligeledes kan vores DMZ zone SSH til vores Firewall, dette er lavet fordi, at når man sidder ved en server og har sat et nyt program op, så kan man lige åbne op for den port det nye program bruger. Dette kan man dog også laves på andre måder, f.eks. når man ændre noget så gør man det via fjernstyring fra det private netværk og således også fjernstyre Firewall'en fra det private netværk.

Derudover kan man hacke en af de programmer som kører på DMZ zonen og får installeret endnu en trojansk hest i en af de programmer, som kører på DMZ zonen. Når så en klient fra det private netværk anvender en applikation på DMZ, kan den trojanske hest flittigt anvendes til at komme videre på vores net.

Derfor er det en god ide at have en "log analyzer" som sender de vigtige informationer fra sine servere til sin e-mail. Og måske kunne det være praktisk at modtage en mail hver gang en bruger åbner en ny "shell" på sin Linux box.

Men truslen er ikke kun her den er også ved brugerne, som kan tage cd'er og disketter med hjemmefra med virus og trojanske heste, eller downloade ting fra Internettet med virus i. Ligeledes er opdateringer til folks styresystemer også en god ide. Derfor er det en god ide f.eks. at opdateringer bliver installeret automatisk og at virus scannerne selv henter opdateringer ned og installere dem. I hvert fald hvis man har mange maskiner til at stå i netværket.

Konklusion

Trods det at opsætningen af Ip-tables virker lidt overskueligt i starten, er det lykkedes at få stablet en velfungerende Firewall på benene. IP-tables har den enormt store fordel at den ikke er begrænset af en grafisk brugerflade med foruddefinerede konfigurationer. Det gør at man har fuldstændig frihed til at åbne og lukke for de porte, IP'er, MAC-adresser osv., man nu har lyst til.
Eftersom Linux jo er et open-source projekt, sidder der mange programmører og udvikler opdateringer, patches og diverse modifikationer. Det er endnu en stor fordel for os som brugere, eftersom vi bare kan downloade og bruge patchene efter behov. Vi downloadede patch-pakken Patch-o-matic og kompilerede den sammen med kernen, dog valgte vi kun at bruge én af de mange patches, nemlig time. Ellers er der mange andre udvidede muligheder for at konfigurere sin Firewall, bla. begrænsning af båndbredde osv. Vi har bare ikke haft tid til at teste alle mulighederne.
En lille ulempe ved opsætning af Firewall med IP-tables, er at det som sagt godt kan blive lidt overskuelig hvis man f.eks. har mange forskellige net at administrere, men denne lille detalje undertrykkes straks af den frie konfigurations-mulighed.

Vi har endnu engang måtte erkende at søgemaskinen Google er et unikt redskab når man arbejder med Linux på bruger-niveau. Vi er faktisk ikke stødt på de helt store revolutionerende fejl under opsætningen af Firewall'en, men har alligevel brugt Google til at søge svar på basale spørgsmål. F.eks. har vi læst en del tutorials omkring IP-tables igennem, for at få et bedre overblik over hvordan det egentlig hænger sammen.

Kildeliste

http://www.braindump.dk/dk/wiki/?wikipage=IpTables
http://www.netfilter.org/documentation/
http://iptables-tutorial.frozentux.net/iptables-tutorial.html
http://www.onlamp.com/linux/cmd/i/iptables.html
http://www.linuxguruz.com/iptables/howto/maniptables.html
http://www.gigascale.org/gsrc/faq/5.html
www.net-faq.dk

Bilag 1

- **Case udarbejdet i samarbejde med ITE 2.1.**

Case

Plot:

Denne case simulerer de forskellige rettigheder i en given virksomhed. Virksomheden skal indeholde et privat netværk med de ansattes PC'ere og derudover en DMZ med FTP og WEB-server. Brugerne bag Firewall'en skal have mulighed for at anvende diverse applikationer på Internettet. Med udgangspunkt i ovenstående er følgende rettigheder udarbejdet:

NB: Vi går ud fra at Administratoren har en PC på det private netværk.

Rettigheder:

Udefra til Firewall'en:

SSH fra administratorens hjemme IP.

Udefra og ind DMZ Netværket:

FTP'en & Web'en skal kunne ses udefra.

Private Netværket til ud:

Brugerne skal have tilladelse til følgende porte ud til Internettet.
MSNM (1863), Skype (33033), ICQ(TCP 5190), Web (80), DNS (53 udp), ICMP, FTP (20,21), HTTPS (443) , POP3 (110) , SMTP (25) , Samba (137,138,139), IRC (6667, 6668,6669), SSH (22)

Private Netværket til Firewall:

SSH (22) fra Administratorens IP.

Private til DMZ Netværket:

De skal have tilladelse til de services som der kører på serveren:
FTP (20,21)
Web (80)

DMZ til Private: Ingenting.

Udefra til Private: Ingenting.

Fra DMZ og ud:

ICMP, DNS, web, FTP, SSH.

Fra DMZ til Firewall:

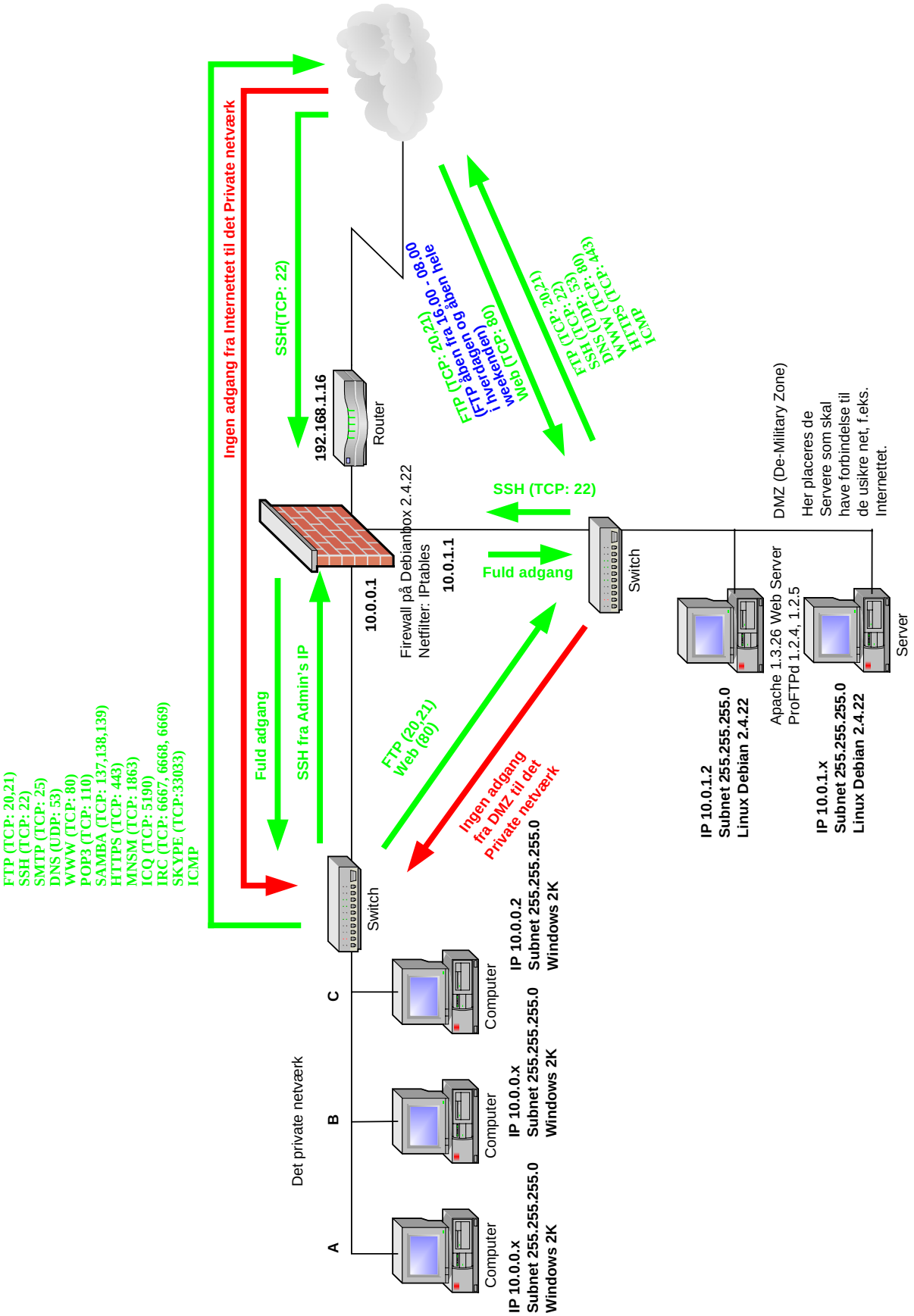
SSH.

Fra Firewall'en til det private netværk og DMZ:

Fuld adgang.

Bilag 2

- **Grafisk oversigt over vores Case**


Ingen adgang fra Internettet til det Private netværk
SSH(TCP: 22)
FTP (TCP: 20,21)
SSH (TCP: 22)
SMTP (TCP: 25)
DNS (UDP: 53)
WWW (TCP: 80)
POP3 (TCP: 110)
SAMBA (TCP: 137,138,139)
HTTPS (TCP: 443)
MNSM (TCP: 1863)
ICQ (TCP: 5190)
IRC (TCP: 6667, 6668, 6669)
SKYPE (TCP:33033)
ICMP
Fuld adgang
SSH fra Admin's IP
192.168.1.16
Router
FTP (TCP: 20,21)
FTP åben fra 16,00 - 08,00
(FTP åben o g åben hele
i hverdagen)
i weekenden)
Web (TCP: 80)
FTP (TCP: 20,21)
SSH (TCP: 22)
DNS (UDP: 53)
WWW (TCP: 80)
HTTPS (TCP: 443)
ICMP (21)
SSH (TCP: 22)
Fuld adgang
10.0.0.1
Firewall på Debianbox 2.4.22
Netfilter: IPtables
10.0.1.1
Switch
DMZ (De-Military Zone)
Her placeres de
Servere som skal
have forbindelse til
de usikre net, f.eks.
Internettet.
FTP (20,21)
Web (80)
Ingen adgang
fra DMZ til det
Private netværk
IP 10.0.1.2
Subnet 255.255.255.0
Linux Debian 2.4.22
Apache 1.3.26 Web Server
ProFTPd 1.2.4, 1.2.5
IP 10.0.1.x
Subnet 255.255.255.0
Linux Debian 2.4.22
Server
Switch
Det private netværk
C
B
A
Computer
IP 10.0.0.2
Subnet 255.255.255.0
Windows 2K
Computer
IP 10.0.0.x
Subnet 255.255.255.0
Windows 2K
Computer
IP 10.0.0.x
Subnet 255.255.255.0
Windows 2K

Bilag 3

- **Det samlede Firewall-script**

```
Diable forwarding
echo 0 > /proc/sys/net/ipv4/ip_forward

LAN_IP_NET='10.0.0.0/24'
DMZ_IP_NET='10.0.1.0/24'
WAN_IP='192.168.1.16'
DMZ_IP='10.0.1.1'

# Flush
iptables -t nat -F POSTROUTING
iptables -t nat -F PREROUTING
iptables -t nat -F OUTPUT
iptables -F

iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -P OUTPUT ACCEPT

# enable Masquerade and forwarding
iptables -t nat -A POSTROUTING -s $LAN_IP_NET -j MASQUERADE
iptables -t nat -A POSTROUTING -s $DMZ_IP_NET -j MASQUERADE
iptables -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT

# STATE RELATED for firewall
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

#Følgende må SSH firewallen, Ole, Kenneth & DMZ
iptables -A INPUT -p tcp --dport 22 -m mac --mac-source 00:0C:6E:21:D8:09 -j ACCEPT     #DALBJERG
iptables -A INPUT -p tcp --dport 22 -m mac --mac-source 00:00:E2:85:6C:E1 -j ACCEPT     #OLE
iptables -A INPUT -p tcp --dport 22 -s $DMZ_IP_NET -d $DMZ_IP -j ACCEPT

#Det lukket netværk & UD
iptables -A FORWARD -p tcp -s $LAN_IP_NET -m multiport --dport 20,21 -j ACCEPT         #FTP
iptables -A FORWARD -p tcp -s $LAN_IP_NET -d ! $DMZ_IP_NET --dport 22 -j ACCEPT        #SSH
iptables -A FORWARD -p tcp -s $LAN_IP_NET -d ! $DMZ_IP_NET  --dport 25 -j ACCEPT       #SMTP
iptables -A FORWARD -p udp -s $LAN_IP_NET -d ! $DMZ_IP_NET  --dport 53 -j ACCEPT       #DNS Opslag
iptables -A FORWARD -p tcp -s $LAN_IP_NET --dport 80 -j ACCEPT                         #Http
iptables -A FORWARD -p tcp -s $LAN_IP_NET -d ! $DMZ_IP_NET  --dport 110 -j ACCEPT      #POP3
iptables -A FORWARD -p tcp -s $LAN_IP_NET -d ! $DMZ_IP_NET  -m multiport --dport 137,138,139 -j ACCEPT  #SAMBA
iptables -A FORWARD -p tcp -s $LAN_IP_NET -d ! $DMZ_IP_NET  --dport 443 -j ACCEPT      #Https
iptables -A FORWARD -p tcp -s $LAN_IP_NET -d ! $DMZ_IP_NET  --dport 1863 -j ACCEPT     #MSN
iptables -A FORWARD -p tcp -s $LAN_IP_NET -d ! $DMZ_IP_NET  --dport 5190 -j ACCEPT     #ICQ
iptables -A FORWARD -p tcp -s $LAN_IP_NET -d ! $DMZ_IP_NET  -m multiport --dport 6667,6668,6669 -j ACCEPT      #IRC
iptables -A FORWARD -p tcp -s $LAN_IP_NET -d ! $DMZ_IP_NET  --dport 33033 -j ACCEPT    #SKYPE
iptables -A FORWARD -p icmp -s $LAN_IP_NET -d ! $DMZ_IP_NET -j ACCEPT                  #ICMP Pakker

#DMZ's netværk
iptables -A FORWARD -s $DMZ_IP_NET -d $LAN_IP_NET -j DROP                  #Dropper alle pakker fra DMZ til LAN
iptables -A FORWARD -p tcp -s $DMZ_IP_NET -m multiport --dport 20,21 -j ACCEPT         #FTP
iptables -A FORWARD -p tcp -s $DMZ_IP_NET --dport 22 -j ACCEPT             #ssh
iptables -A FORWARD -p udp -s $DMZ_IP_NET --dport 53 -j ACCEPT             #DNS
iptables -A FORWARD -p tcp -s $DMZ_IP_NET --dport 80 -j ACCEPT             #HTTP
iptables -A FORWARD -p tcp -s $DMZ_IP_NET --dport 443 -j ACCEPT            #Https
iptables -A FORWARD -p icmp -s $DMZ_IP_NET -j ACCEPT                       #ICMP Pakker
```

```
#Forward af porte fra vores WAN_IP til vores DMZ_IP_NET
#WEB TRAFIK (10.0.1.2), tilladt fra alle ipere.
	iptables -A FORWARD -p tcp -d 10.0.1.2 --dport 80 -j ACCEPT
	iptables -t nat -I PREROUTING -p tcp -d $WAN_IP --dport 80 -j DNAT --to-destination 10.0.1.2:80

#FTP Trafik
	iptables -A FORWARD -p tcp -d 10.0.1.2 -m multiport --dport 20,21 -m time --timestart 16:00 --timestop 00:00 --days Mon,Tue,Wed,Thu,Fri -j ACCEPT
	iptables -A FORWARD -p tcp -d 10.0.1.2 -m multiport --dport 20,21 -m time  --timestart 00:00 --timestop 08:00 --days Mon,Tue,Wed,Thu,Fri -j ACCEPT
	iptables -A FORWARD -p tcp -d 10.0.1.2 -m multiport --dport 20,21 -m time --timestart 00:00 --timestop 23:59 --days Sat,Sun -j ACCEPT
	iptables -A FORWARD -p tcp -d 10.0.1.2 -m multiport --dport 20,21 -m time --timestart 23:59 --timestop 00:00 --days Sat,Sun -j ACCEPT

	iptables -t nat -I PREROUTING -p tcp -d $WAN_IP --dport 20 -j DNAT --to-destination 10.0.1.2:20
	iptables -t nat -I PREROUTING -p tcp -d $WAN_IP --dport 21 -j DNAT --to-destination 10.0.1.2:21


# Enable forwarding
echo 1 > /proc/sys/net/ipv4/ip_forward
```