NTNU

Kunnskap for en bedre verden

Nettverkslaget/IP, og brannmurer

Bjørn Klefstad/Helge Hafting, Institutt for datateknologi og informatikk (IDI), NTNU
Lærestoffet er utviklet for faget <<INFT1007 Datakommunikasjon>>

Resymé: I denne leksjonen skal vi studere nærmere hvilke oppgaver som løses på Nettverkslaget og vi skal studere IP-protokollen i detalj. Vi skal se nærmere på både formatet og adressering.

Vi ser også på hva brannmurer er, og hvordan de fungerer.

Innhold

5 Nettverkslaget/IP, og brannmurer

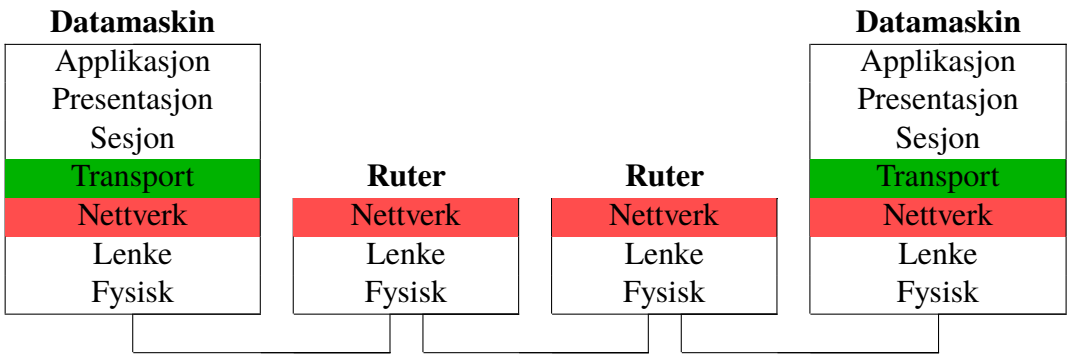
5.1 Nettverkslaget kap 5.1

Oppgaven til nettverkslaget er å sørge for at datapakker ender opp på den ønskede maskinen i internettverket. Som oftest må pakkene rutes (dirigeres) gjennom flere mellomliggende maskiner, rutere, før de kommer fram.

Et internettverk er et datanettverk som er sammensett av flere enkeltnettverk, altså et nettverk av nettverk. Noen ganger blir ordet internetting benyttet om å knytte sammen lokalnettverk, evt. av ulike typer (for eksempel Ethernet, punkt–til–punkt-forbindelser og FDDI) slik at de opererer som et stort nettverk.

Et viktig punkt på nettverkslaget er at det er definert en entydig adressestruktur. Dvs. at alle maskiner i internettverket har entydige adresser. Rutingen gjennom nettverket blir utført på grunnlag av analyse av nettverkslagsadresser og oppslag i rutetabeller. Vi kommer tilbake til rutere og ruting i leksjon 7.

Det vi til vanlig kaller Internett er det globale, åpne nettverket som knytter sammen alle lokale nettverk hos universitet, forskingsinstitutt, bedrifter, Internett-tilbydere osv. over hele verden. I dette nettverket blir det benyttet en familie av nettverkslags-, transportlags- og applikasjonslags-protokoller som man med et samlebegrep kaller TCP/IP. Nettverkslagsprotokollen i denne protokollfamilien heter IP, Internet Protocol.



Figur 5.1: Nettverkslaget i Internettet

Protokoller på nettverkslaget er ikke bare implementert i endeutstyret, som for transportlaget, men også i de mellomliggende maskinene (ruterne), se figur 5.1. Nettverkslaget overfører altså pakker fra maskin til maskin. Dette betyr at dersom vi legger inn funksjonalitet (for eksempel feilkontroll) på dette laget, vil pakkene bli sjekket (for feil) for hver

overføring mellom to maskiner. Eventuell retransmisjon vil også bli foretatt mellom to maskiner.

Dersom vi igjen sammenligner med transportlaget vil feilkontroll på nettverkslaget føre til langt større belastning på nettverket enn feilkontroll på transportlaget. På transportlaget sjekkes det ikke for feil før pakken er kommet frem til mottaker.

Det er derfor viktig at vi bruker protokoller på nettverkslaget som er så enkle som mulig for ikke å belaste nettverket unødvendig. Hvilke funksjonaliteter som det er behov for er selvfølgelig avhengig av egenskapene til selve nettverket. IP er et glimrende eksempel på en relativt enkel protokoll på nettverkslaget.

5.2 Nettbegreper kap 5.2

Tjenestenett Dagens nettverk tilbyr mange tjenester, ikke bare data. En kan f.eks. få telefoni og TV på det samme nettet også.

Logiske nett IP-adressene på nettverkslaget omtales som logiske adresser.

Fysiske nett MAC-adressene på lenkelaget omtales som fysiske adresser.

IP-nett Et IP-nett er et logisk nett på nettverkslaget. Innenfor et IP-nett kan maskiner kommunisere direkte med hverandre, eventuelt via en svitsj/hub. Maskiner i ulike IP-nett kan ikke kommunisere direkte med hverandre, men de ulike nettene kan kobles sammen ved hjelp av rutere. En ruter har alltid flere IP-adresser, en adresse for hvert IP-nett den er koblet til.

Lokalnett LAN Samlebegrep for kommunikasjonsutstyret i en virksomhet. Kan inneholde opptil flere IP-nett, implementert med vanlige teknologier som kabling og trådløse soner.

Virtuelle nett Brukes om to ulike teknologier:

- VLAN Virtuelle lokalnett. Man har flere ulike IP-nett, som bruker samme infrastruktur. (Kabler og svitsjer.) Dette sparer utstyr når man trenger flere IP-nett på ett sted. De ulike IP-nettene er adskilt på vanlig måte. De kan bare kommunisere med hverandre via ruter, selv om de bruker de samme fysiske forbindelsene.
- VPN Virtuelt privat nett. En teknikk for å koble sammen bedriftsinterne IP-nett som er geografisk spredt. Forbindelser mellom ulike lokasjoner går via Internettet, men krypteres slik at utenforstående ikke kan lytte. Slike forbindelser skal være like sikre som interne forbindelser i bedriften.

Hjemmenett	Typisk et privat IP-nett, hvor en eller fler maskiner deler på én IP-addresse. Dette sparer IP-adresser, som det dessverre er få av.
Bedriftsnett	Består av ett eller flere IP-nett, ofte har hver maskin egen IP-adresse.
Internett	Et stort globalt nettverk, som består av hundretusener av IP-nett.
Intranett	Ikke egentlig et nett, men en tjeneste. Brukes typisk om web-tjenester som bare er tilgjengelige internt i en bedrift.
Subnett	Gammel betegnelse på et IP-nett som er skilt ut fra et større IP-nett. I dag benyttes klasseløs ruting, så alle IP-nett kan sees på som «subnett» som er skilt ut fra helheten.

5.3 IP-adresser

5.3.1 IP-adressering kap 5.2 og 5.3

To maskiner som skal kommunisere, er koblet sammen via ett eller flere nettverk (typisk et Ethernet lokalnettverk). Alle noder har IP-adresser som er 32 biter lange, og som er hierarkisk oppbygd. Den hierarkiske oppbygningen gjør det mulig å adresse hele adresserom ved hjelp av én IP-adresse. Mer om dette i leksjon 7, som handler om rutere og ruting.

En IP-adresse er satt sammen av to deler:

- Adressen til nettverket som maskinen står i (nettadresse)
- Maskinen sin lokale adresse i det aktuelle nettverket (nodeadresse)

Hvor stor del av IP-adressen som blir benyttet til nettadresse og hvor stor del som blir benyttet til nodeadresse kan variere fra nettverk til nettverk. For å holde styr på nettopp denne inndelingen skiller vi mellom to ulike adresseringssystemer:

- Classless Internet Domain Routing
- Klassebasert adressering

Merk at vi her snakker om to atskilte systemer som ikke må blandes sammen.

CIDR — «Classless Internet Domain Routing»

For dette adresseringssystemet er det nettmasken som styrer delingen mellom nettadresse og nodeadresse. Ved hjelp av nettmasken kan vi plassere dette skillet der vi måtte ønske. Vi klarer dermed å tilpasse størrelsen på reservert adresserom sånn omtrent til størrelsen på det aktuelle nettverket.

Informasjon om hvor skillet mellom nettadresse og nodeadresse går må finnes i alle maskinene i nettverket. Når en maskin skal settes opp med sin IP-adresse må også nettmasken spesifiseres.

Vi har innenfor CIDR muligheten til å sette skillet mellom nettadresse og nodeadresse tilsvarende inndelingen i klassebasert adressering:

- Tilsvarende klasse A nett kan vi bruke nettmasken 255.0.0.0, eller /8
- Tilsvarende klasse B nett kan vi bruke nettmasken 255.255.0.0, eller /16
- Tilsvarende klasse C nett kan vi bruke nettmasken 255.255.255.0, eller /24

Jeg minner igjen om at vi ikke må blande sammen disse to ulike adresseringssystemene. Det er to atskilte systemer der vi med CIDR kan findele det aktuelle adresserommet i motsetning til klassebasert adressering der vi bare kan oppnå en grovinndeling. Pr i dag er det CIDR som er i bruk.

To måter å notere nettmasker

Nettmasker kan skrives som fire tall med punktum i mellom, på samme måte som ip-adresser. F.eks 255.255.255.0 Nettmasken er et 32-bits binært tall, som vi her har delt opp i 4 byte som er skrevet med vanlige tall. Dette er den gamle måten å notere nettmasker. Den binære nettmasken består imidlertid alltid av en serie med enere, fulgt av en serie med nuller. Vi har derfor fått en mer moderne måte å notere nettmasker, som bare forteller hvor mange ener-bits det er i nettmaska. Så i stedet for 255.255.255.0, kan vi skrive /24 som er kortere. Tilsvarende er 255.255.255.128 det samme som /25, og 255.254.0.0 er det samme som /15. Sammenhengen ser vi, om vi regner om til binære tall. 255.254.0.0 blir 11111111.11111110.00000000.00000000, og her ser vi at det er 15 enere.

Noen nettmasker

Bits	gml	Bits	gml	Bits	gml
/8	255.0.0.0	/16	255.255.0.0	/24	255.255.255.0
/9	255.128.0.0	/17	255.255.128.0	/25	255.255.255.128
/10	255.192.0.0	/18	255.255.192.0	/26	255.255.255.192
/11	255.224.0.0	/19	255.255.224.0	/27	255.255.255.224
/12	255.240.0.0	/20	255.255.240.0	/28	255.255.255.240
/13	255.248.0.0	/21	255.255.248.0	/29	255.255.255.248
/14	255.252.0.0	/22	255.255.252.0	/30	255.255.255.252
/15	255.254.0.0	/23	255.255.254.0		

Se kapittel 5.5 på side 10 for omregning mellom vanlige tall og binære tall.

Oppdeling av nettverk

Dersom vi tar utgangpunkt i adresserommet 158.38.31.0/24, skal vi se nærmere på hvordan disse adressene kan deles opp i flere atskilte nettverk, ved å endre på nettmasken. En ulempe ved slik oppdeling er at hvert nettverk mister to adresser. Den høyeste er reservert for broadcast, og den laveste for nettadressen. Disse adressene kan ikke brukes av nodene. I tillegg går det gjerne med en adresse til ruteren også. Jo mer vi deler opp, jo flere adresser mister vi på denne måten. I utgangspunktet har vi ett nettverk:

Nettadresser	Noder	Brukbare
158.38.31.0/24	256	254

Dersom vi ønsker å dele det opprinnelige adresserommet i to nettverk, kan vi øke nettmasken med 1. Vi får da følgende situasjon:

Nettadresser	Noder	Brukbare
158.38.31.0/25	128	126
158.38.31.128/25	128	126
Totalt	256	252

Dersom vi ønsker å dele det opprinnelige adresserommet i fire nettverk, kan vi øke nettmasken med 2. Vi får da følgende situasjon:

Nettadresser	Noder	Brukbare
158.38.31.0/26	64	62
158.38.31.64/26	64	62
158.38.31.128/26	64	62
158.38.31.192/26	64	62
Totalt	256	248

Dersom vi ønsker å dele det opprinnelige adresserommet i åtte nettverk, kan vi øke nettmasken med 3. Vi får da følgende situasjon:

Nettadresser	Noder	Brukbare
158.38.31.0/27	32	30
158.38.31.32/27	32	30
158.38.31.64/27	32	30
158.38.31.96/27	32	30
158.38.31.128/27	32	30
158.38.31.160/27	32	30
158.38.31.192/27	32	30
158.38.31.224/27	32	30
Totalt	256	240

Slik kan vi fortsette oppdelingen så lenge vi kan øke nettmasken og vil etter hvert få mindre og mindre nettverk. Det er også mulig å dele opp i adresserom av ulik størrelse, men det velger vi å ikke komme nærmere inn på her.

Klassebasert adressering

Ved denne formen for adressering er det hvilken klasse adressen tilhører, som styrer inndelingen i nettadresse og nodeadresse. De ulike klassene finner du beskrevet i boken. Siden vi kun skiller mellom 5 ulike klasser, gir dette en relativt grov inndeling av alle tilgjengelige adresser. Dette medfører at små nettverk vil låse mange ubrukte IP-adresser.

5.3.2 IP-adresser til spesiell bruk kap 5.5

I dette kurset holder det at vi har oversikt over at det finnes en del reserverte IP-adresser og hva som er hensikten med disse.

Private IP-adresser

Dette er en serie reserverte IP-adresser som alle kan bruke fritt så lenge de aktuelle maskinene ikke kobles direkte til Internettet. Hovedpoenget med å benytte nettopp denne serien er å ha muligheten til å kunne koble nettet til Internett via en NAT-ruter uten å måtte omkonfigurere nettet. Merk at de private adressene ikke er rutbare i det globale nettet.

De private adressene er seriene 10.0.0.0/8, 172.16.0.0/12 og 192.168.0.0/16.

Multicasting

Vi har også en serie reserverte adresser som kan brukes til å sende fra en maskin til en gruppe av andre maskiner. Dette forutsetter at ruterne støtter slik funksjonalitet. Dette er praktisk for kringkasting av informasjon, hver pakke sendes bare ut én gang i stedet for at den må sendes separat til alle mottakere.

Nettverks- og kringkastingadresser

Bare 1'ere i nodeadressedelen brukes til å kringkaste en pakke til hele nettverket. Det vil si til og med nærmeste ruter. Bare 0'ere i nodeadressedelen brukes til å adressere et bestemt nettverk. Dermed har alle nettverk to adresser som ikke kan brukes som node-adresser.

Link-lokale adresser

En serie adresser satt av for bruk i små isolerte IP-nett. Disse adressene gjør det enklere for ukyndige å sette opp nettverk. Hver enhet (PCer, nettskrivere, og lignende) finner seg IP-adresser selv. Dette skjer ved at enheten velger en tilfeldig adresse fra serien, og sjekker (ved hjelp av ARP) at den ikke er opptatt. I så fall prøver maskinen igjen med en annen link-lokal adresse.

De link-lokale IP-adressene er serien 169.254.1.0–169.254.254.255 Disse adressene rutes ikke, så de kan bare brukes til direkte kommunikasjon mellom enheter i samme nettverk.

Loopback-adresser

Her har vi den velkjente 127.0.0.1. Hele serien 127.0.0.0/8 fungerer på samme måte. Loopback-adresser rutes ikke på nettet. De brukes når programmer som kjører på samme maskin, snakker med hverandre. F.eks. hvis man kjører en webserver på sin egen maskin, da finner man sidene på `http://127.0.0.1/`.

5.4 Internettprotokollen kap 5.6

Vi gjentar egenskapene til IP her da disse er meget sentrale:

IP er en upålitelig protokoll som opererer forbindelsesløst. Det er altså ingen garanti for at IP-pakkene kommer frem, og at de kommer frem feilfrie. IP-pakkene blir overførte uavhengig av hverandre, og da er det heller ingen garanti for at den kommer frem i samme rekkefølgen som den ble sendt. For eksempel kan ulike pakker (som er uavhengige av hverandre) bli rutet langsmed to alternative veier gjennom nettverket.

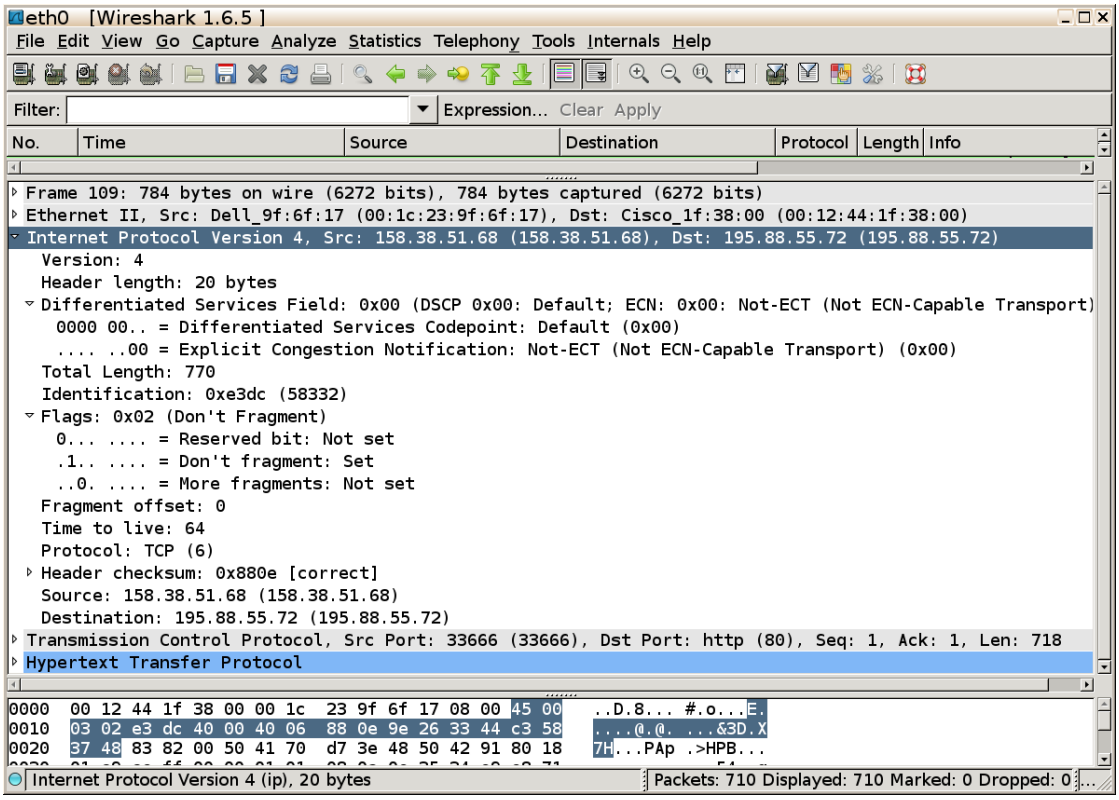
Den voldsomme veksten i bruken av IP-protokollen de siste årene, henger sammen med at den kan overføre data fra hvilken som helst protokoll på laget over og at IP-pakker kan overføres på nær sagt alle ulike typer nettverk.

I dette kapitlet er det IPv4 vi tar for oss.

5.4.1 IP-formatet

Formatet og de ulike feltene som vi finner i en IP-header er beskrevet i læreboken. Her kan det være greit å merke seg at størrelsen på IP-header kan variere. Størrelsen på en IP-pakke tilpasses som regel lenkelagsrammene, som typisk er 1500 byte.

Figur 5.2 på neste side viser konkrete verdier til de ulike feltene i en IP-pakke:



Figur 5.2: Header-felt for en IP-pakke

Versjonsnummer	4
Headerlengde	20 byte
Type of service	Ikke i bruk da alle biter er null
Totallengde	770 byte – betyr at nyttelasten er 750 byte da header er 20 byte
Identifisering	0xe3dc
Flagg	010 –betyr at pakken ikke er fragmentert
Fragmentering	0, ikke i bruk, da flagget for fragmentering ikke er satt
Time to live	64
Protokoll	TCP – transportprotokollen som skal benyttes
Header sjekksum	0x880e – er sjekket og funnet i orden
Avsenders IP-adresse	158.38.51.68
Mottakers IP-adresse	195.88.55.72

Tilleggsinformasjon	Ikke i bruk da header er 20 byte
TCP	Fra port 33666 til port 80 (normalt http)
HTTP	Ikke vist her, men wireshark kan vise detaljer om http-protokollen også.

Gjør en pakkefangst med Wireshark selv og analyser innholdet slik at du kjenner igjen de ulike feltene i headeren.

5.4.2 Fragmentering kap 5.7

Fragmentering utføre underveis i nettverket, ettersom pakken forflytter seg mellom de ulike maskinene. Så fort lenkelagsrammen er for liten må pakken fragmenteres. Pakker kan i blant fragmenteres flere ganger, hvis fragmentene kommer til et nettverk med enda mindre maksstørrelse.

Defragmentering blir gjort hos mottaker for å spare ruterne for denne belastningen. Dessuten er det dumt å sette sammen fragmentene for tidlig, det kan jo hende det kommer enda en link med begrenset maksstørrelse.

5.5 Omregning mellom 10-tallssystemet og binære tall

I tilknytning til IP-adresser er det greit å kunne regne mellom 10-tallssystemet og binære tall.

Eksempel 11010100

Ta utgangspunkt i det binære tallet 11010100. For å regne om til 10-tallssystemet må vi multiplisere alle siffer i det binære tallet med en eller annen potens av 2. Dersom vi starter med sifferet lengst til høyre skal det multipliseres med 2^0. Det andre sifferet skal multipliseres med 2^1, det tredje med 2^2 osv. Deretter summeres de ulike svarene slik:

$$1 \cdot 2^7 + 1 \cdot 2^6 + 0 \cdot 2^5 + 1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 =$$

$$1 \cdot 128 + 1 \cdot 64 + 0 \cdot 32 + 1 \cdot 16 + 0 \cdot 8 + 1 \cdot 4 + 0 \cdot 2 + 0 \cdot 1 =$$

$$128 + 64 + 0 + 16 + 0 + 4 + 0 + 0 = 212$$

Så kan vi prøve å regne den andre veien. Vi tar utgangspunkt i tallet 212 og skal regne om til et binært tall. Vi halverer tallet (deler med 2) til det ikke er noe igjen. Ved hver halvering noterer vi et siffer: 0 hvis divisjonen gikk opp, 1 hvis den ikke gikk opp. Slik får vi de binære sifrene:

$\frac{212}{2} = 106$ 0 i rest, siste siffer i det binære tallet blir altså 0

$\frac{106}{2} = 53$ 0 i rest, nest siste siffer blir også 0

$\frac{53}{2} = 26$ 1 i rest, nest nest siste siffer blir altså 1

$\frac{26}{2} = 13$ 0 i rest, …

$\frac{13}{2} = 6$ 1 i rest, …

$\frac{6}{2} = 3$ 0 i rest, …

$\frac{3}{2} = 1$ 1 i rest, nest første siffer blir altså 1

$\frac{1}{2} = 0$ 1 i rest, første siffer blir 1

Når har vi alle sifrene, og kan skrive opp det binære tallet 11010100.

Eksempel 11000100111

Her har vi 11 binære sifre, så i beregningen bruker vi $2^{10}, 2^9, \ldots, 2^0$:

$1 \cdot 2^{10} + 1 \cdot 2^9 + 0 \cdot 2^8 + 0 \cdot 2^7 + 0 \cdot 2^6 + 1 \cdot 2^5 + 0 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 =$

$1 \cdot 1024 + 1 \cdot 512 + 0 \cdot 256 + 0 \cdot 128 + 0 \cdot 64 + 1 \cdot 32 + 0 \cdot 16 + 0 \cdot 8 + 1 \cdot 4 + 1 \cdot 2 + 1 \cdot 1 =$

$1024 + 512 + 0 + 0 + 0 + 32 + 0 + 0 + 0 + 4 + 2 + 1 = 1575$

For å gå andre vegen og regne 1575 om til et binært tall, deler vi med to til det ikke er noe igjen:

$\frac{1575}{2} = 787$ 1 i rest, så siste siffer i det binære tallet blir 1

$\frac{787}{2} = 393$ 1 i rest, så nest siste siffer blir også 1

$\frac{393}{2} = 196$ 1 i rest, …

$\frac{196}{2} = 98$ 0 i rest, …

$\frac{98}{2} = 49$ 0 i rest, …

$\frac{49}{2} = 24$ 1 i rest, …

$\frac{24}{2} = 12$ 0 i rest, …

$\frac{12}{2} = 6$ 0 i rest, …

$\frac{6}{2} = 3$ 0 i rest, …

$\frac{3}{2} = 1$ 1 i rest, nest første siffer blir altså 1

$\frac{1}{2} = 0$ 1 i rest, første siffer blir 1

Skriver vi opp alle sifrene, får vi det binære tallet 1100010011.

En annen måte, uten divisjon

Tallene i IP-adresser ligger alltid mellom 0 og 255. De er altså på 8 bit. Bitposisjonene i et 8-bits binærtall har verdiene 128, 64, 32, 16, 8, 4, 2 og 1. Legg merke til at hvert tall er dobbelt så stort som det neste i rekka. Dette kan vi bruke for å regne om til binært. Eksempel med tallet 73:

73<128	Første bit blir 0, fordi vi *ikke* kan trekke fra 128
73>64	Denne biten blir 1, fordi vi kan trekke fra 64. Vi trekker 64 fra 73, og står igjen med 9.
9<32	Denne biten blir 0, vi kan ikke trekke fra 32
9<16	Denne biten blir også 0
9>8	Denne biten blir 1. Vi trekker fra 8, og står igjen med 1
1<4	Denne biten blir 0, kan ikke trekke fra 4
1<2	Denne biten blir 0
1=1	Denne biten blir 1

Kort sagt, vi prøver å trekke fra hver av bit-verdiene fra 128 og nedover. Når det går an, skriver vi 1. Når det ikke går, skriver vi 0. 73 ble altså 01001001. Tilsvarende eksempel med tallet 176:

176>128	1 (og 176-128=48)
48<64	0
48>32	1
16=16	1
0<8	0
0<4	0
0<2	0
0<1	0

Så 176 blir altså 10110000 binært. Noen synes denne metoden er enklere, man slipper å dele. I stedet må man huske doblingsrekka.

5.5.1 Heksadesimale tall

I wireshark-eksempelet så vi noen heksadesimale tall, som f.eks. 0xe3dc. Merk at «0x» ikke er en del av tallet, men prefikset «0x» brukes for å angi at det som kommer videre, er et heksadesimalt tall. Det heksadesimale tallet her, er altså «e3dc».

Noen av sifrene i «e3dc» er bokstaver. Det er fordi det heksadesimale tallsystemet har 16 som grunntall, ikke 10. Dermed trengs 16 ulike sifre for å bygge opp tall. Man har valgt å bruke de vanlige sifrene 0–9 som de 10 første, deretter bokstavene a–f som de 6 siste.

Heksadesimalt siffer	0	1	2	3	4	5	6	7
Verdi i titallsystemet	0	1	2	3	4	5	6	7
Verdi i totallsystemet	0000	0001	0010	0011	0100	00101	0110	0111

Heksadesimalt siffer	8	9	a	b	c	d	e	f
Verdi i titallsystemet	8	9	10	11	12	13	14	15
Verdi i totallsystemet	1000	1001	1010	1011	1100	1101	1110	1111

Omregning til titallsystem: $e3dc_{16} = 14 \cdot 16^3 + 3 \cdot 16^2 + 13 \cdot 16 + 12 = 58332$

Grunnen til at heksadesimale tall brukes, er at de er kortere og greiere å håndtere enn tall i totallsystemet. Samtidig er det veldig lett å konvertere mellom heksadesimalt og totallsystem.

Å oversette mellom titallsystem og totallsystem involverer en del regning. Det slipper vi, når vi konverterer mellom heksadesimalt og totallsystem. Dette fordi vi kan oversette ett og ett siffer om gangen. Eksempel:

Tar vi «e3dc» og oversetter siffer for siffer, får vi e=1110, 3=0011, d=1101 og c=1100. Så er det bare å trekke sammen, og vi får e3dc=1110 0011 1101 1100. Å oversette tilbake er like enkelt – det er bare å dele binære tall opp i grupper på fire binære sifre, for deretter å slå dem opp. F.eks:

10011101 = 1001 1101 = 9d.

Lange binære tall er mye å skrive opp, og vanskelige å huske. «e3dc2 er f.eks. greiere å håndtere enn «1110001111011100». Men noen ganger trenger vi den binære formen, f.eks. når vi ser på protokolldetaljer. TCP-flagg er f.eks. definert slik at bestemte bitposisjoner tilsvarer bestemte flagg som SYN, ACK, FIN. Og hvis man jobber med elektronikk, kan man oppleve at ulike bits overføres parallelt på ulike ledninger.

5.6 IPv6 kap 10.3.2

Se kapittel 10.3.2 i læreboken. Det disponible adresserommet med 32 bits adresser (IPv4) har etter hvert blitt for lite. Dette løses av IPv6, som har 128 bits adresser. Samtidig ble også en rekke andre svakheter med IPv4 endret. Se i læreboken for en oversikt over de viktigste endringene.

Til tross for at IPv6 er ferdig utviklet som standard for mange år siden, har implementeringen gått relativt sakte. Dette henger sammen med at det må investeres i nytt utstyr

i kjernenettet før IPv6 kan tas i bruk. Overgangen skjer derfor gradvis, etter hvert som nytt utstyr blir kjøpt inn. IPv6 er utviklet slik at den fungerer sammen med IPv4.

Arbeidet med å utvikle IPv6 startet i 1990 og hadde som målsetting:

- Et adresserom som aldri blir for lite
- Små rutetabeller
- Enklere format og prosedyre, slik at ruting kan gjøres raskere
- Bedre trygghet enn IPv4
- Bedre støtte for Type of Service, spesielt for sanntidsdata
- Bedre mobilitet (roaming uten å skifte adresse)
- Tillate utvikling av protokollen
- Tillate sameksistens med IPv4

Her er det viktig å merke seg forskjellen i oppbygningen av en IP-header for de to ulike versjonene. For IPv6 har headeren nå blitt atskilling enklere, samt at den har fått en fast størrelse. Fast størrelse på header gjør ruterens jobb enklere, det blir lettere å implementere ruting i hardware.

Vi kan også trekke frem at fragmenteringen nå er løst forskjellig fra IPv4. IPv4-pakker fragmenteres av rutere, som dermed må bruke tid og regnekraft på dette. Etterpå må mottaker sette fragmentene sammen til pakker igjen. Når ruteren får en IPv6-pakke som er for stor til å sendes videre, kaster den pakken og sender en ICMPv6-beskjed om hva den maksimale størrelsen er. Endenodene får dermed vite hva maksimal pakkestørrelse er, og sørger selv for å sende tilstrekkelig små pakker. Dermed trenger ikke ruterne dele opp pakker, de bare sender dem videre.

Med den nye standarden har vi også fått bedre støtte for ulike typer datatrafikk. Mer om dette i læreboken.

5.7 Om brannmurer

En brannmur er et filter mellom to eller flere nettverk, som begrenser trafikken av sikkerhetshensyn.

Ofte er det ene nettet resten av Internettet, mens de andre nettene er våre interne nettverk. Det hender imidlertid at man har brannmur mellom interne nettverk også, f.eks. når en del av virksomheten har spesielle behov.

Brannmurens jobb er å stanse nettbaserte angrep og spionasje. Den oppnår dette ved å vurdere all trafikk som er på vei igjennom, og stopper alt som ikke passer med reglene vi har satt.

Noen brannmurer er bokser som er laget for å bare være brannmur, og gjerne også ruter. Andre brannmurer er mer som en vanlig PC, med to eller flere nettkort og passende programvare. Linux med iptables er en grei løsning. Dette er fleksibelt, og man kan evt. kjøre andre beslektede tjenester som epostfilter og webproxy på den samme maskinen.

I tillegg til brannmurer som beskytter nettverk, kan enhver datamaskin ha programvare som beskytter den mot nettbaserte angrep. Dette er som regel en god idé. Man kan ha maskiner med ulike sikkerhetsbehov i samme nett. Og hvis noen først har fått tilgang innenfor brannmuren, er det bra at ikke alt ligger åpent.

5.8 Vanlige topologier

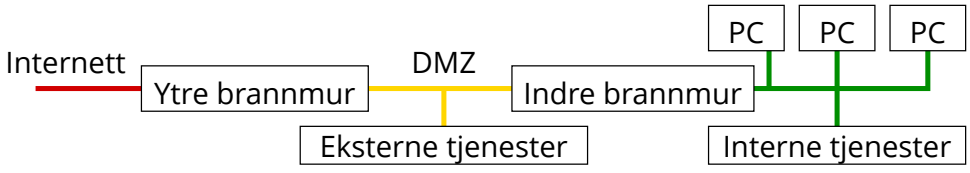
Figurene 5.3 og 5.4 på neste side viser to mye brukte topologier. Begge gir oss to soner:

sikker Den sikre sonen er stedet hvor du har filtjener og vanlige PCer. Disse kan ha tilgang på internett, men det er vanligvis ikke mulig for utenforstående å kontakte maskiner i sikker sone.

DMZ (De-Militarized Zone) Dette er sonen for tjenermaskiner som må kunne kontaktes utenfra. For eksempel webserver og epost.

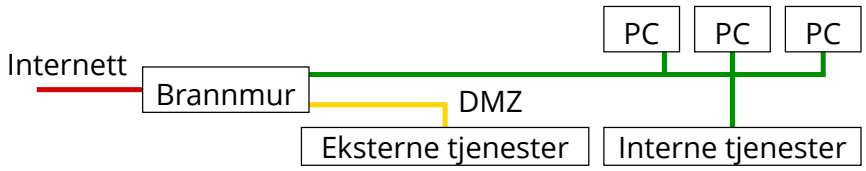
Topologiene kan utvides; bedrifter med flere avdelinger har ofte én sikker sone for hver avdeling.

Opplegget med to murer var mer populært tidligere. Begrenset regnekraft og stor båndbredde gjorde at den ytre muren måtte være enkel. Hvis ikke, ville den forsinke de eksterne tjenestene, og ingen vil ha et tregt nettsted. Den indre muren har mindre trafikk, og kan dermed ha et mer detaljert regelsett. Den er gjerne også strengere, siden omverdenen typisk ikke trenger å kunne kontakte tjenester på det interne nettet. En annen fordel med to murer, er at en inntrenger som utnytter en feil eller svakhet for å komme gjennom den første muren, ikke nødvendigvis kommer gjennom den andre også. Dette forutsetter at de to murene er av ulik type/fabrikat.



Figur 5.3: Ytre og indre brannmur

Etterhvert har vi fått kraftigere prosessorer og bedre implementasjoner. Brannmurer med flere ben har dermed blitt mer populære. Det blir en boks mindre å administrere, og



Figur 5.4: Brannmur med flere ben

mindre forsinkelser fordi pakker fra det interne nettet har ett ledd mindre å gå gjennom. Hvis det ikke er feil i selve brannmuren, er opplegget med én mur minst like sikkert som med to murer. Vi har mulighet for å gjøre alle de samme blokkeringene. Opplegget med to murer har en svakhet; hvis inntrengere tar kontroll over en maskin i DMZ, kan de lytte på trafikk mellom det interne nettet og Internettet. Den muligheten fins ikke med en mur med flere ben.

Brannmuren beskytter selvsagt mot trusler fra Internettet, men den beskytter gjerne nettene mot hverandre også. Hvis hackere tar over en webtjener i DMZ, er det jo fint om de ikke kan komme inn på det interne nettet derfra. Tilsvarende er det fint om et eventuelt virus i det interne nettet ikke kan spre seg til DMZ eller til andre avdelinger som har egne nett.

Normalt kan det ikke være helt stengt mellom DMZ og det interne nettet. Brukerne har jo bruk for tjenestene i DMZ, og noen av dem har sannsynligvis bruk for å administrere tjenermaskinene også.

5.9 Internett-tilgang

Sikkerhetsbehov kan være veldig forskjellige når en kobler et nettverk opp mot internettet. I et åpent miljø klarer en seg med få restriksjoner, mens andre kan ha behov for mer striks kontroll med både inn- og utgående trafikk.

Universiteter og høgskoler er et godt eksempel på det første tilfellet, det er få «hemmeligheter» og stort behov for problemfri tilgang til nettet. På slike steder forskes og eksperimenteres det med nye protokoller og tjenester. Dermed er det ikke aktuelt å sperre alt som en ikke eksplisitt har bruk for, slik det ellers ofte anbefales når en setter opp brannmur. En professor som trenger å bruke en eksperimentell tjeneste på et annet universitet, har ikke tid til å vente på at driftspersonell skal komme på jobb og åpne den aktuelle porten. Spesielt ikke hvis den må åpnes på det andre universitetet også…

Før sammenslåing med NTNU, valgte vi å sperre minst mulig. Vi lot heller tjenermaskinene beskytte seg selv ved å ikke kjøre unødige nettbaserte tjenester, og selv nekte forbindelser utenfra for interne tjenester. Dermed kunne eksperimentelle tjenester være

fritt tilgjengelige uten å hindres av overivrige brannmurer. Filtjenerne kunne f.eks. ikke kontaktes utenfra, men det var ikke brannmurens fortjeneste. Tjenerne var satt opp til å ikke svare på henvendelser som ikke har intern ip-adresse.

Organisasjoner som forsvaret og helsevesenet har derimot mye større behov for konfidensialitet, og velger derfor strengere brannmurer. De kan til og med gå inn for å ha flere nett, hvor noen ikke har forbindelse med internettet i det hele tatt. Bedrifter havner gjerne et sted mellom disse ytterpunktene, med behov for kommunikasjon (web, e-post, filoverføring, hjemmekontorer) samtidig som de har interne filtjenester med behov for konfidensialitet.

5.10 Typer brannmurer

5.10.1 Enkelt pakkefilter

Et pakkefilter ser på pakkene som rutes gjennom brannmuren. For hver pakke som kommer slår brannmuren opp i sine regler, og avgjør om pakken skal få komme igjennom eller ikke. Hvorvidt pakken kommer gjennom avhenger bare av dens innhold, og reglene. Dette gjelder alle pakker; ikke bare pakker utenfra. Pakkefilter er den enkleste formen for brannmur.

De vanligste reglene går på å undersøke portnumre og ip-adresser. Portnumrene forteller oss hvilken tjeneste det dreier seg om, til- og fra-adressene forteller hvor pakken er på vei.

Protokoller

De vanligste ip-baserte protokollene er TCP, UDP og ICMP. En del velger å filtrere ut UDP, med unntak av DNS som trenger UDP port 53. En del video- og telefonitjenester bruker UDP, så pass på om du har bruk for slikt.

Mange legger også begrensninger på ICMP. ICMP brukes av «ping», som kan misbrukes til å kartlegge andres nettverk. (Send ping til alle adresser i nettet, og se på hvilke som svarer.) Det er fornuftig å blokkere innkommende «ping», eller «ICMP echo request». En bør imidlertid ikke blokkere all ICMP. ICMP brukes også til å gi beskjeder som «no route to host» og lignende. Hvis du prøver å åpne `http://158.38.56.1/` får du normalt beskjed om at det ikke går. (Fordi det ikke er noen webserver på den adressen.) Hvis du har blokkert all ICMP, får ikke nettleseren din beskjeden. Dermed blir den stående og venter på et svar som aldri kommer. Etter to minutter får du en TCP-timeout, men det hadde jo vært mye greiere å få beskjeden på sekundet!

Det fins andre ip-baserte protokoller enn TCP, UDP og ICMP. De fleste bruker dem ikke, og kan blokkere alt annet i brannmuren.

Portnumre

Et eksempel kan være å bare tillate port 25 (smtp) hvis til- eller fra-adressen tilhører eposttjeneren. På det viset må epost gå via tjeneren. Dette tiltaket luker ut epost-baserte virus som ofte forsøker å spre seg ved å sende post direkte fra infiserte maskiner til tilfeldige posttjenere ute i verden.

Mange sperrer for tjenester de ikke har bruk for. Det har vist seg vanskelig å sperre torrents basert på portnumre, fordi torrents kan bruke nesten hvilke som helst porter.

Noen tjenester har dårlig sikkerhet. Hvis de brukes internt, bør de absolutt sperres mot Internettet. Windows fildeling er et slikt eksempel; passordene på denne tjenesten er enkle å knekke for alle som kan kontakte tjeneren.

Tvilsomme ip-adresser

I tillegg til portnumre forsøker noen å blokkere ip-adresser med «tvilsomt innhold». Dette er vanskelig i praksis, da det etterhvert er mange «tvilsomme» nettsteder, og det dukker stadig opp nye. Noen blokkerer facebook o.l. for at ansatte ikke skal kaste bort arbeidstid. Men det fins veldig mange andre steder hvor man kan kaste bort tiden også…

Falske ip-adresser, «spoofing»

Vi kan også blokkere pakker med opplagt forfalskede fra-adresser, f.eks. pakker som kommer utenfra men har fra-adresser tilhørende interne nettverk eller ikke-rutbare nett som 127.0.0.0/8, 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16 og 169.254.0.0/16.

Slike pakker brukes i blant for å lure eldre brannmurer som bare ser på fra-adressen. En smart brannmur ser også på hvilket interface pakken kommer fra. Hvis den kommer fra en ekstern tilkobling, skal den ha en ekstern ip-adresse for å godtas. Og tilsvarende – en pakke fra et internt nett må ha en intern ip-adresse. (anti-spoofing)

Noen plukker også ut ip-adresser som ennå ikke er tildelt noen organisasjon. Det er farlig, for gjenværende ip-adresser blir stadig delt ut.

Kjente angrepsadresser

Hvis man har blitt angrepet før, kan det være en idé å blokkere de aktuelle adressene. Det finnes de som vedlikeholder slike svartelister på nettet. Men vær klar over at mange adresser er dynamiske. Adressen som tilhørte en hacker i går, kan tilhøre en kunde i dag.

Noen brannmurer løser dette ved å blokkere angripere en kort stund, f.eks. en time. Det er nok til at de fleste angripere går annensteds. Og prøver de på nytt, blokkeres de bare på nytt igjen.

Source routing

Source routing er en teknikk som nå er gammeldags, men som kan misbrukes. Som navnet antyder, bestemmes rutingen av den som sender pakkene, heller enn av ruterne underveis. Dette gjøres ved at hver pakke får med en liste over rutere den skal innom. En maskin som svarer på en slik pakke, sender svarpakken tilbake samme vei. Det får den til ved å snu lista over rutere før den sender pakka tilbake med source routing. Dette var nyttig i gamle dager, for å komme rundt feilkonfigurerte rutere. Teknikken er lite egnet nå som Internettet er så stort at de fleste ikke har oversikt over det. For hvem vet *hvor* det står en feilkonfigurert ruter i dag?

En angriper vil typisk misbruke source routing for å få frem pakker med falsk fra-adresse. Den falske fra-adressen brukes for å lure naive brannmurer som stoler på fra-adressen, eller tjenermaskiner som bare svarer på interne adresser.

Angriperens problem er at svaret på en slik henvendelse blir sendt til den falske fra-adressen. Dermed får han ikke se svaret, kommunikasjon blir umulig. Men source routing gir nye muligheter. Angriper kan f.eks. oppgi sin egen ip-adresse som en av ruterne pakka skal rutes innom. Dermed vil svar på henvendelsen sendes tilbake til angriper med source routing, og kan fanges opp med en pakkesniffer.

I praksis gir source routing hackerne mulighet for å forfalske en hvilken som helst ip-adresse på et hvilket som helst nett – og likevel får svar tilbake.

Dette unngår vi ved å ikke slippe inn pakker utenfra som bruker source routing. Det er som regel lurt å stoppe slike pakker om de kommer innenfra også.

Source routing kan også misbrukes til å øke belastningen på Internettet, ved å sende med en liste over rutere som alle ligger langt fra hverandre.

Annet, surfing fra sikre nett

I tillegg til porter og ip-adresser, kan vi vurdere andre felter i IP og TCP også. Et vanlig triks er å se på TCP-flaggene. Ofte ønsker vi å la brukere surfe på nettet, de må altså få lov

til å åpne forbindelser ut – og de må få ta imot svar. Men vi ønsker ikke å la utenforstående få lov til å åpne forbindelser inn. Dette kan vi gjøre ved å stoppe pakker utenfra som har SYN-flagget satt, men ikke også ACK. [1]

Dette gir samme sikkerhet som en NAT-ruter, som også gjør det umulig å åpne forbindelser utenfra.

Annet, umulige pakker

Pakkefilter kan også luke ut en del «umulige» pakker:

- Pakker med umulige kombinasjoner av flagg, f.eks. SYN+FIN, eller alle[2] eller ingen[3] av flaggene. Slike kombinasjoner forekommer ikke i standard kommunikasjon, så hvis en slik pakke dukker opp er det noen som prøver å lage problemer. Historisk har en del TCP-implementasjoner hatt problemer med slikt, og dermed gitt hackere muligheten til å kræsje fjerntliggende maskiner med slike tullepakker.
- Ugyldige fragmenter. Ip-pakker må i blant deles opp i fragmenter, og settes sammen igjen av mottaker. Et ugyldig fragment kan f.eks. være laget slik at mottaker setter sammen en pakke som blir for stor. (Større enn pakkestørrelsen, eller større enn max pakkestørrelse som er 64 kB.) Enkelte gamle operativsystemer kræsjer om de får slike pakker.

5.10.2 Pakkefilter med tilstand/sesjonsfilter

Et pakkefilter med tilstandsinformasjon, tar vare på informasjon om forbindelsene. Dermed kan vi gå mer avansert til verks. Nå kan pakkene vurderes ved å se dem i sammenheng med tidligere pakker. Et sesjonsfilter kan sjekke de samme tingene som det enkle pakkefilteret, og mer:

- Hvor mye tid har gått siden forrige pakke i denne sesjonen? Kommer de for tett indikerer de dos-angrep, kommer de for sjelden kan være et forsøk på å binde opp ressurser over lang tid.
- Øker sekvensnumrene/ACK-numrene som forventet? Hvis ikke kan det være falske pakker eller forsøk på å gjette sekvensnumre.
- Pakker som ikke er del av en eksisterende forbindelse kan avvises. Slike pakker kan være forsøk på å utnytte kjente feil i enkelte TCP/IP-implementasjoner.

[1] En ny forbindelse åpnes ved å sende SYN, den andre enden svarer med SYN+ACK. Deretter utveksles flere pakker, men ingen av disse inneholder SYN. For mer informasjon om dette, se faget «Datakommunikasjon» og «three-way handshake»

[2] En TCP-pakke med alle flaggene satt kalles gjerne en «christmas tree packet».

[3] En pakke uten flagg kalles en NULL-pakke.

- Serier med fragmenterte pakker kan settes sammen før pakka sendes videre. Dette for å være sikker på at det ikke er noe tull med fragmenteringen.

  Dette er nyttig også av en annen grunn. Fragmenterte pakker kan være delt opp midt i portnummeret, eller noe annet vi ønsker å teste. Ved å sette sammen fragmentene, blir det mulig å teste pakka, noe som ellers ikke ville vært mulig. Hackere bruker i blant fragmentering for å lure brannmurer som ikke setter sammen fragmenter.

- Noen protokoller krever spesielle tiltak, f.eks. FTP[4]. Når du laster ned en fil med FTP, vil maskinen din åpne en port for lytting. Portnummeret sendes via den normale forbindelsen til FTP-tjeneren, som deretter åpner en forbindelse *inn* på din maskin til denne porten for å overføre filen. Hvis vi ønsker en streng brannmur, vil vi ikke la alle porter stå åpne for tilgang utenfra, bare fordi FTP kan komme til å trenge dem. Men en brannmur som forstår protokollen, kan gjøre dette:

  1. Legg merke til portnummeret når du prøver å laste ned en fil. Legg også merke til FTP-tjenerens IP-adresse.
  2. Åpne tilgang for det aktuelle portnummeret for den aktuelle tjenermaskinen, med en passende timeout.
  3. Stenge porten igjen når overføringen er slutt.

5.10.3 Innholdsfiltre

Filtrering av e-post med tanke på virus/spam, blokkering/logging av URLer, og blokkering av java/activex fra nettsider er alt sammen eksempler på innholdsfiltrering. For å få til dette må brannmuren være i stand til å samle opp flere pakker tilhørende en forbindelse og forstå innholdet. Epost kan ikke sjekkes ved å se på en og en pakke, da en virussignatur kan strekke seg over flere pakker. Tilsvarende kan en http-strøm med en uønsket URL være delt opp.

Epost er såpass komplisert at det som regel filtreres på eposttjeneren. Det meste av epostprogramvare har mulighet for å kjøre brevene gjennom eksterne filterprogrammer. Dermed kan man luke ut virus på signaturer, og søppelpost ut fra frekvensen på «tvilsomme ord». Noen gjør det enkelt, og kaster alle eksekverbare vedlegg. Lett å sette opp, om man ikke har behov for å motta programvare i eposten.

Nettsider kan filtreres med en proxy, som f.eks. kan kjøre virussjekk på nedlastet materiale. Eller simpelthen blokkere eksekverbare filer, da det er mye enklere. Om ønskelig kan en også blokkere java/javascript.

[4] File Transfer Protocol

Nettsider kan til en viss grad også filtreres i nettleseren. Det har etterhvert dukket opp plugins som ghostery, noscript, adblock m.fl. som kan luke ut mye uønsket materiale. Slike programmer er ofte innrettet mot «privatlivets fred», og kan hindre servere i å spore surfingen din via cookies.

Vi har allerede sett at FTP kan trenge avansert filtrering, ved hjelp av filtre som forstår protokollen. Tilsvarende finnes multimediaprotokoller som bruker en TCP-forbindelse til å velge innhold men UDP for å overføre innholdet, f.eks. filmer. En avansert brannmur kan følge med på hva som skjer på protokollnivå, og tilpasse seg.

Innholdsfiltre kan også kjenne igjen protokoller som kjøres på uvante portnumre. De kan f.eks. oppdage torrents, uansett hvilke portnumre de bruker. Dette fordi de ser på innholdet i pakkene, og matcher det opp mot hvordan torrent-protokollen fungerer.

Et filter som ser på innholdet i pakkene, kan komme på kant med regler for personvern. I Norge må man gjøre brukere oppmerksom på at man har et slikt filter, på samme måte som at man må gjøre oppmerksom på videoovervåking.

5.10.4 Personlige brannmurer

Pakkefiltre kan kjøre på enkeltmaskiner også, og gi ekstra beskyttelse. F.eks. hvis en maskin innenfor brannmuren har fått virus.

En slik brannmur kan også se på hvilke programmer som kommuniserer, og gi meldinger av typen «program x prøvde å kontakte zz.com på port y». Så kan man velge om man vil tillate eller ikke. Å holde på slik blir vel fort for masete for de fleste, men det kan være nyttig for å bygge opp et brannmuroppsett som senere kan distribueres til brukermaskiner.

Vær oppmerksom på at regler som bruker navnet på programmer, lett kan lures ved å bytte navn på programfilen.

5.10.5 Utgående filter

En god brannmur filtrerer ikke bare det som kommer inn. Den bør også se på utgående trafikk, fordi:

- En intern maskin kan rammes av virus, hackes eller innlemmes i et botnett. Vi vil ikke at en slik maskin skal kunne angripe utenforstående. I første omgang får firmaet som eier maskinen, skyld for angrepet. Om en da vrir seg unna ansvar ved å vise til at maskinen var kompromittert utenfra, viser en for all verden hvor dårlig datasikkerhet firmaet har. Uansett ender en opp med et dårlig rykte.

- Spyware og keyloggere sender hjem «rapporter». Det er selvfølgelig fint om en får stoppet slikt. Oppdager brannmuren en keylogger, kan den varsle driftspersonell som kan reparere/reinstallere den aktuelle maskinen. I praksis er ikke dette så lett å oppdage, som oftest kontakter spyware port 80 på en utenforstående maskin. For brannmuren vil det dermed nesten se ut som vanlig nettsurfing.

5.10.6 Autentisering

Autentisering bør brukes på all infrastruktur som har webgrensesnitt. Vi ønsker ikke at hvem som helst skal kunne se på f.eks. ruteroppsett, eller enda verre – endre på det. Vær oppmerksom på unntakene, på IIE ligger f.eks en god del slik informasjon åpent, så studentene skal kunne se på nettet vårt som et praktisk eksempel. Å holde det hemmelig kan imidlertid gi noe ekstra sikkerhet, da det er vanskeligere å angripe et nett når man ikke kjenner den interne organiseringen i det hele tatt.

De fleste bokser med webgrensesnitt har en eller annen form for autentisering. Hvis dette ikke er bra nok, kan brannmuren steppe inn i stedet. HTTP har jo en mekanisme for autentisering, og brannmuren kan ta i bruk denne selv. Dermed kommer man ikke videre til webgrensensittet før man har autentisert seg overfor brannmuren.

Tilsvarende triks kan gjøres med andre protokoller som har autentisering, f.eks. FTP.

5.10.7 NAT

NAT[5] er en betegnelse som dekker flere fenomener. Hjemmebrukere kjenner til NAT som et opplegg som lar flere PCer dele på en IP-adresse. Dette gir også en viss grad av sikkerhet, da det samtidig blir umulig å opprette forbindelser utenfra og inn. Hvis man trenger slik funksjonalitet, er brannmuren et bra sted å gjøre det.

NAT beskytter ikke mot alt. Spesielt gjøres det ingenting med utgående trafikk. NAT beskytter mot oppkoblinger utenfra, men en har altså likevel behov for brannmurfunksjoner. Brannmuren kan hindre virus i å komme inn via epost og nettsider, og den kan holde utkikk etter spyware som sender informasjon hjem.

5.11 Tilhørende kapitler i Innføring i Datakommunikasjon

Kapittel	Pensum	Navn
5	Ja	Nettverkslaget kap. 5.1–5.7
10	Ja	IP versjon 6, kap. 10.3.2

[5] Network Address Translation