Kommunikasjon - Tjenester og Nett

Chapter 1 - Computer networks and the internett

1.1 What is the internet?

1.1.1 A Nuts-and-Bolts Description

- The Internet is a computer network that interconnects billions of computing devices throughout the world. Devices that are hooked up to the internet is called **hosts** or **end systems**.
- Transmission rate - measure of how fast links can transmit data in bits/second.
- Packet switches - takes packets from one destination to another. Two most common types in today's internet is **routers** and l**ink-layer switches.**
    - Link-Layer switches are typically used in access networks
    - Routers are typically used in network core.
    - The sequence of communication links and packet switching is known as route or path through the network
- Endepunktene aksesserer internett gjennom **Internet Service Providers** (ISP). Finnes mange type ISPer, og hver ISP i seg selv er et nettverk av packet switches og communication links.
- End systems, packet switches og andre bruker **protokoller** som kontrollerer sendingen og mottakingen av informasjon på internett.
    - Transmission Control Protocol (TCP)
    - Internett Protocol (IP) - bestemmer formatet på pakkene som sendes og mottas
- Felles standarder for hva protokollene gjør, IETF standarden kalles request for comments (RFC)

1.1.2 A Service Description

- Applikasjonene som bruker internett kalles distributed applications, siden de involverer flere endepunkter som flytter data mellom hverandre
- Endesystemene koblet til internett utgjør et **socket interface** som spesifiserer hvordan et program som kjører på den ene endesystemet spør internett-infrastrukturen om å levere data til et gitt destinasjonprogram som kjører på den andre siden

1.1.3 What is a Protocol?

- All aktivitet på internett som involverer to eller flere komponenter bruker protokoller
- En protokoll definerer formatet og rekkefølgen på meldinger utvekslet mellom to eller flere kommuniserende enheter, i tillegg til handlingene utført ved transmittering eller mottak av en meldinger eller en hendelse.

1.2 Network edge

- Endesystemene i et nettverk kalles hosts, host = end system
- Hosts kan deles inn i klienter og servere
    - Klienter er normalt sett laptoper, pcer, mobiler
    - Servere er mer kraftfulle maskiner som lagrer og distribuerer websider, streamer video. I dag befinner disse seg i store datasentre
- Internet of Things
    - Alt er koblet på internett, kjøleskap, biler, sykler, briller, klokker, leker osv

1.2.1 Access Networks

- Mest vanlige type bredbånd er gjennom **Digital subscriber line** (DSL) og kabel
    - Får vanligvis DSL internett aksess via telefoni, så når DSL er brukt er telefonselskapet også ISPen.
- Fra pc til modem til splitter (separerer telefonsignal og datasignal) til digital subscriber line access multiplexer (DSLAM) som befinner seg hos teleselskapet til internett
- Båndbredde - bits per sekund - avhenger av delt eller dedikert
- Fiber og coax brukes også, hybrid fiver coax (HFC)
    - Fiber to the home (FTTH) er en direkte fiberkabel til CO (central office) fra hjemmet
- Vi har to typer fiber
    - Active optical networks (AON)
    - Passive optical networks (PON)
- Kan også bruke satelittsignaler for å få internett steder der kabel og DSL ikke er tilgjengelig
- Kabel internetttilgang krever spesielle modem, kalt kabelmodem. Er ofte en egen device som man kobler til pcen med Ethernet port.
- Når flere bruker samme internett, fungerer det slik at alle sapper sendt fra et endepunkt sendes på samme downstream til hvert hjem, og motsatt. Dette gjør at nettet blir tregere hvis flere er aktive samtidig.
- **Local Area Network (LAN)** kobler et endesystem til en enderuter.
    - Ethernet er den mest vanlige teknologien, kobler kabelen til en Ethernet svitsj

    - Switchen kan koble seg på mange andre pcer og samtidig internett
- WLAN - Wireless LAN - basert på Wi-Fi tekonologi
- 3G og LTE/4G har mye lengre rekkevidde og benyttes derfor når klienten ikke er i nærheten av et fast punkt

1.2.2 Physical Media

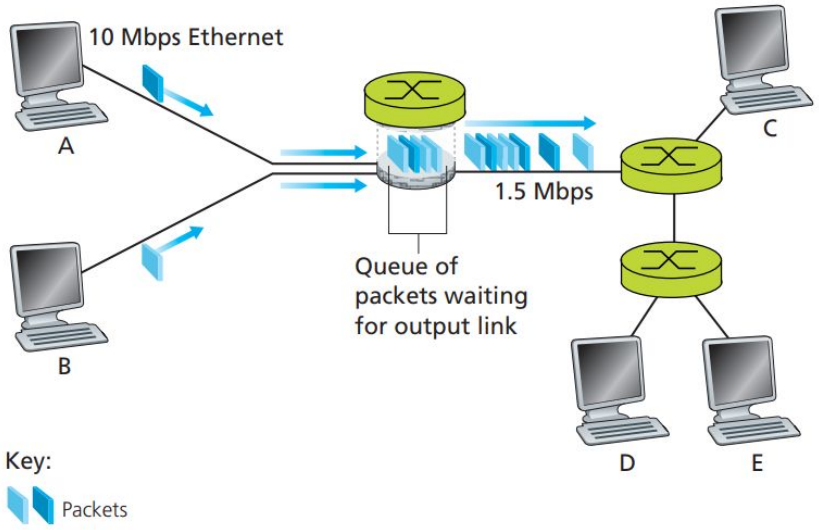
- Typer fysisk medium - coax kabel, twisted-pair kobberledning, fiber optic kabel, radio spectrum, satelitt radio spectrum
- Guided Media og Unguided Media
    - Guided Media - bølgene er guidet gjennom et solid medium, som feks kabel
    - Unguided Media - bølger som utsprer seg i atmosfæren, wireless LAN og satelitt
- Twisted-pair kobberledning - mest brukt og billigste, twisted for å redusere interferens fra lignende kabler i nærheten.
- Unshielded twisted pair (UTP) brukt for computer nettverk i bygninger, feks LAN.
- Coax kabel - har også kobber, raskere og bedre laget kabel
- Fiber optics - tynn og fleksibel, sender lyspulser som representerer bits, immune mot elektromagnetisk interferens
- Bakkesendt radio kanaler - elektromagnetiske bølger
- Satelitt - linker flere bakkestasjoner. geostationary satelitter holder seg på samme sted hele tiden, LEO satelitter går i bane rundt jorden og kan kommunisere seg i mellom

1.3 Network core

1.3.1 Packet Switching

- Pakker - små mengder data. Transmitteres over hver kommunikasjonslink i en rate lik den fulle transmisjonsraten til linken. Bruker store-and-forward transmission i inputene til linkene.
    - Betyr at pakkesvitsjen må motta hele pakken før det første bitet kan sendes videre
- Tiden det tar å sende L bits over en link med overføringshastighet R bits/sek er gitt ved L/R.
    - Ved tid 0 har man startet å transmittere pakken
    - Ved tid L/R har kilden transmittert hele pakken, men ruteren har bare mottatt og lagret dette. Begynner nå å trasmittere packet på output link mot målet.
    - Ved 2L/R tid har den transmittert hele pakken, og hele pakken er mottatt hos mottakeren. Total delay er altså 2L/R.
- dend–to–end = NL/R - tiden for delay der man har N linker med N-1 rutere

- Har en output buffer/kø der lagrer pakkene som ruteren skal sende videre inn i en link. Hvis noe skal sendes videre, men linken er opptatt, må pakken vente i køen. Det gjør at man også har kø-delays i tillegg til store-and-forward delays
- Hvis bufferet fylles opp, og det kommer en ny pakke, får vi packet loss. Da droppes enten den nye pakken, eller en pakke i køen



-
- Når en pakke ankommer ruteren, slår ruteren opp i sin forwarding table, som mapper destinasjonsadressen til ruterens utgående link, for å finne ut av hvilken utgående forbindelse pakken skal sendes videre på
    - Routing er prosessen som lager forwarding tables som skal brukes på veien. En routing protocol kan feks bestemme korteste vei fra hver ruter til hver destinasjon og bruke resultatet til å lage forwarding table i ruterne

1.3.2 Circuit Switching (linjesvitsjing)

- I linjesvitsjing må senderen reservere hele tilkoblingen når det skal brukes. For eksempel ved telefonringing.
- Man er garantert en konstant rate for å sende data i motsetning til pakkesvitsjing (der det kan oppstå kø osv)
- Implementeres med enten FDM (frequency-divison multiplexing) eller TDM (time-division multiplexing)
    - FDM - reserverer én bestemt frekvens
    - TDM - reserverer et bestemt tidsintervall

1.3.3 A network of networks

- Har en access ISP og en global ISP. Alle ISPene i verden er ikke koblet sammen direkte, det ville ikke lønnet seg.
    - Access ISP betaler til en global transit ISP, og kalles customer, mens global transit ISP kalles provider

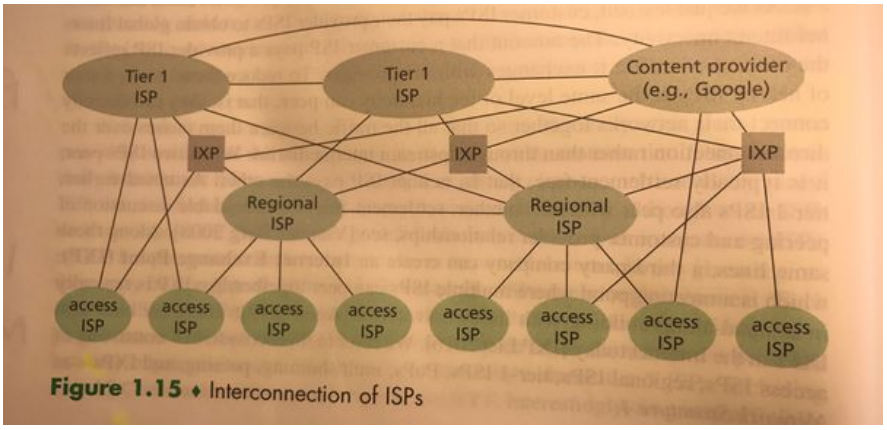
- For å bygge et nettverk som ligner dagens Internett, må vi legge til Points of Presence (PoPs - gruppe av en eller flere rutere på samme lokasjon i en providers nettverk), multi-homing (koble til to eller flere provider ISPer), peering (at flere customer ISPer kobler sammen, peerer, slik at all trafikken mellom dem går gjennom direkte kobling) og Internet exchange points (IXPs - møtepunkt for flere ISPer kan peere).
- Siste del i "nettverk av nettverk" består av å legge til content-provider nettverk (feks Google).
- Totalt består det av ca et dusin Tier-1 ISPer som alle er koblet sammen, og som leverer til lavere ISP-nivåer som regional ISP og access ISP
- 

Figure 1.15 ♦ Interconnection of ISPs

1.4 Packet-switched networks

1.4.1 Overview of Delay in packet switched networks

- Fire hovedtyper av forsinkelse, total forsinkelse er summen av disse
- Prosesseringsforsinkelse - processing (< mikrosekunder)
  - Tiden det tar å undersøke pakkens header og hvor den skal videresendes
- Køforsinkelse - queue (< milisek)
  - Tiden det tar å vente i kø på at andre pakker skal bli sendt ut på linken før deg
- Overføringsforsinkelse - transmission(<milisek)
  - Tiden det tar å skyve pakken ut på linken (som er veien mellom nodene). Dette tar L/R tid.
- Forplantningsforsinkelse - propagation (<milisek)
  - Tiden det tar å reise fra en node til neste node. Tar d/s tid.
- Nodal delay er gitt ved d_(end-to-end)= d_proc + d_queue + d_trans + d_prop
- Total end-to-end-delay er tiden det tar fra start til slutt, altså veien gjennom alle ruterne. Denne delayen er:
  - d_(end-to-end)= N(d_proc+d_trans+d_prop)

    - I en kobling med flere switcher mellom den, brukes transmission delay som flaskehals
- En viktig faktor i queuing delay er traffic intensity.
    - Dette er forholdet La/R, der a er pakker per sekund og L er bit, R er transmission rate.
    - Dersom La/R > 1 vil det komme flere pakker inn til køen enn køen klarer å sende videre, og queuing delay vil stige mot uendelig
    - Ønskelig å designe et nettverkssystem der traffic intensity aldri blir over 1.
- Instantaneous througput til enhver tid er raten man mottar en fil med akkurat der og da, målt i bits/sek.
    - Dersom en fil består av F bits og det tar T sekunder å motta den så er average throughtput F/T
-

1.5 Protocol layers and service models

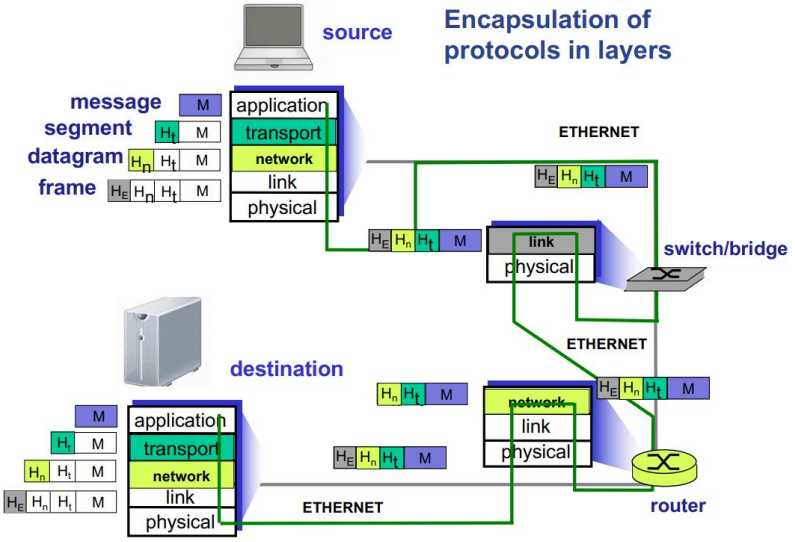
1.5.1 Layered Architecture

- Internettprotokollen er delt inn i 5 lag

Navn	Forklaring	Eksempler
Application layer	Toppnivå-protokoller som direkte samhandler med programvare.	HTTP, SMTP, FTP, DNS
Transport layer	Sørger for end-to-end kommunikasjon.	TCP, UDP
Network layer	Fikser alt som ligger mellom hosts, eks. hvor en pakke skal sendes videre for å komme til sin destinasjon. Dette er det øverste laget i en router.	IP
Link Layer	Protokoller for hvordan data skal legges over på det fysiske laget.	Ethernet, 802.11b/g/n
Physical layer	Det fysiske, altså typen kabel eller antenne.	Twisted-pair, Nettverksadapter, Fiber ..

-
- Syv-lags ISO OSI har i tillegg Presentation og Session som lag 2 og 3.
    - Presentation - lar applikasjoner tolke meningen med dataene, feks kryptering, kompresjon osv
    - Session - synkronisering, recovery av data exchange

1.5.2 Encapsulation



- Ved source sender application layer en message til transport layer. Her legges til en header fra transport layer, og sammen med message utgjør de transport layer segment
- Sendes deretter videre til network layer, som legger på en header. Vi har nå et datagram. Sendes videre til link-layer, som også legger på en header, og kalles nå frame.

1.6 Networks under attack - security

- Malware er skadelig programvare som hackere kan implementere i våre enheter over nett
  - Mye av malwaren er self-replicating - hvis én host blir rammet, vil den jobbe seg videre fra denne hosten til andre hosts. Kan dermed spre seg eksponentielt fort
  - Botnet er et stort nettverk av flere tusen enheter som også er kompromittert
- Virus er malware som trenger en form for brukerinteraksjon for å infisere enheten
- Ormer er malware som kan gå inn i en enhet uten en brukerinteraksjon.
- Denial-of-service (DoS) angrep er et angrep der man hindrer noe eller noen å få tilgang til informasjon eller ressurser de vil ha tilgang til. Har tre kategorier
  - Sårbarhetsangrep - sender noen få, vell-smidde meldinger til en sårbar applikasjon
  - Båndbredde flom - sender en mengde pakker til målet, så mange pakker at målet ikke klarer å ta i mot alt, crasher
  - Forbindelse flom - angriperen lager en stor mengde halv-åpne eller full-åpne TCP forbindelser mot målet

- Distributed Dos (DDoS), angriper styrer flere kilder og sender traffikk fra alle mot målet samtidig. Vanskeligere å oppdage og beskytte seg mot i forhold til DoS.
- Packet sniffer er en som lagrer en kopi av alle pakkene som sendes mellom to enheter
- IP Spoofing er at man utgir seg for å være en person man kan stole på ved å falsifisere data. Kan løse dette ved en-point autentisering, en mekanisme som lar oss bestemme med sikkerhet om en melding kommer fra hvor vi tror den gjør.

1.7 History of computer networking and the Internet

- Utviklingen av pakkesvitsjing 1961-1972
    - ARPAnet, første pakkesvitsjings datanettverk og forfader til dagens internett (1969 ish)
- Proprietære nettverk og internettworking 1972-1980
    - TCP, IP og UDP kom på plass på slutten av 70-tallet.
- Spredning av nettverk 1980-1990
    - Hundretusener var nå koblet opp mot ARPAnet
    - Minitel - et prosjekt om å få internett inn i husstandene
- Internetteksplosjonen 90-tallet
    - World Wide Web kom, oppfunnet på CERN
    - HTML, HTTP og Web server kom
    - Email, chat, peer-to-peer fildeling
- The new millennium
    - Sånn det er i dag :)

Chapter 2 - Application Layer

2.1.1 Network Application Architectures

- Applikasjonsarkitekturen er designet av applikasjonsutvikleren og forteller hvordan applikasjonen er strukturert over de forskjellige endepunktene.
- Har to typer applikasjonsarkitekturer
    - Klient-tjener
        - Består av en alltid-på-host kalt tjener som behandler forespørsler fra andre hoster kalt klienter
        - Serveren har én bestemt IP-adresse, som alltid kan kontaktes ved å sende en pakke til denne adressen.
        - Problemer kan oppstå dersom svært mange klienter prøver å koble seg opp mot tjeneren samtidig. Da kan nettsiden kræsje.
        - Populære nettsider som Google, Facebook osv håndterer dette ved å benytte et data center som setter opp en virtuell tjener bestående av en større mengde tjenere.

- P2P - Peer-to-peer
  - Minimalt eller ikke-eksisterende avhengighet av dedikerte servere
  - Benytter seg heller av dirkete kommunikasjon mellom par av koblede hosts, kalt peers. Dette er typisk laptoper eid av vanlige brukere
  - Brukes av blant annet BitTorrent og Skype
  - Fordelen ved P2P er at man ikke trenger en tjener tilkoblet til en hver tid, og peers tar rollen som tjener ved å distribuere filer til andre peers. Dette er konstnadseffektivt da man ikke trenger mye infrastruktur eller båndbredde til en tjener.
  - Ulemper er blant annet sikkerhet, ytelse og pålitelighet
  - Self-scalability, hver peer legger til service kapasitet til systemet ved å distribuere filer til andre peers

2.1.2 Process Communicating

- En prosess må kommunisere gjennom minst én socket. En socket er et grensesnitt mellom applikasjons -og transportlageret. Hver socket er koblet til et portnummer som er unikt for datamaskinen.
  - På denne måten er det mulig å vite hvilket program som skal motta pakker etter at de har kommet til riktig IP adresse.
  - Analogi: Prosessen er huset, socket er døren. Skal du fra hus til hus må du gjennom døren. Man antar at det er noe som kan transportere meldingen videre til neste socket og prosess(transportlaget)
  - Alle meldinger som sendes må derfor ha både IP-adresse og portnummer
  - Application Programming Interface (API) - interfacet mellom applikasjonslaget og transportlaget

2.1.3 Transport Services Available to Applications

- Tjenester transportlaget kan tilby til applikasjonslaget
  - Pålitelig dataoverføring
    - Dataen som er sendt garanteres å komme frem dit den skal
  - Gjennomstrømming
    - Garanterer gjennomstrømming med en spesifikk hastighet, for eksempel r bits/sek
    - Applikasjoner med gjennomstrømmingskrav kalles båndbredde-sensitive applikasjoner
    - Elastiske applikasjoner bruker den gjennomstrømmingen som er tilgjengelig.

    - Timing
        - Kan for eksempel garantere at hver bit som sendes over socketen kommer frem senest 100msec senere
    - Sikkerhet
        - Kan kryptere data

2.1.4 Transport Services Provided by the Internet

- TCP - Transmission Control Protocol
    - Forsikrer at serveren finnes og at pakkene kommer i riktig rekkefølge uten tap.
    - Er connection-oriented (bruker 3-veis handshaking) og kan overføre begge veiene
    - Klienten og tjeneren har altså på forhånd utvekslet informasjon vedrørende transportoverføringen med hverandre
    - Tilbyr pålitelig overføring av data (uten error og i rett orden)
    - Tilbyr også congestion-control
    - Brukes av mail (SMTP), nettet (HTTP), filoverføring (FTP) og noe streaming
- UDP - User Datagram Protocol
    - Tilbyr minimalt med tjenester
    - Er connection-less (ingen handshaking)
    - Er upålitelig, tilbyr ingen garanti for at meldingen når frem
    - Tilbyr ingen kontroll over gjennomstrømming (congestion control)
    - Senderen kan pumpe data til hvilken rate den måtte ønske, men den kan begrenses av båndbredden på lenkene
    - Brukes iblant av multimedia applikasjoner som kan tolerere noen form for tap og IP-telefoni
- Hverken TCP eller UDP bruker kryptering, derfor er SSL (Secure Sockets Layer) utviklet, er en enhancement til TCP.

2.1.5 Application-Layer Protocols

- En applikasjonsnivå-protokoll definerer hvordan en applikasjon prosesserer og kjører på forskjellige endesystemer. Gjør dette gjennom:
    - Type melding som sendes (respons eller request)
    - Syntaksen til meldingen
    - Meningen av informasjonen i feltene
    - Regler for når og hvordan en prosess sender og svarer på meldinger
- Det er viktig å huske på at en applikasjonslag-protokoll kun er en del av nettverksapplikasjonen. Nettets applikasjonslag-protokoll kalles http.

2.2 The Web and HTTP

2.2.1 Overview of HTTP

- Http (HyperText Transfer Protocol) er nettets applikasjonslagprotokoll og er implementert både i klienter og tjenere. Det definerer hvordan nettklienter forespør nettsider fra servere, og hvordan servere overfører nettsider til klienter
    - Brukes til å overføre informasjon om nettsider. Når klienten spør om en nettside, sender serveren først HTML-filen, og andre objekter i hver sin pakke etterpå.
    - Støtter coockies og web caching
    - Bruker TCP som underliggende transportprotokoll
- Hver URL har to deler; hostname og vei
- HTTP tjeneren holder ingen form for info om klientene, og kalles derfor tilstandsløs protokoll (stateless protocol). Hver gang de kobles opp mot hverandre, behandles det som om det er første gang de har "møttes".
- Connectionless: klienten sender forespørsel til server, og disconnecter etter det. Connection åpnes igjen når server svarer
- Kan levere hvilken som helst data, så lenge begge datamaskinene kan lese det
- Består av 3 deler:
    - Start line, headers, body
    - Request -og responsmeldinger er forskjellige

2.2.2 Non-Persistent and Persistent Connections

- Kan ha ikke-vedvarende og vedvarende koblinger (persistant og non-persistant)
    - Ved vedvarende kan flere forespørsler sendes over samme TCP-kobling
    - Ikke-vedvarende tar som regel to RTTer (en til å opprette og en til å sende)
- Når en klient forespør en nettside med 11 objekter der det benyttes en HTTP med en ikke-vedvarende tilkobling må det settes opp 11 TCP koblinger
    - Kan derimot benytte seg av parallelle tilkoblinger som innebærer at det settes opp flere TCP koblinger parallelt slik at man slipper å sette opp alle 11 i serie. Dermed reduserer man responstiden
- RTT er tiden det tar for en liten pakke å reise fra en klient til en tjener, og så tilbake til klienten igjen. Inkluderer PROP, QUE & PROC.
- HTTP har persistant connection som standard

2.2.3 HTTP Message Format

- Finnes to typer http meldingsformater

- Forespørsler

```
GET   /somedir/page.html      HTTP/1.1    (request line)
Host:  www.someschool.edu                 (header line)
Connection: close                         (header line)
User-agent:        Mozilla/5.0            (header line)
Accept-language:   fr                     (header line)
```

- Svar

```
HTTP/1.1    200    OK                                  (status line)
Connection: close                                      (header line)
Date:   Sat, 07 Jul 2007 12:00:15 GMT                  (header line)
Server: Apache/1.3.0 (Unix)                            (header line)
Last Modified: Sun, 6 May 2007 09:23:24 GMT            (header line)
Content-Length: 6821                                   (header line)
Content-Type: text/html                                (header line)
Enitity Body:   data data data...                      (entity body)
```

- Cookies brukes for å identifisere brukeren. Har fire komponenter
  - Header line i http svar-meldingen
  - Header line i http forespør-meldingen
  - Fil på brukerens endesystem som styres av nettleseren
  - En database på nettsiden

2.2.5 Web Caching

- Web cache er en nettverksentitet som kan besvare HTTP-forespørsler fra klienter. Den ser om den har objektene klienten ber om lokalt, og hvis ikke sender den selv en request til serveren og henter disse objektene.
- Når en klient skal aksessere en nettside, benytter tjeneren seg av web cache (proxy-server) som har egen disklagring med kopier av nylig forespurte objekter
  - Fungerer både som tjener og klient
  - Kan redusere responstiden kraftig
    - Blir kjøpt og installert av ISP-er
    - Går fortere fordi tilkoblingen mellom klient og cache gjerne har større båndbredde enn den ut mot nettet (Nettet på NTNU er raskere internt enn hvis du må hente data utenfra)
  - Kan også redusere trafikken på lenkene til internettet

        - Bruker conditional GET med *if-modified-since*
            - Mekanisme i http som lar en cache verifisere at objektene er up to date
        - Content Distribution Networks (CDNs) gjør at web cache spiller en viktig rolle på Internett. Et CDN selskap installerer mange geografisk distribuert cacher på internett. Har delt CDN og dedikert CDN.

2.3 Electronic Mail in the Internet

- Mail har tre komponenter
    - Brukeragenter
        - Tilbyr brukere å lese, skrive, videresende og sende mailer
        - Outlook og Apple Mail er eksempler
    - Mailservere
        - Kjernen i mailinfrastrukturen
        - Hver bruker har en mailboks lagret på en mailserver
        - Dersom avsenders tjener ikke klarer å legge mailen inn i mottakers tjener, prøver den på nytt hvert 30 min.
    - SMTP - Simple Mail Transfer Protocol
        - Bruker pålitelig dataoverføring gjennom TCP
        - Er en push-protokoll
        - Hele mailen blir omgjort til 7-bits ASCII fordi den er gammel
        - Sørger for at mailen blir sendt fra avsenderens mailtjeneser til mottakerens mailtjener
        - Bruker ikke mellomliggende mailtjenere for å sende en mail. Dersom en mail sendes fra Hong Kong til Chicago så er TCP koblingen en direkte kobling mellom Hong Kong og Chicago (den går ikke via noen andre steder)
        - Fungerer ved at den først etablerer en TCP kobling til port 25, handshaker og sender meldingen
        - Bruker vedvarende kobling

2.3.2 Comparison with HTTP

- Http er en pull protokoll (den henter ned fra internett), mens SMTP er en push protokoll (pusher til mailserver)
- SMTP må motta alt på 7-bits ASCII

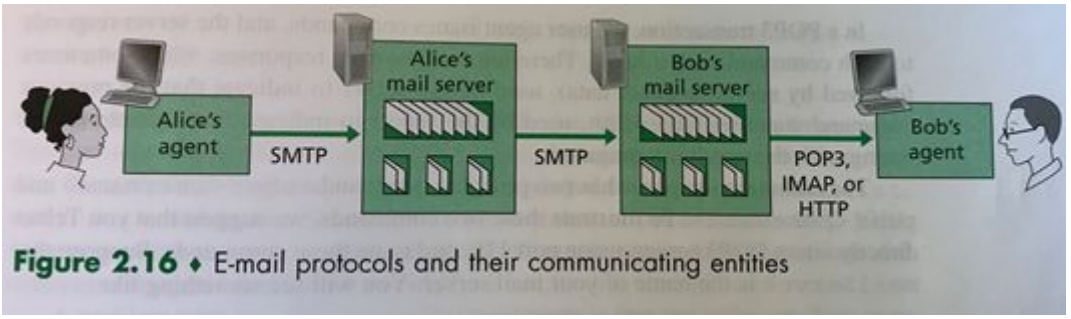
- 

Figure 2.16 ♦ E-mail protocols and their communicating entities
    - Bruke POP3 eller IMAP

2.3.4 Mail Access Protocols

- POP3 - Post office protocol
    - Veldig enkel protokoll med begrenset funksjonalitet
    - Vanlig å bruke hvis du sjekker mail vanligvis bare fra én bestemt host, feks pcen din hjemme.
    - Etter at brukeragenten har satt opp en TCP tilkobling til mailtjeneren, går POP3 gjennom følgende faser:
        - Autorisering (sender brukernavn og passord)
        - Transaksjon (mottar meldingen og kan markere mail for sletting)
        - Oppdatering (etter klienten har skrevet quit - sletter meldingene fra over)
    - Brukeragenten sender kommandoer, og serveren svarer på alle kommandoer (+OK eller -ERR)
    - Mailen slettes fra serveren når den lastes ned på én klient
- IMAP - Internet Message Access Protocol
    - Mer komplisert ved at den assosierer hver mail med en mappe (INBOX-mappa først når mailen kommer)
    - Holder et mappehierarki på en ekstern server som gjør at mappene vedlikeholdes i mailen uansett enhet (dvs på forskjellige maskiner, iPad, telefon)
- Har også webbasert email, hvor http brukes til å kommunisere
- File Transfer Protocol (FTP)
    - Brukes for å overføre filer på nett
    - Bruker to TCP-forbindelser samtidig
        - Blir sendt data over den ene, mens den andre tar av seg autorisering, endring av mapper på serveren og lignende
    - I motsetning til HTTP, må FTP vite om tilstanden til brukeren

2.4 DNS - The Internet's Directory Service

2.4.1 Services Provided by DNS

- Domain Name System - DNS

- Oversetter domenenavn til IP-adresser
    - Grunnet forskjellig lengde og standarder, er dette vanskelig for rutere å behandle
    - Verter identifiseres derfor med IP-adresse
    - Består av 4 bytes, feks 121.7.106.83
    - Er hierarkisk ettersom når vi ser fra venstre mot høyre, får vi mer og mer info om hvor i internett verten er lokalisert
- Bruker UDP, port 53
- Bruker cache for å redusere forsinkelsen det tar å slå opp IP-adresser
    - Sletter infoen normalt etter to dager
- Tilbyr andre tjenester som
    - Host aliasing
        - En host med et komplisert hostname kan ha en eller flere alias som er koblet til den samme IP-adressen og dermed fører deg samme sted. Poenget med aliasene er at de er enklere. Et hostname som har flere alias (som er mindre kompliserte) kalles et kanonisk hostname
    - Mailtjener aliasing
        - Samme som over, men med mailadresser. DNS kan sørge for at en mailapplikasjon finner det kanoniske hostname til et alias hostname sammen med IP-adressen til hosten. Dermed kan aliaset bob@gmail.com brukes selv om mailadressen (den kanoniske) er relay1.west-coast@gmail.com
    - Load distribution
        - Høyt traffikerte nettsider som cnn.com kan repliseres over flere nett-tjenere, der hver tjener kjører på forskjellige endesystemer og har ulike IP-adresser. For slike nettsider har man altså en mengde IP-adresser som er koblet til det kanoniske hostnamet til siden (dette ligger lagret i DNS databasen). Dermed kan DNS distribuere trafikken blant de repliserte tjenerne. Dette brukes også på mail.

2.4.2 Overview of How DNS Works

- DNS er bygget opp av en stor mengde tjenere som er organisert på en hierarkisk måte rundt om hele verden. Vi skiller mellom tre (fire) klasser av DNS-servere

    - Rot DNS-tjenere
        - Over 400 av disse i verden
        - Gir IP-adresser til toppnivå-tjenere
    - Toppnivå-tjenere
        - For hvert toppdomene (com, uk, fr, no, net) finnes en slik
        - Gir IP-adressen for autoritative DNS serveren
    - Autoritære DNS-tjenere
        - Kontaktes tilslutt og tilbyr IP-adressen til vertsnavnet
        - Store organisasjoner med tilgjengelige verter må ha en slik
    - Lokale DNS tjenere
        - Tilbys av enhver ISP
- Kan sende både iterative og rekursive spørringer
- DNS caching brukes for å korte ned delays og redusere antall DNS meldniger som rikosjerer rundt på internett

2.4.3 DNS Records and Messages

- DNS tjenerne som sammen implementerer den distribuerte DNS databasen lagrer resource records (RRs) inkludert RRs som tilbyr hostname til IP-adresse mappinger
- En RR er en tuppel som inkluderer følgende felter: (Name, Value, Type, TTL), der TTL er levetiden i cache
    - Om type=A, så er Name et hostname og Value er den tilhørende IP-adressen
    - Om type=NS, så er Name et domene (foo.com) og Value et hostname til en autoritær DNS-tjener som da kan finne IP-adressen til hostene i domenet (det som står foran foo.com). Value kan da for eksempel være dns.foo.com
    - Om type=CNAME så er Value et kanonisk hostname for hosten sitt alias hostname Name.
    - Om type=MX så er Value et kanonisk navn til mailtjeneren som har et alias hostname Name
- Hvordan en DNS-melding er bygget opp

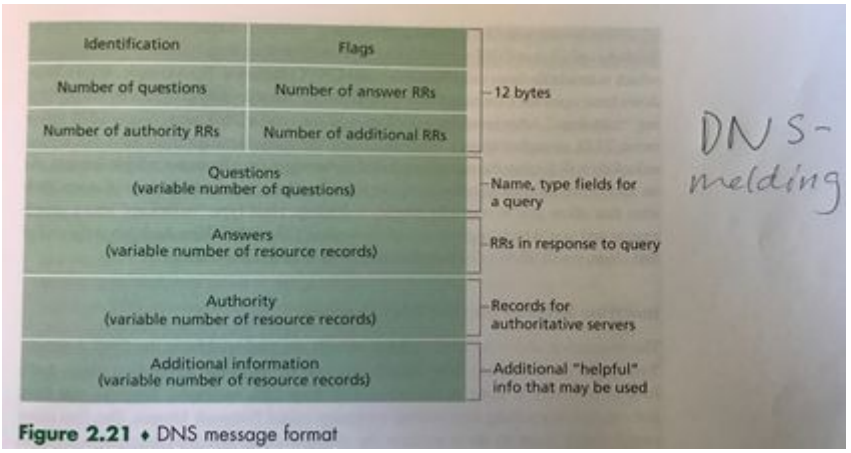
- 

Figure 2.21 ♦ DNS message format
    - Første 12 bytes består av header section, som identifiserer, har flagg og RRs
    - Question section består av info om spørring som blir laget, består av et navnfelt og et typefelt som identifiserer typen spørsmål som spørres om navnet
    - Answer section holder RRs for navnet som var spurt om
    - Authority section inneholder records av andre autoriserbare servere
    - Additional section inneholder andre nytte records.

2.5 Peer-to-peer

- I et P2P nettverk er alle klienter også servere
- Hver peer kan re-distribuere filen den mottar fra en annen peer slik at man assisterer tjeneren i distribusjonsprosessen.
    - Med vanlig fil-distribusjon vil tjeneren måtte sende en kopi fra filen til hver klient, noe som krever en stor mengde båndbredde hos tjeneren
- Distribusjonstiden (Distribution Time)
    - Tiden det tar å få sendt en kopi av en fil til alle N peers
    - La Us og Ui være opplastningstiden til hhv serveren og den i-te peeren, og Di er nedlastningstiden til den i-te peeren. Størrelsen på filen i bits settes til F. La oss først se på distribusjonstiden ved klient-tjener arkitektur D(cs).
        - D(cs) >= max {NF/Us , F/Dmin}
        - Tiden det tar å laste opp N*F bits til serveren er opplastningstiden Us
        - Tiden det tar for den tregeste klienten å laste ned F bits som distribueres Dmin er da min{D1, D2, ...}
    - Ser så på P2P:
        - Ved begynnelsen har kun serveren tilgang til filen. Serveren må altså til å begynne med å sende F bits til Internettet hvor

peersene kan få tilgang. Dermed er distribusjonstiden minst F/Us
    - Som ved klient-tjener kan ikke peeren med lavest nedlastningshastighet motta alle bitsene på mindre enn F/Dmin sekunder. Dermed er distribusjonstiden minst F/Dmin
    - Opplastningskapasiteten til hele systemet er opplastningskapasiteten til tjeneren + alle peersene, Utotal = Us + U1 +...+Un. Systemet må laste opp F bits til alle N peers. Kan ikke gjøres raskere enn
        - NF / Ut+SUM[1-n](Ui)
    - Min distribusjonstid for P2P er da:
        - Dp2p >= max {F/Ut, F/Dmin, NF / Ut+SUM[1-n](Ui)}

2.6 Video Streaming and Content Distribution Networks

2.6.1 Internet Video

- I http-streaming lagres videoen på en http-server som en vanlig fil med en URL
- Når en bruker ønsker å se filmen, etablerer den en TCP tilkobling med serveren og sender en http-GET spørring for den URLen. Tjeneren sender så filen med en tilbakemelding så fort som mulig
- Problem:
    - Alle mottar samme encodingen uavhengig av båndbredde
    - Derfor har man utviklet DASH (Dynamic Adaptive Streaming over HTTP)
    - Der blir videoen encodet til forskjellige versjoner med forskjellig bitrate
    - Den inkluderer også en manifest-fil som er en oversikt med URL til alle versjonene
- For å håndtere den massive etterspørselen etter videospilling, bruker de største strømmeselskapene CDN (Content Distribution Networks) som er lokalisert på mange geografiske lokasjoner med kopier av videoene.
- To server placement filosofier
    - Enter Deep - Gå dypt inn i aksessnettverket til ISP, ved å deployere serverclustere i aksess ISPer over hele verden
    - Bring Home - bringer ISP hjem ved å bygge store clustere på færre steder

Chapter 3 - Transport Layer

3.1 - Introduksjon og transportlag-tjenester

- Transportlagets sentrale rolle er å tilby kommunikasjonstjenester direkte til applikasjonsprossessene som kjører på forskjellige endesystemer

- Transportlagsprotokollene er implementert i endesystemene, ikke i ruterne.
- På sendersiden konverterer transportlaget meldingene den mottar fra avsenderens applikasjonsprosess til transportlags-pakker kalt segmenter.
    - Dette gjøres ved å dele meldingene inn i mindre biter og legge til en transportlags-header til hver bit.
    - Transportlaget sender segmentet videre til nettverkslaget på senderens endesystem hvor segmentet innkapsles med en nettverkslag-pakke (et datagram) og så sendes til destinasjonen.
    - På mottakersiden trekker nettverkslaget segmentet ut fra datagrammet og sender det til transportlaget. Transportlaget prosesserer så det mottatte segmentet og gjør dataen tilgjengelig for mottakerens applikasjon.
- Der transportlags-protokollen tilbyr logisk kommunikasjon mellom prosesser som kjører på forskjellige verter, tilbyr nettverkslags-protokoller logisk kommunikasjon mellom verter.
- En transportprotokoll kan tilby pålitelig dataoverføring selv om den underliggende nettverksprotokollen ikke tilbyr en slik tjeneste.
- Kort om IP
    - Best-effort, kan ikke garantere noe
    - TCP og UDP utvider vert-til-vert-kommunikasjon til prosess-til-prosess-kommunikasjon. Dette kalles transportlags-multipleksing og transportlags-demultipleksing

3.2 - Multipleksing og demultipleksing

- Jobben ved å levere dataen i transportlags-segmentet til korrekt socket kalles demultipleksing (analogi: samle tilsendte brev og dele ut til rette personene)
- Jobben ved å samle data på verten fra forskjellige sockets, innkaplse denne med header-info og sende dette til nettverkslaget kalles multipleksing (analogi: samle brev som skal sendes og gi til mailpersonen/nettverkslaget)
- Transportlagsmultipleksing krever
    - Socketene har unike identifikatorer
    - At hvert segment har spesielle felter som indikerer til hvilken socket segmentet skal sendes. Disse feltene er kildens portnummer og destinasjonens portnummer.
        - Hvert portnummer er et 16-bits tall mellom 0 og 65535. Portnr mellom 0 og 1023 er vekjente portnumre og er reservert av velkjente applikasjonsprotokoller som HTTP(80) og FTP(21)
- En UDP-socket er fullstendig identifiserbar av en tuppel bestående av destinasjonens IP-adresse og portnummer.
- TCP krever derimot i tillegg kildens IP-adresse og portnummer
    - En forskjell blir da at to segmenter med like destinasjonsportnumre og IP-adresser, men ulike for kildene vil ved UDP føres til samme socket,

men ved TCP føres til to forskjellige sockets (det opprettes en ny socket dersom ikke det finnes en med de fire variablene fra før).

3.3 - Forbindelsesløs transport: UDP

- UDP (User Datagram Protocol) er forbindelsesløs
- Ved siden av multipleksing/demultipleksing og noe feilsjekkeing, gir den ingenting mer til IP
- UDP tar meldinger fra applikasjonsprosessen, tilegner felter for kilde- og destinasjonsportnummer til multipleksing/demultipleksing tjenesten, legget til to andre felter og sender det resulterende segmentet til nettverkslaget.
    - Nettverkslaget innkaplser så segmentet til et IP datagram og gjør sitt beste forsøk på å sende det til den mottakende hosten.
    - Ingen handshaking i UDP mellom sendende og mottakende transportlags-entiteter før segmentet sendes, og UDP sies derfor å være forbindelsesløs/connectionless
- DNS tar i bruk UDP
- UDP vs TCP
    - Bedre applikasjonsnivåkontroll over hva slags data som sendes og når. UDP pakker og sender dataen med en gang, mens TCP har noen kontrollmekanismer for å se om lenkene mellom hostene er svært trafikkerte, i tillegg til at TCP prøver å sende pakkene om og om igjen helt til mottaker har bekreftet (uansett hvor lang tid det tar). Real-time applikasjoner krever ofte en minimum overføringsrate og tillater noe tap av data og egner seg dermed bedre med UDP. Feks videostream og telefoni
    - Ingen krav for etablering av forbindelse. TCP bruker en treveis handshaking før den sender pakken, mens UDP bare sender uten noen formelle krav. UDP har med andre ord ingen forsinkelse i etableringen av en forbindelse. Dette er kanskje viktigste grunnen til at DNS velger UDP over TCP.
    - Ingen tilkoblingstilstand. UDP holder i motsetning til TCP ingen informasjon om tilstandene til endesystemene. Dette er grunnen til at TCP tilbyr pålitelig overføring av data, men ikke UDP. Fordelen er at hver klient krever mindre tracking og dermed kan en UDP tjener normalt tjene flere klienter av gangen enn en TCP tjener.
    - Liten pakke. TCP segmenter har 20 bytes header overhead i hvert segment, UDP har 8 bytes.
- Det er mulig å ha pålitelig overføring av data med UDP, dersom pålitelighet bygges direkte inn i applikasjonen.
- UDP-headeren har kun 4 filter, hver bestående av 2 bytes.

- UDP sjekksum er en tjeneste for å detektere feil ved sending av et segment. MAO om den har blitt endret ved uhell mens den forflyttet seg fra kilden til destinasjonen.
    - Starter med å ta summen av alle 16-bits segmentene
    - Så toerkomplement av denne
    - Når den kommer fram, adderer den alle 16-bits segmentene inkludert den siste summen. Dersom dette blir 1111111111111111, er det good.
    - Siden det ikke kan garanteres at det gjennomføres feilsjekkeing lenger ned (link-layer protocol) må det gjennomføres i transportlaget. UDP gjør ingenting for å rette en eventuell feil.

3.4 - Prinsipper ved pålitelig overføring av data

- Ved en pålitelig overføring blir ingen bits endret eller tapt og alle kommer i den rekkefølgen de ble sendt
- **ARQ-protokollen** er en måte å sjekke om informasjonen har blitt mottatt riktig. For å oppnå dette, kreves tre ting:
    - Feildetektering: En måte å finne ut om det har skjedd en bit-feil
    - Tilbakemelding fra mottaker: Muligheten til å sende tilbakemelding om pakken har blitt mottatt riktig. Det sendes en ACK når alt er bra, og NAK når det har skjedde en feil.
    - Senderen må kunne sende på nytt: Når det har skjedd en feil, må det sendes på nytt.
- Senderen vil ikke sende en ny pakke før den vet at mottakeren har mottatt pakken korrekt. Derfor kalles slike protokoller **stopp-og-vent** protokoller.
- Problemet er at vi ikke tar hensyn til om ACK eller NAK pakken blir corrupted (hvis denne endres bare 1 bit innebærer det fatale konsekvenser). Muligheter for å fikse dette:
    - Legge til nok checksummer til at man også kan endre bitfeilen
    - Sende datapakken på nytt når den mottar en korrupt ACK/NAK
    - Legge til et sekvensnummer, slik at mottakeren bare trenger å sjekke dette nummeret for å bestemme om det er en retransmisjon eller ikke.
- Dersom vi tar hensyn til at kanalen kan miste pakke i tillegg til å korruptere bits blir det mer komplisert. Måten det løses på er at senderen velger et tidspunkt hvor den synes det virker sannsynlig at pakken blir tapt og resender den dersom denne tiden passerer uten svar. For å få til dette trengs en countdown timer. Dette kalles rdt3.0-protokollen (alternating-bit protocol).
- Rdt3.0 er en velfungerende prototype for pålitelig overføring av data, men den er ikke spesielt rask (hovedproblemet er at den er en stopp-og-vent protokoll)
- For å løse dette brukes et pipelined design der senderen kan sende flere pakker før den mottar ACK. Her økes intervallet for sekvensområdet og avsender og mottaker bruker en større buffer. To protokoller som implementerer pipelining er:

- Go-Back-N
  - Kan maks ha N ikke-acknowledge pakker i pipelinen
  - Kan fortsatt sende pakker selv om den venter på ACKs, men bare N slike pakker
  - Kalles sliding window protocol
  - Når rdt_send() kommer ovenfra (applikasjonslageret) må GBN sjekke at ikke vinduet (N) er fullt. Hvis det er fullt varsler den lageret over som da må prøve igjen senere (evt bruker et flagg som kalles når vinduet er ledig igjen). Er det ikke fullt, lages og sendes pakken og vinduet oppdateres.
  - Kumulativ ACK. En Ack for pakken med sekvensnummer n indikerer at alle pakker med sekvensnummer opp til og inkludert n har blitt korrekt mottatt av mottakeren.
  - Timeout. Dersom en timeout inntreffer må senderen resende alle pakkene som tidligere har blitt sendt, men ikke fått ACK.
  - Dersom den mottar en pakke med et sekvensnummer høyere enn denne variabelen, sletter den denne. Da vet den at det har skjedd en feil med den forventede og begge kommer dermed til å bli sendt på nytt, så er ikke vits i å lagre i buffer siden den blir sendt på nytt uansett.
- Selektive repeat (SR)
  - Hovedproblemet med GBN er at en enkel pakkefeil kan føre til at mange pakker må sendes på nytt. Denne unødvendigheten prøver SR å unngå så godt som mulig.
  - En begrensning på N brukes fortsatt
  - Vil derimot ACK korrekte mottatte pakker selv om disse ikke kommer i rekkefølge.
  - Avsenderen sender bare pakkene som den ikke har fått en ACK tilbake for på nytt. Pakkene som kommer i feil rekkefølge blir bufret hos mottakeren, frem til den har alle pakker den trenger for å sette de i riktig rekkefølge, før de blir sendt videre i mottakeren sitt system.

3.5 - Transportprotokoll med forbindelse: TCP

- Transmission control protocol
- En TCP forbindelse tilbyr full-dupleks tjeneste. Det vil si at data kan flyte mellom prosess A og B samtidig som data kan flyte mellom prosess B og A.
- Maksimal mengde data som kan plasseres i et segment begrenses av den maksimale segmentstørrelsen (MSS).
  - Inkluderer ikke headerne
  - Settes som regel ved å først finne lengden på den største linklags-rammen som skal sendes (MTU), for så å plusse på

headerlengden (som regel 40 bytes). Gjøres slik at både transport og nettverksdatagram passer inn i linklayer-framen

- I tillegg til feltene som UDP inneholder, har også TCP
    - 32-bit sekvensnummer og 32-bit ACK nr
        - Brukes for pålitelighet
        - ACK-nummeret er sekvensnummeret til den neste byten den forventer å motta
    - 16-bit mottaker-vindu felt som brukes til flytkontroll
    - 4-bit felt til headerlengden
    - Options-felt av variabel lengde som brukes når sender og mottaker diskuterer MSS og en timestamp.
    - Flag-felt av 6 bits med blant annet ACK, RST, SYN og FIN brukes ifm forbindelsen til setup og teardown. Dersom PSH er på skal mottakeren sende data til de øvre laget umiddelbart. URG brukes for å markere om det er ugrent data (markert av sender).
- TCP bruker i likhet med rdt-protokollene en timeout-mekanisme for å håndtere tapte segmenter.
    - Den må hvertfall være lenger enn RTT
    - RTT måler tiden fra et segment sendes til det godkjennes (ACK). Det måles ofte såkalte sampleRTTs, men en enkel måling er ikke representativ. Derfor har man en estimatedRTT.
        - Estimated RTT=(1- α)*EstimatedRTT+ α*SampleRTT
    - Ofte er α = 0,125. Dermed blir de nye målingene ganske mye vektet (EWMA - Exponential Weigthed Moving Average). Dette er naturlig da denne reflekterer bedre den nåværende opphopningen i nettverket.
- Man kan også ha en DevRTT (deviated) som viser hvor mye SampleRTT fraviker fra EstimatedRTT. Med mye fluktuasjoner vil DevRTT være stor.
    - DevRTT=(1- β)* DevRTT+ β*abs(SampleRTT-EstimatedRTT)
    - Anbefalt verdi for β = 0,25
- Hva skal TCPs timeout settes til? Viktig at den er større enn EstimatedRTT, men ikke for mye.
    - TimeoutInterval=EstimatedRTT+4*DevRTT
    - Reflekterer at timeoutintervallet skal være minst like stort som EstimatedRTT og større jo mer avvik der er mellom målingene (samtidig som intervallet skal være så lavt som mulig for å begrense forsinkelse).
- TCP ligner på GBN ettersom man må kunne holde info om det minste sekvensnummeret til en overført, men ikke ACK segmentet og sekvensnummeret til den neste byten som skal sendes.
- Den ligner også på SR dersom man benytter selektiv ACK (dvs tillater ACK på out-of-order segments) og dersom man dropper å sende på nytt de

segmentene som allerede har blitt selektiv ACK tidligere. TCP er en hybrid mellom GBN og SR.

- TCP tilbyr en flytkontroll-tjeneste for å eliminere muligheten for overflyt i mottakerens buffer. TCP lar senderen ha en variabel kalt mottakervindu. Den brukes til å gi senderen en ide av hvor mye ledig plass i bufferen mottakeren har. Siden TCP er full-dupleks (frem og tilbake) så har hver side av koblingen et distinkt mottakervindu. Vi må ha:

$$LastByteRcvd - LastByteRead \leq RcvBuffer$$

Gitt så

$$rwnd = RcvBuffer - [LastByteRcvd - LastByteRead]$$

Da må senderen passe på at

  - $$LastByteSent - LastByteACKed \leq rwnd$$

- TCP spesifikasjoner krever at host A fortsetter å sende segmenter med et databyte når rwnd = 0. Dette er fordi host B må varsle host A etterhvert som den leser (slik at ikke host A tror rwnd = 0 hele tiden dersom ikke host B har noe å sende) slik at den må sende tilbake ACK og dermed oppdatere host A på verdien av rwnd.
- UDP tilbyr ikke flytkontroll
- Etablering av TCP-kobling (3-veis handshaking)
  - 1. Klienten sender et spesielt TCP segment (ingen applikasjonslag-data). SYN = 1 og derfor kalles segmentet for SYN-segment. I tillegg sender klienten med et tilfeldig generert segmentnummer av sikkerhetsgrunner, client_isn. Segmentet innkapsles med et IP-datagram før det sendes til tjener.
  - 2. Tjeneren henter ut TCP SYN segmentet fra datagrammet, allokerer TCP buffere og variabler til forbindelsen og sender et forbindelsessegment til TCP klienten (her er faren for DDoS-angrep). SYN er også her lik 1. ACK-feltet settes til client_isn + 1, og tjeneren velger et eget sekvensnummer, server_isn. Dette segmentet kalles SYNACK-segment.
  - 3. Klienten allokerer buffere og variabler til forbindelsen. ACK-felt = server_isn + 1. Syn = 0. Her kan man for første gang sende klient-tjener data (fra applikasjonsprosessen) i segmentet.
- Når forbindelsen slutter, sender begge et segment med FIN=1.

3.6 - Tilnærminger til metningskontroll

- Kan skille mellom ulike tiltak ved å se på hvorvidt nettverkslaget tilbyr assistanse til transportlaget:

    - *Ende-til-ende kontroll.* Nettverkslaget tilbyr ingen eksplisitt assistanse til transportlaget. TCP-segment-tap (som indikeres ved en timeout eller en trippel duplikat ACK) tas som en indikasjon på opphopning, og TCP reduserer vindusstørrelsen sin tilsvarende.
    - *Nettverk-assistert kontroll.* Nettverkslag-komponenter (rutere) tilbyr eksplisitt tilbakemelding til sender hva angår opphopningstilstanden til nettverket, for eksempel en bit. Tilbakemeldingen kan sendes fra en nettverksruter til senderen gjennom en choke packet. En annen form for varsling kan skje dersom en ruter oppdaterer et felt i pakken fra sender til mottaker. Mottaker varsler så senderen (gjennom ACK eller en ny pakke). Denne tilbakemeldingen tar da minst en full RTT.

3.7-TCP metningskontroll

- TCP må tilby ende-til-ende kontroll siden IP-laget ikke tilbyr noen eksplisitt tilbakemelding til endesystemene hva angår nettverksopphopning. TCP sin tilnærming er å la senderen begrense raten den sender trafikk til forbindelsen som en funksjon av den oppfattede opphopningen til nettverket.
- TCP-mekanismen for kontroll av opphopning holder styr på en variabel kalt congestion window, cwnd. Mengden ikke-ACK data kan ikke overstige den minste av denne og rwnd (mottakervindu).
    - LastByteSent - LastByteACKed ≤ min{rvwn, cwnd}
    - Dersom vi antar rwnd stor nok, at senderen hele tiden har data å sende og at tap og overføringsforsinkelser er neglisjerbare, så kan senderen overføre cwnd bytes data ved starten av hver RTT. Overføringsraten er da cwnd/RTT og senderen kan justere raten ved å justere verdien av cwnd.
- Når det er opphopning på nettverket så vil en av ruterne på veien mellom TCP-sender og TCP-mottaker overflyte, som gjør at et datagram droppet. Dette innebærer tap av en pakke som senderen merket enten gjennom en timeout eller tre duplikate ACK. Dette oppfatter senderen som et tegn på opphopning.
- TCP bruker ACK (dersom ACK kommer hyppig med høye rater øker den cwnd) til å trigge økoningen av cwnd. Den sies derfor å være self-clocking.
    - Et tapt segment indikerer at TCP-senderen sin rate skal reduseres.
    - Et segment som blir ACKed indikerer at nettverket klarer å sende senderens segmenter til mottakeren, og at senderens rate kan økes.
    - Bandwidth probing. Øk overføringsraten i samsvar med ankommende ACKs helt til tap av segment inntreffer. Da reduseres hastigheten litt igjen, før den økes videre for å se om opphopningsraten har endret seg
- **Slow start** - raten begynner typisk på MSS/RTT (som ofte er lavt - derav slow start). Deretter økes raten med 1 MSS for hvert segment senderen mottar ACK for (dermed dobles senderens rate for hver RTT). Dersom tap av et segment

inntreffer setter TCP cwnd = 1*MSS og begynner slow start prosessen på nytt. Den setter ssthresh = cwnd/2 (halvparten av det cwnd var når man oppfattet tapet av pakken). Når cwnd = ssthresh er slow start over, og TCP går over i congestion avoidance mode, der den øker cwnd mer forsiktig.

- **Congestion Avoidance -** Når denne tilstanden inntas, er cwnd ca halvparten av hva den var da congestion sist ble oppdaget. Det betyr at congestion kan være rett rundt hjørnet, og dermed øker TCP cwnd sakte i stedet. Cwnd økes med én MSS hver RTT. Dette er en lineær økning. Ved timeout settes cwnd til 1 MSS, og ssthresh er cwnd/2 ved tidspunktet for tapet. Når det ikke er timeout, men en trippel duplikat-ACK, gjøres det annerledes. Cwnd halveres (og legger til 3 MSS for å gjøre opp før trippel ACK) og setter ssthresh til halvparten av cwnd ved tidspunktet de tre ACKene ble mottat. Går da inn i fast recovery state.
- **Fast recovery** - Dersom fast recovery er implementert (som er vanlig i dag - og anbefalt) vil man kunne gå fra congestion avoidance til fast recovery ved tre duplikate ACK. Da settes ssthresh = cwnd/2 og cwns = ssthresh + 3*MSS. Deretter økes cwnd med 1*MSS for hvert duplikate ACK. Dersom timeout inntreffer går man tilbake til slow start, mens dersom en ny ACK (for et nytt segment) inntreffer går man tilbake til congestion avoidance-tilstanden (med cwnd = ssthresh).
- Kontroll av opphopning sies å være fair dersom gjennomsnittlig overføringsrate til hver forbindelse er omtrent R/K der K er antall TCP forbindelser. Dette innebærer at hver forbindelse fordeler båndbredden likt mellom seg.
- Der TCP opphopningskontroll vil redusere overføringsraten ved tap av pakker, vil UDP ikke gjøre dette, og dermed kan UDP kilder ta trafikk fra TCP. UDP kontroll av opphopning er ikke fair. TCP-baserte applikasjoner som bruker flere, parallelle forbindelser er ikke fair, da de får en større del av båndbredden i en opphopet link.
- Nettverks-assistert congestion control er kjent som Explicit Congestion Notification (ECN). I nettverkslaget er to bits i Type of Service i IP datagramheaderen brukt for ECN. En ruter kan bruke dette til å indikerer at den opplever congestion. Kan også brukes av den sendende host for å informere rutere om at sender og mottaker er ECN-kapable, altså kapable til å gjøre handlinger som respons på ECN-indikert nettverkscongestion.
    - Signaliseres ved at mottakeren får en ECN indikasjon fra et datagram, og den informerer senderen om dette ved å sette ECE (Explicit Congestion Notification Echo) bitet i en mottaker-til-sender ACK. Svarer på dette ved sende medling om å halvere cwnd.

Chapter 4 - Network layer part 1

4.1 - Oversikt over nettverkslaget

- Hovedrollen til nettverkslaget er å flytte pakker fra en sendende vert til en mottagende vert. På veien bruker den følgende:
    - Videresending (forwarding): Når en pakke ankommer en ruter sin input-lenke må ruteren videresende denne pakken til den passende output-lenken. Ofte implementert i hardware.
    - Ruting (Routing): Bestemmer/planlegger veien pakkene skal ta fra sender til mottaker, ofte implementert i software
- Enhver ruter har en videresendingstabell (forwarding table). Denne blir brukt når ruteren videresender pakken ved å se på verdien i pakkens header. Routing algorithms bestemmer verdiene som skal settes i tabellen. Algoritmene kan være sentraliserte (en algoritme kjører sentralt og laster ned tabellene til hver av ruterne) eller desentralisert (med en distribuert routing algorithm som kjører på hver ruter).
- Nettverkstjenestemodellen definerer karakteristikkene til ende-til-ende transport av pakker mellom senderens og mottakerens endesystemer. Mulige tjenester nettverkslaget kan tilby:
    - Garantert leveranse
    - Garantert leveranse med en øvre grense på forsinkelsen
    - In-order pakkelevereanser - pakkene ankommer destinasjonen i rekkefølgen de ble sendt
    - Garantert minimum båndbredde - Hver pakke har et garantert minimum på båndbredde som gjør at dersom senderens host overfører bits på en rate som er lavere enn den spesifiserte raten vil ingen pakker bli tapt og alle komme frem innen en forhåndskjent host-til-host forsinkelse.
    - Sikkerhetstjenester - datagrammene kan krypteres ved senderen og dekrypteres hos mottakeren
- Internettets nettverkslag tilbyr en enkelt tjeneste kjent som best-effort service
    - Timing mellom pakker ikke garantert å holdes konstant, in-order pakkeleveranser er ikke garantert, heller ikke at pakkene blir levert

4.2 - Hva er på innsiden av en ruter?

- Består av fire hovedkomponenter:
    - Input-port
    - Svitsj (kobler input-portene til output-portene)
    - Output-port
    - Routing processor
- Forwarding table er enten beregnet og oppdatert av routing prosessoren eller mottatt fra en utenforstående SDN-kontrollør
    - To typer forwarding

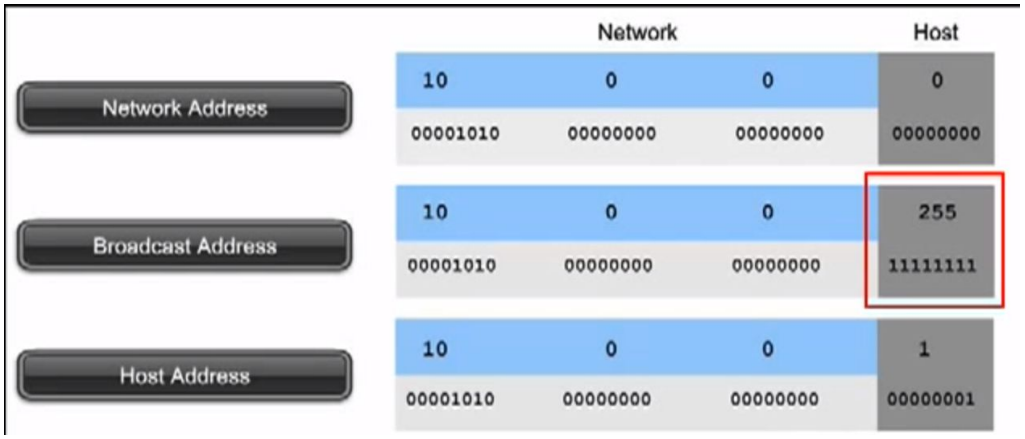
        - Destination-based forwarding: Sender pakken på output-port ettersom hvor pakken skal
        - Generalized forwarding: Sender pakken på output-port basert på type pakke
- Når det er flere matcher, bruker rutere longest prefix matching
- Når den utgående porten til pakken er blitt funnet (vha tabellen) kan pakken videresendes til svitsjingfabrikken. Pakken kan dog bli midlertidig blokkert fra denne fabrikken, dersom pakke fra andre inngående lenker bruker den. En blokkert pakke må da settes i kø før de kan gå gjennom svitsjingfabrikken.
- Svitsjefabrikken er midt i hjertet av ruteren. Det er gjennom denne at pakkene faktisk svitsjer, det vil si videresendes, fra en inngående port til en utgående port. Svitsjing kan oppnås på flere måter:
    - Svitsjing via minne. Svitsjing mellom inngående og utgående porter gjøres gjennom direkte kontroll av CPU (routing prosessor). Inngående og utgående porter fungerer som tradisjonelle I/O enheter i et tradisjonelt operativsystem. Lookup av adresser og lagring av pakker i riktige minnelokasjoner gjøres av prosessorer på de inngående line cards. Rutere som svitsjer via minne ligner veldig på multiprosessorer som deler minne. Da er prosessorene på et linjekort og svitsjer pakker i minnet til den riktige utgående porten.
    - Svitsjing via bus. Alle output-portene mottar pakken, men bare den porten som matcher nummeret beholder den. Dersom flere pakker ankommer ruteren samtidig, på forskjellige porter, må alle bortsett fra én vente siden bare én pakke kan krysse bussen samtidig.
    - Svitsjing via et sammenkoblingsnettverk. En **crossbar switch** er et sammenkoblingsnettverk som består av 2n busser som kobler sammen n inngående porter og n utgående porter. Denne løsingen er nyttig for å unngå at båndbredden begrenses av busshastigheten til en enkelt delt buss som over. Kan videresende flere pakker i parallell. Den er også non-blocking, altså vil ikke en pakke på vei mot en output port bli blokkert med mindre det er en annen pakke på vei til samme. Derimot, hvis pakkene kommer fra ulik input-port, så må man vente ved inputporten, siden det bare er én pakke av gangen over bussen.
- Output-port prosessing tar pakkene som har blitt lagret i den utgående porten sitt minne og overfører dem til den utgående lenken.
- Pakke-køer kan forekomme ved den inngående porten og utgående porten. Etterhvert som disse køene blir store vil ruterens bufferrom bli utslitt/fullt og tap av pakker vil forekomme (pakker kan som nevnt enten gå tapt over lenker eller droppet av en ruter).
- **Head of the line blocking** kan oppstå når man har en FCFS (first-come-first-served) strategi. Dette skjer dersom pakke C i køen (som har sin utgående port tilgjengelig) blokkeres av at pakke B (først i køen) må vente

på at en pakke A overføres til den utgående porten før pakke B kan overføres til den samme porten. Pakke C opplever da head of line blocking.

- Forskjellige implementasjoner for å unngå pakketap
    - FIFO - First in first out
        - Én kø som det velges pakke fra, fremst i køen får kjøre først
    - Prioritetskø
        - Flere køer, deler inn pakkene etter prioritet, og velger den ene køen som har prioritet hele tiden. Valg av pakker i den køen gjøres om FIFO.
    - Round Robin og WFQ (Weighted fair queuing - pakker sorteres i klasser)
        - Round Robin - Går sekvensielt gjennom køene og velger hvem som skal sendes etter rekkefølge
        - WFG - Som Round Robin, men klassene/køene er vektet

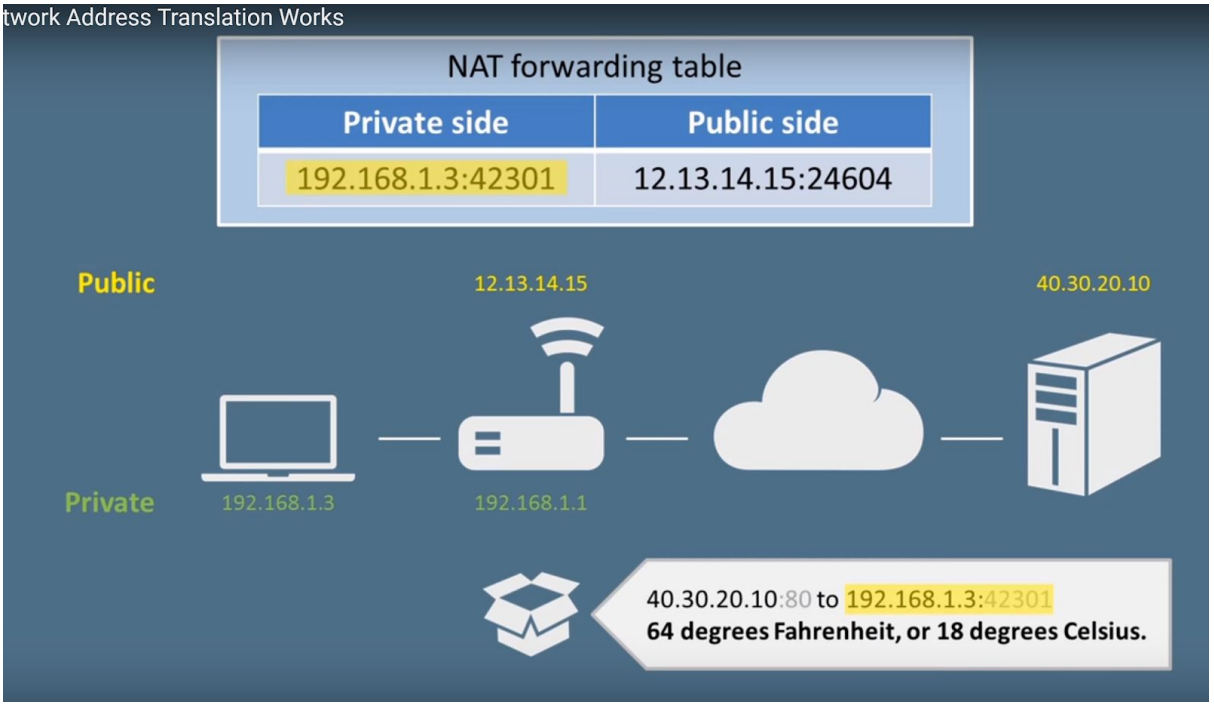
4.3 - IP

- Nøkkelfeltene i IPv4 datagrammet er som følger
    - Version number - 4 bits som spesifiserer IP protokoll versjonene til datagrammet. Ved hjelp av denne kan ruteren bestemme hvordan resten av IP datagrammet skal tolkes.
    - Header length - 4 bits som bestemmer hvor i IP datagrammet dataen faktisk begynner
    - Type of Service - TOS bitsene tillater forskjellige typer IP datagrammer å bli skilt fra hverandre, feks real-time datagram som telefoni, eller non-real-time som FTP. To av disse bitsene er også brukt for Explicit Congestion Notification
    - Datagram length - den totale lengden til IP datagrammet (header plus data) målt i bytes. 16 bits felt, altså er maks størrelsen 65 535 bytes. de er sjelden større enn 1500 bytes.
    - Identifier, flags, fragmentation offset - hjelper endesystemene med å samle sammen eventuelle fragmenter av pakker. IPv6 tillater ikke fragmentering
    - Time-to-live - sikrer at datagrammene ikke sirkulerer i evig tid. Feltet reduseres med 1 for hver gang datagrammet prosesseres av en ruter og datagrammet kastes om feltet når 0.
    - Protocol - Verdien av dette feltet indikerer hva slags transportprotokoll datadelen av datagrammet skal sendes til (6 er TCP, 17 UDP).
    - Header checksum - hjelper ruteren å detektere bitfeil. Pakker med feil kastes ofte. Checksum må regnes på nytt på hver ruter da TTL feltet kan endre seg.
    - Source and destination IP address
    - Options

    - Data (payload) - oftest transportlagsegmentet som skal sendes til destinasjonen. Kan også frakte annen type data som ICMP meldinger
- IP-datagram har totalt 20 bytes header. Dersom den også har et TCP segment, har den 40 bytes header
- De tre første bytesene består av "nettverks"delen, og den siste byten er host-delen.
    - 

    - Et lite nettverk med flere devices har adressen 10.0.0.0, mens alle devices innenfor der har egne adresser, men lik nettverks-del. Kalles nettverkaddresse
    - Broadcast-adresse er at man sender ut noe til alle på den IP-adresse, slutter med .255
    - Alle devices innenfor nettverket har sin egen adresse fra .1 til .254
- Hoster kan kommuniserer på 3 ulike måter
    - Unicast
        - Sender ut melding til en bestemt adresse
    - Directed Broadcast
        - Sender ut melding til alle host i et spesifikt nettverket
        - Når de mottas blir de behandlet som en Unicast
    - Limited Broadcast
        - Kommunikasjon med alle hosts på local network
    - Multicast
        - Sender ut melding til flere bestemte adresser, i stedet for å sende mange Unicast
- Dersom lengden på pakken er større enn MTU (maksimal mengde data link-layer frame kan bære) vil man ikke få plass til pakken i payload feltet til link-layer frame. Begrensningen er ikke noe problem i seg selv, problemet ligger i at hver link gjennom ruten kan bruke forskjellige linklags protokoller, med ulik MTU
    - Måten man løser dette på er å fragmentere payload i pakken (dele den opp) i to eller flere mindre IP datagram (fragmenter), innkapsler disse i separate link-layer frames og sender de så over den utgående lenken nå som de er små nok.

    - Fragmentene må samles sammen igjen før de når transportlaget ved destinasjonen. Siden ruterne allerede har nok å gjøre, bestemte designere av IPv4 at jobben med å samle sammen fragmentene skulle gjøres i endesystemene istedet for hos nettverksruterne. Fragmenter kan gå tapt, og flag-bitet settes til 0 for siste fragmentet for å vise endesystemet ved destinasjonen at den er sist i en del av en større pakke.
    - Offset benyttes for å finne hvor fragmentet passer inn i det originale IP datagrammet slik at de kan samles sammen riktig og endesystemet kan oppdage om noen andre fragmenter (ikke det siste) har gått tapt.
    - I IPv4 headeren er det 13-bit fragmenteringsoffsett, 16-bit Identifier og Flagg som brukes ifm fragmentering.
- Et endesystem har typisk kun én lenke til nettverket. Datagrammer sendes over denne. Grensen mellom endesystemet og den fysiske lenken kalles et grensesnitt (interface).
- En IP-adresse er egentlig assosiert med et grensesnitt, ikke en ruter
- IPv4 er 32-bits lang, og skiller hver av de fire bytene med punktum.
- /N er subnett-masken og tilsier at N av de 32 bitene definerer subnett-adressen
- Internett sin adresseringsstrategi er kjent som Classless Interdomain Routing (CIDR)
- De x mest signifikante bitsene til en adresse på formen a.b.c.d/x danner nettverksdelen av IP adressen og refereres ofte til som adressens prefix. Organisasjoner tilegnes typisk en blokk med adresser med en felles prefix på x bits. Kun de første x bitsene til IP-adressen vurderes av en ruter utenfor organisasjonens nettverk (den må sendes hit før de siste bitsene trenger å vurderes). De siste (32 - x) bitsene er de som må vurderes av ruterne på innsiden av organisasjonen. Når et endesystem sender et datagram med IP adresse 255.255.255.255 sendes det til alle endesystemene i subnettverket.
- DHCP (Dynamic Host Configuration Protocol - en systemadministrator) tilegner IP-adresser til endesystemer automatisk. Denne kan konfigureres til å være den samme for et endesystem for hver gang de kobler seg på nettverket, eller så kan det være midlertidig og endre seg hver tilkobling. For en ny klient (endesystem) som ønsker å oppnå nettverkskonfigurasjon (inkluderte en IP-adresse) er DHCP en 4-stegs prosess.
    - DHCP server-discovery - for å finne en DHCP tjener å kommunisere med sender klienten en DHCP discover message (på UDP) til port 67. Denne har destinasjons IP adresse 255.255.255.255. og en kilde IP adresse 0.0.0.0. Linklaget sender da denne til alle nodene på subnettverket.
    - DHCP server offer(s) - En tjener som ser meldingen svarer til klienten med en DHCP offer message som også sendes til alle nodene ved bruk

av 255.255.255.255. Tilbudsmeldingen må inneholde en foreslått IP adresse som klienten kan bruke med en IP adresse lease time hvor adressen er gyldig og kan brukes (ofte timer eller dager). DHCP tilbyr en mekanisme som gjør at leien kan fornyes.
    - DHCP request - Klienten vil velge mellom en eller flere tilbud fra tjenere med en DHCP request message.
    - DHCP ACK - Den valgte tjeneren responderer med en ACK melding.
- Fordi DHCP evner å automatisere de nettverksrelaterte aspektene ved å koble et endesystem til et nettverk refereres den ofte til som plug-and-play protocol.
- Dersom det er flere DHCP servere, vil alle tilby en IP-adresse, og klienten kan velge en av disse. Dersom det er flere klienter som spør om en IP-adresse samtidig, holder DHCP oversikt over dette med et eget transaction number for requestene.
- En IP-adresse er tilgjengelig i et tidsrom. Når tidsrommet er ute, vil den samme IP-adressen være tilgjengelig for bruk igjen. Man gjenbruker IP-adresser.
- NAT (Network Address Translation) er en måte å allokere IP-adresser på. Den løser problemet man får dersom alle de allokerte host IP-adressene i en organisasjon er tatt og man trenger flere, men ingen er tilgjengelige.
- 

    - Vi har en privat IP-adresse og en offentlig IP-adresse. Ruteren skjuler vår IP-adresse fra omverdenen. Ruteren vår endrer IP-adressen før den sender noe over internett. Lagrer private IP-adressen i en forwarding table. Når den får en pakke tilbake igjen fra internett, så er pakken sendt til den offentlige IP-adressen. Ruteren slår da opp i forwarding table og finner klientens private IP- og sender videre.

- Et problem med NAT er at det forstyrrer P2P applikasjoner. Hvis en TCP er bak en NAT kan den ikke oppføre seg som en tjener og akseptere TCP forbindelser.
- De viktigste endringene introdusert i IPv6 finner vi i datagram formatet, det har alltid en fast størrelse, slik at prosesseringen av IPv6 er raskere enn IPv4 selvom den er større.
    - Utvidet adresseringskapasitet - størrelsen på IP-adressene er økt fra 32 til 128 bits. IPv6 har introdusert en anycast address som tillater et datagram å bli levert til hvem som helst blant en gruppe hosts
    - A streamlined 40 bit header - flere felter har blitt droppet eller blitt frivillige. Denne tillater raskere prosessering av IP datagrammet
    - Flow labeling and priority - man kan merke pakker som hører i samme kategori og gi prioriteter
    - Version
    - Flow label
    - Traffic class - likt som TOS i IPv4 (kan gi prioritet på pakker)
    - Payload length - 16-bit verdi som gir antall bytes i datagrammet etter headeren
    - Next header - protokollen som datagrammet skal leveres til (UDP/TCP++)
    - Hop limit - Som TTL i IPv4
    - Source and destination adresses - 128 bits hver
    - Data
- Disse feltene er borte
    - Fragmentation/Reassembly - kan ikke lenger gjøres i ruterne, kun i endesystemene. Om en for stor pakke mottas hos en ruter sendes en ICMP feilmelding om at pakken er for stor og så må endesystemene fragmentere en og sende på nytt
    - Header checksum - kostbar operasjon som også blir gjort i transportlaget og linklaget. Droppet for hastighetens skyld
    - Options - ikke droppet, men kan nå være en av de mulige next header som pekes på av IPv6
- Tunneling
    - IPv6 systemer kan bruke IPv4, men ikke omvendt. Det tar lang tid å få alle devices over på IPv6.
    - Tunneling er en metode for IPv4-til-IPv6 overgang
    - Scenario: To IPv6 noder vil kommunisere, men er hver koblet til en IPv4 ruter mellom seg. Mengden IPv4-rutere mellom IPv6 noder kalles en tunnel.
    - IPv6 senderen tar et helt IPv6 datagram og legger det i payload-feltet til et IPv4 datagram. IPv4 ruterne behandler denne som hvilken som helst datagram. Når den mottas hos IPv6 node på andre siden av tunnelen,

ser den at payloaden inneholder IPv6 datagram (fordi protokollnr-feltet i IPv4 datagrammet er satt til 41). IPv6 datagrammet hentes da ut og sendes videre som et vanlig IPv6 datagram.

Chapter 5 - Network layer part 2

- To måter videresendingstabeller lages og installeres
    - Per-ruter kontroll
        - Routing algorithms kjører i hver ruter
        - Både videresending og routing er funksjoner i hver ruter
    - Logisk sentralisert kontroll
        - En logisk sentralisert kontrollør lager tabellene
- Routing algorithms
    - Målet er å finne en bra path fra sender til mottaker
    - Least-cost path - veien med minst kostnad
    - Shortest path - vei med færrest linker
    - Centralized routing algorithm
        - Finner least-cost path ved å ta alle nodene og kantene inn som input, man må vite denne informasjonen før selve utregningen
        - Algoritmer med global status informasjon kalles link-state (LS) algoritmer
    - Decentralized routing algorithms
        - Finner least-cost path på en iterativ og distribuert måte gjort av ruterne.
        - Ingen node har komplett oversikt over kostnaden til linkene i nettverket, men går gjennom naboene for å finne korteste vei (BFS?)
        - Distance vector (DV) typisk algoritme, siden hver node har en vektor som estimerer kostnaden til alle andre noder.
    - Static routing algorithms - ruter endres sakte over tid, ofte som resultat av menneskelig handling (endre link-kostnaden manuelt)
    - Dynamic routing algorithms - Endrer paths ettersom nettverstrafikken lader eller topologisk endrer seg. Kan kjøres periodisk eller etter endring. Bedre egnet for endringer, men sårbare for routing loops og route oscillation
    - Load-sensitive algorithm - link kostnader endres dynamisk for å reflektere current metningsnivå i underliggende link. Ble brukt i tidlige ARPAnet, men oppdaget problemer
    - Load-insensitive - brukes i dag, en links konstnad reflekterer nødvendigvis ikke metningsnivået
- ICMP - Internet Control Message Protocol
    - ICMP brukes av verter og rutere for å sende nettverklagsinfo til hverandre. Brukes mest til feilrapportering.

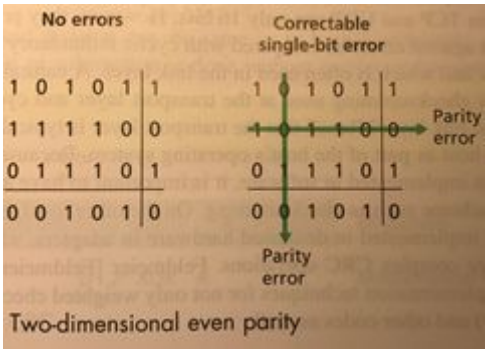
    - ICMP meldinger fraktes gjennom IP datagram, i payload
    - De har et type og et kodefelt, og inneholder headeren og de første 8 bitsene av IP datagrammet som genererte ICMP meldingen

Chapter 6 - The link layer and LANs

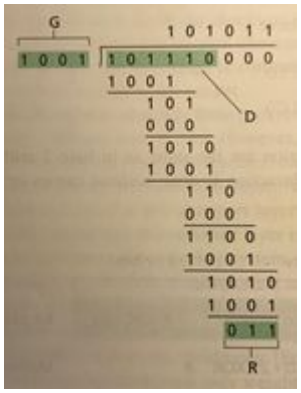
6.1 - Introduksjon

- Her vil rutere og endesystemer kalles noder, da det ikke er interessant om en node er den ene eller den andre
- Kommunikasjonskanalene som kobler sammen nabonoder vil kalles koblinger (lenker/links)
- Over en gitt link sendes innkapslede datagrammer i en frame/ramme
- Tjenester som linkslag-protokollen tilbyr inkluderer
    - Framing. Nesten alle protokollene innkapsler datagrammene i en koblingslagsramme før den overføres over koblingen. Rammen består av et datafelt hvor datagrammet innsettes og et nummer fra header feltet. Strukturen til rammene bestemmes av protokollen
    - Link access. Et medium aksesskontroll (MAC - Media Access Control) protokoll spesifiserer reglene for hvordan en ramme overføres til koblingen
    - Reliable delivery. Garanterer å flytte datagrammene over koblingen uten feil. Som med TCP oppnås pålitelig overføring ofte ved ACK og retransmission. Brukes ofte ved koblinger som er utsatt for høye error rates slik som en trådløs kobling med mål om å rette feilene lokalt i stedet for å tvinge ende-til-ende retransmission av transport eller applikasjonsprotokollen. For low bit-error koblinger (coax, fiber) er denne tjenesten ofte unødvendig.
    - Error detection. De fleste koblingsprotokollene har en mekanisme for å detektere bit-feil. Dette er nyttig da det ikke er noen grunn til å videresende datagrammer der det har skjedd en feil. Feildetektering i koblingslaget er ofte mer sofistikert enn i transport/nettverkslaget og implementeres i hardware.
    - Error correction. Her detekteres ikke bare feilen, men det bestemmes nøyaktig hvor i framen feilen har skjedd og så rettes den opp.
- Linklaget er for det meste implementert i nettverksadapteren, også kjent som nettverksgrensesnittkortet. Mesteparten av funksjonaliteten er implementert i en chip (NIC).
- Senderen tar datagrammet som har blitt lagret i minnet av et høyere lag, innkapsler dette i en linklagsramme og sender denne på kommunikasjonslinken. Mottakeren mottar hele framen, og uthenter nettverkslagets datagram. Encapsulation og decapsulation.
- Delt inn i to subnet - LLC (Logical link control) og MAC (Media access control)

6.2 - Feildetektering og retting

- Dataen (inkluderer ofte sekvensnummer og andre felter - ikke bare meldingen) som sendes (D) og som skal beskyttes utstilles med feil-oppdaging og -korrigeringsbits (EDC)
- På mottakersiden mottas en sekvens med bits der i blant D' og EDC'. Dersom D' og EDC' ikke er lik D og EDC har det skjedd en feil ved sendingen. Error-detection og correction-teknikker tillater mottakeren å finne ut av feil. Man ønsker dermed å ha teknikker som holder sannsynligheten for at feil ikke oppdages som lav. Vi ser nå på tre teknikker for å oppdage feil i den overførte dataen
- Paritetsbit
    - En av de enkleste formene for feildetektering
    - Velger om man skal ha et odde eller partall av enere
    - Dersom dette ikke stememr overens når den kommer frem, er det noe feil
    - Målinger har vist at bitfeil ofte skjer i burst (flere av gangen) og sannsynligheten for at feilen ikke oppdages ved denne teknikken er ca 50%
    - En todimensjonal paritetsskjekk med paritetsbit for hver rad og kolonne gjør det mulig å vite nøyaktig hvor feilen oppsto, og dermed rette den. Det gjør det også mulig å detektere feil selv om det skyldes to bits som er endret. Egenskapen med å kunne detektere og korrigerefeil kalles forward error correction (FEC). FEC teknikker er verdifulle da de reduserer antall pakker som må sendes på nytt. I tillegg muliggjør de korrigering med en gang (uten å vente en full RTT).
    - Detektere og korrigere 1 databit feil
    - Detektere, men ikke korrigere 2 databit feil
    - Detektere og korrigere 1 paritetsbit feil
    - 

- Sjekksum
    - Bytes blir lest for 16-bit heltall og summeres
    - Mottakeren sjekker ved å ta 1er komplementet (se 3.3)
    - Hvis svaret blir 16 1ere, så er det riktig, feil hvis vi har en 0er
    - Krever lite overhead i pakken, men svak i forhold til CRC.

    - Sjekksum er i både transportlag og nettverkslag, brukes fordi error-detection skjer i software, og det er viktig at error-detection er rask og enkel. Linklaget er mer hardware, slik at CRC kan benyttes
- CRC
    - Cyclic redundancy check
    - Også kalt polynomisk kode, mye brukt i dag
    - Sender og mottaker blir først enige om en generator (G)
    - Se side 477
    - 

6.3 - Multiple Access Links and protocols

- En punkt-til-punkt kobling består av en sender på en ende og en mottaker på den andre enden. PPP og HDLC er eksempler på protokoller denne typen kobling bruker.
- Den andre typen kobling er en broadcast kobling. Denne kan ha flere sendende og mottakende noder som alle er koblet til den samme delte broadcast channel (kringkastiningskanalen)
    - Broadcast brukes fordi når en node overfører en frame (innkaplset datagram), så vil kanalen broadcaste framen og hver av de andre nodene mottar en kopi
- Hvordan koordinere aksessen til flere sendende og mottakende noder til en delt broadkastlinje kalles multiple access problem
- Datanettverk har et sett med protokoller for å regulere overføringer i en broadcastlinje kalt multiple access protocols. Disse protokollene passer på at flere ting ikke blir sendt samtidig og dermed kolliderer og går tapt. Enhver slik protokoll kan klassifiseres i en av tre kategorier; channel partitioning protocols, random access protocols og taking-turns protocols.
- En slik protokoll bør sørge for
    - Når bare en node har data å sende, har denne gjennomstrømming på R bits/s
    - Når vi har M noder som har data å sende, har alle en gj.snitt gjennomstrømming på R/M
    - Den er desentralisert (ingen masternode)
    - Enkel å implementere

- Channel Partitioning Protocols
    - Bruker enten TDM (Time Division Multiplexing) eller FDM (Frequency Division Multiplexing). Multiplexing er å slå sammen og sende flere ting på en gang uten at de interfarer med hverandre, og kan skilles igjen etter sending.
    - Anta N noder og overføringsrate R. TDM deler tiden inn i tidsrammer og deler videre tidsrammene inn i N tidsrom (ikke forveksle tidsrammer med rammepakkene/frame) som fordeles mellom de N nodene. I en tidsramme får typisk alle nodene et tidsrom hvor de får overført en pakke. TDM umuliggjør kollisjoner og her helt fair: Hver node får en overføringsrate på R/N bps i hver tidsramme. TDM har likevel to ankepunkter.
        - En node blir begrenset til R/N bps selv når det er den eneste noden som ønsker å sende noe. For det andre må en node alltid vente på sin tur i overføringssekvensen.
    - FDM deler R bps inn i frekvenser (hver med båndbredde R/N) og tilegner hver frekvens til en av de N nodene. FDM lager dermed N mindre kanaler av R/N bps ut av en større kanal på R bps. De sendes samtidig, men har hvert sitt frekvensrom. FDM deler de samme fordelene og ulempene som TDM.
    - En tredje slik protokoll er en CDMA (Code-Divison Multiple Access). Den tilegner en kode til hver node. CDMA nettverk har den herlige egenskapen at forskjellige noder kan overføre pakker samtidig og mottakeren kan motta disse korrekt (gitt at mottakeren kjenner til sendernes koder).
- Random access protocols
    - En sendende node har alltid gjennomstrømming på R bps
    - Når det oppstår en kollisjon, vil alle nodene involvert i kollisjonen sende rammen på nytt etter et tilfeldig tidsintervall. Den er tilfeldig slik at ikke samme kollisjon oppstår igjen.
    - **Slotted ALOHA**
        - Antar alle frames har nøyaktig L bits, et tidsintervall er akkurat nok til å sende en ramme, nodene starter å overføre kun i begynnelsen av tidsintervallet, nodene er synkrone så de vet når tidsintervallet starter. Dersom to eller flere frames kolliderer, observerer alle nodene dette før tidsintervallet er over. La p være sannsynlighet. Slotted ALOHA fungerer dermed på følgende måte:
            - Når noden har en ny frame som skal sendes, venter den til begynnelsen av neste tidsintervall og overfører hele rammen.

            - Dersom det ikke er en kollisjon, er den ferdig og trenger ikke å vurdere å sende på nytt
            - Dersom det er en kollisjon, vil noden overføre pakken i hvert påfølgende tidsintervall (med sannsynlighet p) helt til pakken er overført uten kollisjoner
        - Nodene gjennomfører på en måte et mynktkast med sannsynlighet p for at de skal overføre igjen i neste periode og sannsynlighet 1-p for at de skal stå over og vente enda en tidsperiode før de kaster mynten igjen (random forsinkelse)
        - Gitt N noder er sannsynligheten for at nøyaktig én node overfører vellykket lik Np*(1-p)^(N-1). Dette er da effektiviteten til protokollen (sier hvor ofte vi får en overføring uten kollisjon). Vi må finne en p som maksimerer denne. Dersom vi antar mange noder N tar vi grenseverdien når N går mot uendelig. Maksimal effektivitet oppnås da til 1/e = 0,37. 37% av tidsperiodene er vellykket, 37% går tomme og 26% har kollisjoner.
        - Raskere enn (Pure) ALOHA
    - (Pure) ALOHA
        - Forskjellen er at man ved ALOHA umiddelbart sender pakken (dvs når datagram sendes ned fra nettverkslaget til den sendende noden) uten bruk av tidsrom.
        - Tilsvarende, dersom det oppstår kollisjon vil man med en gang sende pakken på nytt med en sannsynlighet p. Dersom denne ikke inntreffer venter man en frame transmission time (tiden det tar å overføre en pakke).
        - Sannsynlighet for en vellykket overføring ved ALOHA blir da (1-p)^(2*(N-1)).
        - Maksimal effektivitet er her 1/2e = 0,185. Dette er prisen man må betale for at ALOHA protokollen er helt desentralisert.
    - CSMA
        - Carrier sense multiple access
        - To viktige regler
            - Lytt før du overfører - En node lytter til kanalen før den begynner å overføre en pakke. Dersom en annen pakke blir overført, så venter den til denne er ferdig ved å trekke seg tilbake en gitt tidsperiode før den igjen forsøker å lytte. Er kanalen ledig vil den begynne overføringen. Dette kalles carrier sensing.
            - Dersom en annen begynner overføring mens du overfører, slutt å overføre - Dette kalles **collision detection.** Om dette skjer, vet noden at en kollisjon vil

inntreffe og den slutter derfor å overføre og bruker en eller annen protokoll for å bestemme når den skal overføre igjen. Dette er CSMA/CD.

- Taking turns protokoller
    - ALOHA og CSMA tilfredsstiller ikke kravet om at når det er M aktive noder, kan de sende med en gj.snitt hastighet R/M bps.
    - Polling protocol er en kjent taking turns protocol. Denne krever at en node blir tilkjent som masternode som styrer hvem som skal overføre etter tur. Polling protocol eliminerer muligheten for kollisjoner og er dermed mer effektiv. Men da denne er sentralisert (ved masternoden) får man en ekstra forsinkelse (polling delay - dersom en node som har turen ikke vil sende noe feks). Dersom masternode feiler, så er også hele kanalen ikke-operativ.
    - Token passing protocol bruker en token (en spesiell frame) som sendes mellom nodene i en gitt rekkefølge. Noden med token kan overføre. Denne metoden er desentralisert og veldig effektiv. Her også kan feil ved en node medføre at kanalen krasjer.
- Data-Over-Cable Service Interface Specification (DOCSIS) spesisfierer arkitekuren til kabeldata-nettverket og dens protokoll
    - Bruker FDM for å dele opp downstream og upstream segmenter
    - Cable Modem Termination System - CMTS
        - Cable access network knytter sammen flere residential cable modems til en CMTS
    - Downstreamen ikke noe problem, upstreamen er verre
    - CMTS gir mini-slots til upstream transmitting ved å sende en MAP melding på downstream for å spesifisere hvilket kabelmodem som kan transmittere under hvilket tidsintervall

6.4 - Svitsjede LANs

- Adapterne til verter og rutere har linklagsadresser (MAC-adresser). En vert eller ruter med flere nettverksgrensesnitt vil dermed ha flere MAC og IP adresser assosiert med seg.
- Hver node har sin egen faste MAC-adresse.
- Svitsjer har ikke MAC-adresser. Dette fordi jobben til linklaget er å sende datagram mellom verter og rutere, altså trenger man ikke adressen til svitsjen.
- MAC-adressen er 6 bytes lang og er unik. IEEE styrer MAC adresserommet og passer på dette. Den er også den samme, uansett hvor datamaskinen flytter seg til
- ARP (Address Resolution Protocol) er protokollen som oversetter mellom nettverkslagsadressene (IP-adresser) og link layer adresser (MAC). Dette gjør den ved å slå opp i en tabell.

    - ARP finner kun adressen til endesystemer eller rutere i det samme subnettverket. ARP tabellen inneholder også en TTL (Time to live) verdi som indikerer når mappingen skal slettes fra tabellen. Dersom den aktuelle adressen ikke er i tabellen må senderen lage en spesiell pakke. Denne har flere felter deriblant MAC og IP adresse til sender og mottaker. Denne skal sendes til alle endesystemene i subnettet, og endesystemet med den IP-adressen som står i feltet for mottaker svarer med MAC-adressen sin.
    - Spør i subnettet: Leter etter IP-adresse 192.168.x.x, sender ut broadcast til subnettet for å finne MAC-adressen. Noden med denne IP-adressen svarer med sin MAC-adresse. Senderen lagrer så denne, og de kan kommunisere med hverandre.
    - ARP er både en nettverksprotokoll (den har felter som inneholder nettverksadresser) og en linklayer protokoll (felter for linklayer adresser).
- Ethernet er en veldig populær protokoll bruk i LAN der man utveksler frames gjennom Ethernetkabler (twisted copper pair)
    - Data field - 46 til 1500 bytes
    - Destination address - MAC adressen, 6 bytes
    - Source address - MAC adressen, 6 bytes
    - Type field - 2 bytes som gir Ethernet muligheten til å multiplekse nettverkslagsprotokoller
    - CRC - 4 bytes for feildetektering
    - Preamble - 8 bytes. De første 7 skal vekke mottakerens adaptere og synkronisere klokken med den til senderen
- Alle Ethernet teknologier er forbindelsesløs service til nettverkslaget. Når adapter A vil sende en frame til adapter B, så sender A framen til LAN uten å handshake med B.
    - Unreliable service, fordi når B mottar frame fra A, og kjører den gjennom CRC, så sier den ikke ifra om den besto testen eller ikke (ikke ACK eller NAK). Hvis den ikke består, blir den bare rejected av B.
    - Dette gjør Ethernet enkel og billig, men skaper også problemer og strømmer av datagrammer kan ha gaps.
    - Oppdager B gapsene? Kommer an på om det er UDP eller TCP. Hvis UDP, ja man ser gaps. Hvis TCP så vil ikke A få noen ACK, som fører til at den sender på nytt. Ethernet vet ikke selv om det sender en ny datagram pakke eller om det er en retransmission av en gammel, eller om datagrammet den sender har data som allerede har blitt sendt
- Svitsjen er transparent til endesystemene/ruterne innen subnettverket; det vil si at en node adresserer en frame til en annen (i stedet for til svitsjen) og sender dermed rammen inn i LAN, uvitende om at svitsjen vil motta framen og videresende den.

- Svitsjer har to hovedfunksjoner:
    - Filtrering: bestemmer om en ramme skal sendes videre til et grensesnitt, eller droppes
    - Videresending: bestemmer til hvilket grensesnitt rammen skal sendes og så flytter den dit
    - Begge disse bruker en svitsjtabell (som inneholder MAC adressen, grensesnittet som fører den dit og en tidsvariabel for når den ble plassert i tabellen)
    - Svitsjene har egenskapen at tabellene deres bygges på egenhånd, uten noen innvendinger fra nettverksadministrasjon. Derfor sies svitsjene å være self-learning. Derfor har de også plug-and-play egenskapen
- Fordeler/egenskaper ved svitsjing, kontra broadcastlinjer
    - Eliminerer kollisjoner - overfører bare en ramme over et segment av gangen
    - Heterogene koblinger - koblingene isoleres fra hverandre og kan ha forskjellige overføringshastigheter.
    - Management - svitsjen gjør mye selv og er desentralisert
- Rutere videresender på IP-adresser, svitsjer på MAC-adresser. De jobber på forskjellige nivåer i hierarkiet. Ruter er en nivå-3 pakkesvitsj (network layer), mens svitsjer er nivå-2 pakkesvits (linklayer). Funksjonene til rutere og svitsjer er forskjellige selv om de noen ganger integreres til en enkel enhet.
- Fordeler med svitsjer:
    - Plug and play
    - Høye filtrerings- og videresendingsrater
    - Må kun prosessere frames opp til nivå 2
- Ulemper med svitsjer
    - Et stort svitsjnettverk vil kreve en stor ARP tabell -> mye ARP trafikk
    - Kan forekomme broadcaststormer (at en host klikker og sender en endeløs strøm av Ethernet broadcast frames)
- Fordeler med rutere
    - Fordi nettverksadressering er hierarkisk vil pakker normalt ikke gå i sykel gjennom rutere selv når nettverket har overflødige veier.
        - Pakker er ikke begrenset til et spenntre (som ved svitsj) og kan bruke beste vei mellom kilde og destinasjon
    - Tilbyr brannmurbeskyttelse mot nivå 2 broadcaststormer
- Ulempen ved rutere
    - Ikke plug-and-play -> trenger at IP adressene blir konfigurert
    - Høyere prosesseringstid per pakke enn svitsjer (fordi de må prosessere opp gjennom nivå 3).

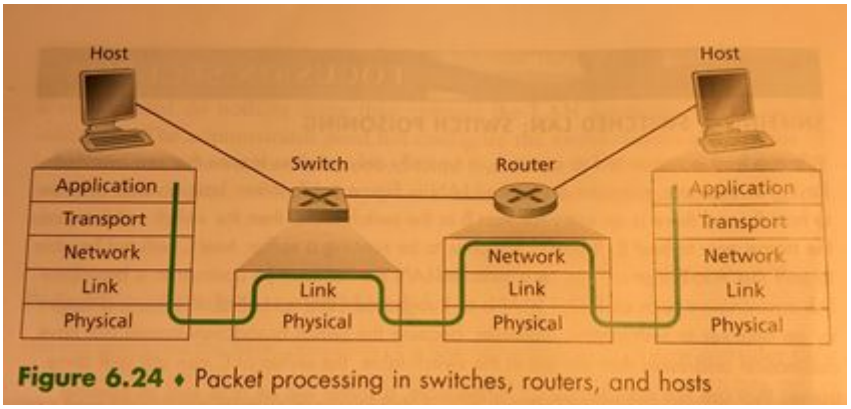
- 

Figure 6.24 ♦ Packet processing in switches, routers, and hosts

Chapter 7 - Trådløse- og mobile nettverk

7.1 - Intro

- Vi kan identifisere følgende elementer i et trådløst nettverk
  - Trådøse endesystemer - laptop, tablet, smarttelefon etc
  - Trådløse koblinger - et endesystem knyttes til en base station eller et annet trådløst endesystem ved trådløse kommunikasjonskoblinger
  - Basestasjon - Ansvar for å sende og motta data til og fra trådløse endesystemer som den assosieres med. Også for å koordinere overføringer fra flere endesystemer. Aksesspunkter (access point) i LAN er eksempel på en basestasjon. Når et mobilt endesystem beveger seg fra området til en basestasjon og inn i området til en annen, vil det endre basestasjon. Dette kalles handoff.
- Nettverksinfrastruktur - Dette er et større nettverk der et trådløst endesystem kan ønske å kommunisere
- Infrastructure mode - Når et endesystem er assosiert med en basestasjon er det i infrastructure mode. Flere hosts er koblet til et Aksesspunkt (AP), som kommuniserer med "omverdenen"
- Ad hoc - Trådløse endesystemer har ingen infrastruktur på dette, og må selv stå for routing, address assignment og DNS-like name translation. De er koblet mot hverandre heller enn mot et AP.
- Noen typer trådløse nettverk er:
  - Single-hop, infrastrukturbasert - En basestasjon som er koblet til et større nettverk. All kommunikasjon mellom basestasjon og trådløse endesystemer skjer over én hop. 4G LTE nettverk er slik
  - Single-hop, infrastrukturløs - Ingen basestasjon, Bluetooth eksempel
  - Multi-hop, infrastrukturbasert - Noen noder må kommunisere gjennom andre trådløse noder for å kommunisere med basestasjonen
  - Multi-hop, infrastrukturløs - ingen basestasjon, nodene kan være mobile

7.2 - Trådløse linker og nettverkskarakteristikk

- Det er stort sett på link layer at vi finner forskjeller mellom trådløse og ikke-trådløse nettverk. Forskjellene på en ikke-trådløs og en trådløs kobling er:
    - Synkende signalstryke: Trådløse signaler sin styrke synker med avstanden og når den passerer gjennom ting
    - Interferens fra andre kilder: To kilder som overfører over samme frekvensbånd vil forstyrre hverandre
    - Multipath propagering: Oppstår når deler av elektromagnetiske bølger reflekterer av objekter på bakken og tar ulike veier/lengder mellom sender og mottaker. Dette resulterer i at mottakeren mottar blurry signaler
- Mye tyder dermed på at bitfeil vil være mer vanlig ved trådløse koblinger enn ikke-trådløse. Trådløse link-protokoller har dermed kraftigere CRC koder, samt link-level pålitelig overføring av data protokoller som sender korrupte rammer på nytt
- Når en mottaker mottar et trådløst signal, mottas det både elektromagnetisk signal (det som ble sendt) og bakgrunnsstøy. Signal-to-noise ratio, SNR, er et relativt mål på styrken til signalet (informasjonen som overføres) og støyet. En høyere SNR gjør det lettere for mottakeren å trekke ut signalet fra støyet.
    - For et gitt moduleringsskjema: Jo høyere SNR, desto lavere BER (bit error rate)
    - For en gitt SNR vil et moduleringsskjema med høyere bit transmission rate ha en høyere BER
    - Dynamisk selektering av moduleringsteknikker kan brukes for å tilpasse moduleringsteknikken til forholdene ved overføring.
- Skjulte terminaler - problemet
    - Hvis både stasjon A og stasjon C prøver å overføre noe til B, uten at A og C kan se hverandres overføringen pga hindringer i miljøet (bygning, fjell etc). Det oppstår en undetected kollisjon i B.
- Fading
    - At signalet fra A og C ikke er sterke nok til å oppdage hverandre, men de er sterke nok til å interferere med hverandre i B.

7.3 - WiFi: 802.11 Trådløse LAN

- IEEE 802.11 trådløs LAN er kjent som WiFi og er den mest populære trådløse LAN
- Den fundamentale blokken i arkitekturen er et basic service set, BSS. Et BSS inneholder en eller flere trådløse stasjoner og en sentral basestasjon, kjent som aksesspunktet (AP). Hver 802.11 trådløse enhet har en 6-byte MAC adresse som lagres i stasjonens adapter. Det samme har hver AP.
- IEEE 802.11 stasjoner kan også gruppere seg sammen og danne ad hoc nettverk (et nettverk uten sentral kontroll - infrastrukturløs)

- I 802.11 må trådløse stasjoner assosiere seg med en AP før de kan sende og motta nettverslagsdata
    - Når administratoren installerer en AP blir APen tilegnet en et/to ord Service Set IDentifier (SSID). AP må også tilegnes et kanalnummer. 802.11 opererer i frekvensbånd 2.4 GHz til 2.4835 GHz. Innen dette båndet finner man 11 delvis overlappende kanaler. To kanaler overlapper ikke dersom de er skilt av minst 4 eller flere kanaler. Mengden 1,6,11 er dermed den ene mengden av 3 ikke overlappende kanaler.
- En WiFi jungel er et fysisk sted hvor en trådløs stasjon mottar sterkt nok signal fra to eller flere AP.
- For å få internettilgang må den trådløse enheten velge nøyaktig et av nettverkene og dermed assosiere seg med nøyaktig en AP.
    - Med assosiere menes at enheten lager en virtuell linje mellom seg selv og AP
- 802.11 standarden krever at AP periodisk sender beacon frames som inneholder AP sin SSID og MAC-adresse. Enheten vår fanger opp disse beacon frames sendingene.
    - Typisk velger enheten den AP sin beacon frame som mottas med høyest signalstyrke.
    - Prosessen med å skanne kanaler og lytte etter beacon frames kalles passiv skanning.
    - En trådløs enhet kan også utføre aktiv skanning ved å broadcaste en frame som vil mottas av alle AP-ene innen enhetens rekkevidde. Det forekommer en request/response handshake før enheten kan sende en DHCP discovery message for å oppnå IP-adressen til nettverket.
    - Noen ganger må enhetene identifisere/autentisere seg før de kan knytte seg til en AP (ofte gjennom MAC-adressen), evt brukernavn og passord.
- En multiple access protocol trengs for å koordinere overføringer på et trådløst nettverk (da flere enheter muligens ønsker å overføre data samtidig). Designere av 802.11 har valgt random access protocol for 802.11 (CSMA/CA)
    - Hver stasjon lytter på kanalen før de overfører, og lar være å sende dersom kanalen er opptatt
    - I motsetning til Ethernet (som også bruker CSMA) så bruker 802.11 collision avoidance teknikker, mens Ethernet bruker collision detection (CSMA/CD)
        - Kostbart å bygge hardware
        - Ville uansett ikke oppdaget alle kollisjonene
    - 802.11 bruker også en link-layer ACK/retransmission scheme (ARQ) på grunn av frekvente bitfeil. Dette innebærer at når destinasjonstasjonen mottar en frame som har bestått CRC, så venter den en kort periode (Short Inter-frame Spacing, SIFS) før den sender tilbake en ACK frame.

Hvis avsenderen ikke mottar denne innen en viss tidsperiode, vil den sende framen på nytt ved bruk av CSMA/CA.

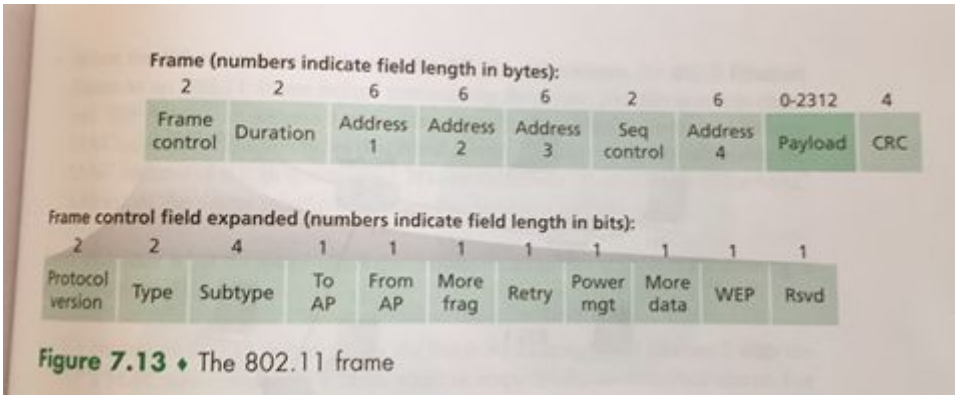
- DIFS - Distributed Inter-frame Space - en kort tidsperiode rett før man skal sende en frame over kanalen,
- SIFS - Short Inter-frame Spacing - en kort tidsperiode fra man mottar signalet til man sender ut ACK igjen.
- Når to noder skal sende en pakke samtidig, men merker at kanalen er opptatt vil det ved collision detection bli en kollisjon, da begge vil sende samtidig rett etter kanalen blir ledig.
    - Ved collision avoidance vil derimot begge bestemme en random backoff verdi som indikerer hvor lenge de skal vente til de forsøker igjen. Får de forskjellige verdier vil de ikke kollidere, noe som ville vært ille siden CSMA/CA ikke klarer å merke når kollisjoner oppstår.
- Skjulte terminaler innebærer at stasjonene ikke kan høre at kanalen er opptatt. 802.11 MAC protokollen inkluderer en reserveringsordning som hjelper å unngå kollisjoner selv ved skjulte terminaler.
    - For at en stasjon ikke skal sende en frame mens det allerede er en annen på kanalen, må den sende en RTS (Request to Send) kontrollframe med hvor lang tid det vil ta å overføre pakken til AP. Om kanalen er tilgjengelig svarer AP med en CTS (clear to send) frame. Denne indikerer i tillegg til alle andre stasjoner at de ikke skal overføre noe i denne perioden.
    - Selv om RTS og CTS hjelper å redusere kollisjoner, introduserer det forsinkelse og konsumerer kanalressurser. På grunn av dette brukes det bare ved lange DATA frames der risikoen for kollisjoner er større. Man setter en RTS-threshold og sier at hvis framen er lenger enn threshold, så skal det brukes RTS/CTS
- IEEE 802.11 framen:
    - 

Figure 7.13 ♦ The 802.11 frame
    - Payload - består typisk av et IP diagram eller en ARP pakke
    - Adressefelter - største forskjellene fra Ethernet er at 802.11 inneholder fire adressefelter, der hver kan holde en 6-byte MAC adresse. Det fjerde adressefeltet benyttes når AP videresender frames til hverandre i ad

hoc modus. Adresse 1 er for mottaker, 2 for avsender og 3 er for rutergrensesnittet som knytter sammen nettverkene.
    - Seq control - Tillater mottaker å skille mellom nye overføringer og retransmisjoner
    - Duration - sier noe om hvor lang tid overføringen tar (både for data og for RTC/CTS)
- Når en stasjon forflytter seg mellom BSS-er innen samme IP nettverk, trengs ikke IP-adressen å oppdateres. Den tilhørende MAC-adressen må derimot oppdateres fra AP1 sin til AP2.
    - Dette kan feks gjøres ved at AP2 sender en broadcast Ethernet frame med H1 sin source address til svitsjen (siden det er samme IP-nettverk, ruter om det er ulike IP-nettverk) rett etter den nye assosiasjonen til H1. Svitsjen oppdaterer da sin videresendingstabell (forwarding table) og tillater H1 å nås gjennom AP2.
- 802.11 implementasjoner har en rate adaption teknikk (styrer SNR/BER) som innebærer at den skifter underliggende moduleringsteknikker basert på karakteristikkene til kanalen (likner på kontrollmekanismen til TCP)
- Power management
    - Metode for å minimere tiden 802.11 noder er aktive.
    - En node indikerer til aksesspunktet at den vil gå i sovemodus ved å sette power-management bitet i headeren til 1. En timer blir da satt til å vekke noden rett før APen vil sende ut beacon frame. AP vet at denne noden skal “sove” og kommer ikke til å sende noen frames til den node, men lagre frames til den i buffer.
    - Noden våkner til angitt tid og er aktiv. Beacon frame som sendes ut inneholder en liste med noder sine frames som har blitt cached. Hvis det er ingenting i buffer, kan noden sove igjen. Hvis det er frames i buffer, kan noden spørre etter disse ved å sende en polling message til AP.
    - Teknikken gjør at en node som har ingen frames å sende eller motta kan sove 99% av tiden.

7.4 - Mobil internettaksess

- Når vi ikke kan benytte WiFi, kan vi bruke GSM nettet
- Førstegenerasjons 1G systemer ble kun designet for lydkommunikasjon. Disse er så godt som distinkte nå og ble erstattet av 2G digitale systemer (også kun for lyd). 2.5G ble designet til å støtte data. 3G systemer støtter lyd og data med høyhastighetskoblinger. 4G systemene i dag er basert på LTE systemer og tilbyr integrerte lyd og data i multi-Megabit hastigheter
- Mobilt (cellular) referer til det faktum at et område som dekkes av et mobilt nettverk er delt inn i flere geografiske områder kjent som cells. Hver celle

inneholder en base transceiver station (BTS) som overfører signal til og mottar fra mobile stasjoner i cellen.
- En GSM sin base station controller (BSC) vil typisk tjene titalls BTS-er. Rollen til BSC er å allokere BTS radiokanaler til mobile brukere ved å utføre paging (finne cellen hvor mobilbrukeren holder til) og handoff
- 3G bruker GSM nettet til tale, og legger på datafunksjonalitet
- Finnes to typer noder i 3G nett:
    - SGSN - ansvarlig for å levere datagram til/fra mobilnoder
    - GGSN - kobler sammen flere SGSN til et større internett, siste delen før noden kobles til internett
- 4G / LTE (Long-Term Evolution
    - Endring fra 3G: Alt over IP og enhanced radio access network
    - LTE har både tale og data i samme IP datagram (i motsetning til 3G)
    - Data og kontrollplanet er separert
    - Bruker TDM og FDM. Hver aktive node får en eller flere 0,5 ms time slots og en eller flere frekvenskanaler. Jo flere slots man har, jo høyere transmission rate.

Chapter 8 - Security

8.1 - Hva er nettverkssikkerhet

- Sikker kommunikasjon har følgende egenskaper:
    - Konfidensialitet: Bare senderen og den tiltenkte mottakeren skal kunne forstå innholdet i meldingen
    - Meldingsintegritet: Passer på at meldingen ikke endres
    - Endepunktautorisering: Både sender og mottaker skal kunne bekrefte identiteten til den andre parten involvert i kommunikasjonen
    - Operasjonell sikkerhet: Passe på at ikke nettverkene kan bli compromised
- En inntrenger kan
    - Eavesdroppe (overhøre) - sniffing og recording kontroll -og datameldinger på kanalen
    - Modifisere, innsette og slette meldinger eller innhold i meldinger

8.2 - Kryptografi

- Kryptografi tillater senderen å kryptere data slik at ingen andre enn den tilegnede mottakeren kan forstå meldingen
    - Sendingen i dens opprinnelige form kalles klartekst
    - Den krypterte teksten kalles ciphertext
- Symmetrisk nøkkel kryptografi
    - En variant er Ceasar cipher
    - Bytter ut hver bokstav med bokstaven k bokstaver senere

- En forbedring av Ceasar er å bruke en monoalfabetisk cipher, som erstatter en bokstav i alfabetet med en annen (som en hash-tabell). Her er det hele 25! (10^26) mulige ordninger av bokstaver, istedet for 25. Men selv her kan man med diverse statistiske beregninger, og ved eventuell informasjon om avsender og mottaker klare å redusere antall muligheter betraktelig
- Vi kan identifisere tre forskjellige scenarioer inntrenger har, basert på informasjonsgrunnlaget:
    - Kun ciphertekst-angrep: Uten noen info om avsenderen eller mottakeren kan det kun benyttes statistiske beregninger (som å analysere frekvente ord og bokstaver) for å redusere krypteringsmulighetene
    - Angrep med bakgrunn i kjennskap til klarteksten: Dersom inntrengeren for eksempel vet at ordene Bob og Alice er i meldingen kan 7 av ordningene avdekkes og "kun" 18! gjenstår.
    - Valgte klartekst-angrep: Ved et slikt angrep klarer inntrengeren å velge klarteksten og oppnå ciphertekstformen.
- I dag er det to klasser av symmetriske krypteringsteknikker; strømcifere og blokkcifere.
    - Ved blokkcifer vil meldingen som skal krypteres prosesseres i blokker av k bits. Dersom k=64 blir meldingen delt inn i blokker av 64 bits og hver blokk krypteres individuelt. Ved en monoalfabetisk cifer er det (2^k)! mulige mappinger/krypteringer. Dersom Alice og Bob begge kjenner nøkkelen kan de lett kryptere og dekryptere meldingene. For k=64 må Alice og Bob holde styr på en tabell med 2^64 input verdier. Dette er uaktuelt. I stedet bruker man typisk funksjoner som tilfeldig permuterer bits der man deler 64 bits inn i mindre deler og hver del prosesseres hver for seg med en egen tabell før man slår de sammen, endrer rekkefølge og gjør prosessen på nytt n ganger.
    - Populære blockciphere er DES, 3DES og AES.
- Cifer-blokk kjeding
    - Håndterer problemet som oppstår når to blokker klartekst er identiske og dermed krypteres likt, som gjør det lettere å dekryptere meldingen for inntrengere, ved å introdusere et snev av tilfeldighet inn i ciferteksten slik at identiske klartekstblokker produserer ulike cifertekstblokker. Ideen er at senderen lager et tilfeldig k-bit tall r(i) for den i-te blokken og beregner c(i) = Ks(m(i) XOR r(i)). Mottaker mottar både c(i) og r(i) slik at den kan dekrytpere meldingen.
    - Problemet som oppstår nå er at Alice må overføre dobbelt så mange bits. For å løse det bruker blokkcifer en teknikk kalt **Cipher Block Chaining (CBC).**
- Offentlig nøkkel kryptering

    - Ideen med offentlig kryptering er ganske enkel. I stedet for at sender og mottaker deler to nøkler, deler de nå en offentlig nøkkel, mens mottakeren har en privat dekrypteringsnøkkel. Den offentlige nøkkelen er tilgjengelig for alle. Kb- (privat) og Kb+ (offentlig).
    - Kb-(kb+(m)) = m
    - Dersom offentlig kryptering skal fungere må kryptering/dekryptering gjøres slik at det er umulig å bestemme den private nøkkelen til mottakeren
- RSA er en slik algoritme
    - For å generere offentlige og private nøkler, skjer følgende
        - Velg to store primtall p og q
        - Beregn n =pq og z = (p-1)(q-1)
        - Velg et nummer e mindre enn n, og som ikke har noen felles faktor med z
        - Finn et tall d slik at ed-1 er delelig med z (ed mod z = 1)
        - Den offentlige nøkkelen er (n,e), den private er (n,d)
        - For å kryptere bruker man $c = m^e \bmod n$
        - Dekryptere med $m = c^d \bmod n$.

8.3 - Meldingsintegritet og digitale signaturer

- For å autentisere en melding
    - 1. forsikre seg om at meldingen faktisk kommer fra den vi tror er avsenderen.
    - 2. forsikre seg om at meldingen ikke har blitt endret på veien
- En kryptografisk hashfunksjon tar en input m og gjør den om til H(m), der H er en hashfunksjon og H(m) er kjent som hashen. En kryptografisk hashfunksjon krever følgende egenskap:
    - Det er matematisk umulig å finne to ulike meldinger x og y der H(x) = H(y)
    - En inntrenger kan altså ikke bytte en melding med en annen som er beskyttet av hashfunksjonen. Checksum overholder ikke denne egenskapen, flere forskjellige inputverdier kan gi samme hashverdi.
    - Hashfunksjonene vi bruker i dag i forbindelse med kryptering er MD5 og SHA-1.
- Hvordan sørger vi for meldingsintegritet?
    - Alice lager en melding m og kalkulerer H(m).
    - Alice legger så H(m) til i meldingen m og lager en utvidet melding (m, H(m)) og sender den til Bob
    - Bob mottar meldingen (m, h) og beregner H(m). Dersom H(m) = h er alt OK.

- Så i tillegg til hashfunksjonen, vil Alice og Bob trenge en hemmelig s som de deler. S er en streng av bits som kalles autentiseringsnøkkel. Da vil meldingsintegritet skje som følger:
    - Alice lager en melding m, slår sammen m med s for å lage h+s og beregner og h(m+s). H(m+s) kalles MAC - message authentication code
    - Alice legger så H(m+s) til i meldingen m og lager en utvidet melding (m, H(m+s)) og sender den til Bob
    - Bob mottar meldingen (m, h) og vet s. Han beregner MAC (H(m+s)). Dersom H(m+s) = h er alt OK
    - MAC trenger ikke en krypteringsalgoritme, dette er for meldingsintegritet og ikke konfidensialitet. Alt sendes som klartekst, men hvis utregningen ikke stemmer så beviser det at noen har tuklet med en
    - Bruker man kryptering og MAC opprettholder man både konfidensialtet og integritet
        - Encryption does not provide integrity by itself
        - MAC (integrity) does not provide confidentiality by itself
    - HMAC er mest populær i dag, den kjører data og autentiseringsnøkkelen gjennom hashfunksjonen to ganger.
        - Bruker enten MD5 eller SHA-1
    - MD5 er ikke sikker lenger (står ikke i pensum, men google/youtube)
        - Den kan crackes via et googlesøk
- Anta Bob ønsker å signere et dokument m
    - For å signere dokumentet bruker Bob sin private nøkkel Kb- til å beregne Kb-(m). Bob ønsker ikke å endre innholdet i dokumentet, men kun signere
    - Alice tar da Bob sin offentlige nøkkel Kb+ som er assosiert med dokumentet. Så beregner hun Kb+(kb-(m)) = m. Alice vet nå at kun Bob kan ha signert dokumentet fordi:
        - Den som signerte dokumentet må ha brukt den private nøkkelen Kb- som kun Bob har tilgang til.
    - Det er også viktig å notere at dersom m er blitt modifisert til m' så vil ikke signaturen Bob lagde for m være gyldig for m', siden Kb+(kb-(m)) != m'. Digitale signaturer tilbyr altså også meldingsintegritet.
    - Kryptering og dekryptering krever mye ressurser. En løsning er å bruke hashfunksjon for digital signatur, altså sender Kb-(H(m)). Denne er mindre og krever mindre ressurser å regne på
- Å bygge en offentlig nøkkel til en spesiell entitet gjøres av en Certification Authority (CA). CA sin jobb er å validere entiteter og identifisere sertifikater. En CA har følgende roller:
    - Verifiserer at en entitet (en person, ruter osv) er den den begir seg ut for å være

    - Når CA verifiserer identiteten til en entitet, lager den et sertifikat som knytter entiteten sin offentlige nøkkel til entiteten.

8.5 - Sikker E-mail

- Det er mulig å tilby sikkerhetstjenester i alle fire øverste lagene i protokollstakken.
- Når sikkerhet tilbys i applikasjonslaget, vil applikasjonen som bruker protokollen høste av en eller flere sikkerhetstjenester som konfidensialitet, integritet eller autentisering.
- Når sikkerhet tilbys i transportlaget, vil alle applikasjoner som bruker protokollen nyte av sikkerhetstjenestene til transportprotokollen.
- Når sikkerhet tilbys i nettverkslaget vil alle transportlagssegmenter nyt av sikkerhetstjenestene i nettverkslaget (på en host-to-host basis).
- Når sikkerhet tilbys på linklaget vil alle frames som reiser over koblingen motta sikkerhetstjenester fra koblingen
- Sikker email:
    - Konfidensialitet kan tilbys ved kryptering, best ved offentlige krypteringsteknikker. Autentisering og integritet løses ved digital signering.
    - Bruker en session key for å kryptere.
        - Velges en random symmetrisk session key Ks, krypterer meldingen m med session key, krypterer dette med mottakers public key, slår sammen kryptert melding og nøkkel for å lage en pakke og sender til mottakeren
        - Mottakeren bruker sin private nøkkel til å dekryptere og skaffe session key, og bruker denne session key til å dekryptere meldingen

8.6 - Sikre TCP-koblinger: SSL (og TLS)

- Denne forsterkede versjonen av TCP er allmenn kjent som Secure Socket Layer (SSL). SSL brukes av nettleseren når URL-en begynner med https i stedet for http. Uten sikkerhetstiltak kan man ved kjøp på nett risikere
    - Ingen konfidensialitet som gjør at andre kan få kreditkortopplysninger
    - Ingen integritet som gjør at andre kan manipulere ordene dine
    - Ingen autentisering som gjør at siden du handler på kan drives av andre enn de du tror
- SSL i det store bildet
    - En simplifisert beskrivelse av SSL, kalt nesten-SSL. Nesten-SSL har tre faser: Handshake, key derivation og data transfer
        - Handshake - I denne fasen må Bob:
            - Etablere en TCP kobling med Alice
            - Verifisere at Alice faktisk er Alice

            - Sende Alice en master sikkerhetsnøkkel som vil brukes av både Alice og Bob til å generere alle symmetriske nøkler de trenger for SSL session.
                - Masternøkkelen krypteres av senderen med mottakeren sin offentlige nøkkel, og mottakeren dekrypterer med sin private nøkkel
        - Key derivation - Alice og Bob bruker MS (master securitykey) til å generere fire nøkler
            - Eb = krypteringsnøkkel for dataen som sendes fra Bob til Alice
            - Mb = MAC nøkkel for dataen som sendes fra Bob til Alice
            - Ea = krypteringsnøkkel for dataen som sendes fra Alice til Bob
            - Ma = MAC nøkkel for dataen som sendes fra Alice til Bob
        - De to krypteringsnøklene brukes til å kryptere data, mens de to MAC nøklene brukes til å verifisere integriteten til dataen.
        - Data transfer - Nå som Alice og Bob deler de samme fire nøklene kan de starte å sende sikret data til hverandre over TCP koblingen. SSL bryter datastrømmene inn i records og legger en MAC til hver record for integritetssjekk og krypterer så record + MAC. Bob bruker Mb for å lage MAC-en. Den kommer som input sammen med Mb inn i en hash-funksjon. Bob krypterer pakken + record med Eb før den sendes til TCP.
- For å unngå at inntrengere kan endre rekkefølgen og fjerne pakker, da info om sekvensnummer ikke er krypterte, velger man å lagre infoen om sekvensnummer i MAC adressen. MAC-en er da en hash av dataen, MAC nøkkelen og det daværende sekvensnummeret
- SSL-handshake
    - 1. Klient sender en liste med krypteringsalgoritmer den kan, i tillegg til en nonce (Number once).
    - 2. Tjeneren velger en symmestrisk, offentlig og MAC algoritme fra listen som den kan, sender tilbake til klienten hva en valgte, i tillegg et sertifikat og nonce
    - 3. Klienten verifiserer sertifikatet, henter ut tjeneren sin offentlige nøkkel. lager en Pre-Master Secret (PMS), krypterer PMS med tjeneren sin offentlige nøkkel og sender kryptert PMS til tjener
    - 4. Klienten og tjeneren regner ut MS hver for seg ved å bruke den samme nøkkelfunksjonen ut i fra PMS og nonces. MS er deretter delt opp i fire (som vi så); to krypteringsnøkler og to MAC nøkler. Heretter er alle meldinger mellom disse kryptert og autentisert
    - 5. Klienten sender en MAC av alle handshake meldingene
    - 6. Tjeneren sender en MAC av alle handshake meldingene.

    - De to siste er for å unngå tampering
        - Listen med algoritmer i 1. er sendt i klartekst og kan endres av en inntrenger, slik at tjeneren bare kan velge mellom svake algoritmer. Derfor kan vi med sjekken i 5 og 6 sjekke om det har skjedd en endring
- Nonces gjør at man ikke kan sende en lik melding som tidligere har blitt gjort, med samme nonces. Siden det byttes hver gang vil ikke en som har sniffet på en melding, kunne sende denne på nytt med samme nonce og bestå integritetssjekken
    - Nonces brukes mot connection replay attack og sekvensnummer for å forsvare mot replaying individuelle pakker under en pågående session
- SSL session kan ikke avsluttes vanlig med TCP FIN, da kunne hvem som helst bare gått inn og sendt TCP FIN. Løses ved å legge med i type-feltet om en record skal avslutte SSL session eller ikke. Hvis mottaker får TCP FIN før en slik melding, så vet mottakeren at det er noe feil.

8.7 - Nettverklagssikkerhet

- IP sikkerhetsprotokollen IPsec tilbyr sikkerhet på nettverkslaget. IPsec sikrer IP datagrammer mellom to nettverkslagsentiteter, inkludert endesystemer og rutere. Mange instutisjoner og organisasjoner bruker IPsec til å lage virtuelle private nettverk (VPN) som kjører på det offentlige Internett.
- Innføring av konfidensialitet på nettverkslaget innebærer at entitetene krypterer payloaden i datagrammene. Payloaden kan være TCP, UDP, ICMP melding osv. Med dette på plass ville vi kunne skjule alt som sendes mellom to entiteter - network layer security gir altså “blanket coverage”
- En instutisjon som strekker seg over flere geografiske områder vil ofte ønske et IP nettverk der endesystemene og tjenerne kan sende konfidensiell data til hverandre på en sikker måte.
    - For å få til dette kan man utplassere et eget fysisk nettverk, inkludert rutere, koblinger, DNS infrastruktur som er separat fra Internett.
    - Et slikt disjunkt nettverk, dedikert til en partikulær organisasjon, kalles et privat nettverk.
    - Dette kan være svært kostbart, så i stedet velger man stort sett å lage et VPN på det offentlige Internett. Med VPN sendes institusjonens trafikk over det offentlige Internett i stedet for over et fysisk uavhengig nettverk.
    - For å bevare konfidensialitet krypteres inter-office trafikken før det når det offentlige Internett.

8.8 - Sikre trådløse LAN

- Det virker å være enighet i at de originale spesifikasjonene for 802.11 inneholder mange sikkerhetshull. Vi diskuterer nå disse

sikkerhetsmekanismene kjent som Wired Equivalent Privacy (WEP). Som navnet antyder er det meningen at WEP skal tilby sikkerhet på lik linje med ikke-trådløse nettverk, men det er som nevnt noen hull i sikkerheten her.

- IEEE 802.11 WEP protokollen tilbyr autentisering og kryptering mellom et endesystem og et aksesspunkt ved symmetrisk kryptering
  - 1) En trådløs host ber om autentisering av et AP
  - 2) AP svarer to autentiseringen med en 128 byte nonce
  - 3) Trådløs host krypterer nonce med symmetrisk nøkkel som den deler med AP
  - 4) AP dekrypterer noncen.
- En hemmelig symmetrisk nøkkel på 40 bit (kjent hos sender og mottaker) kombineres med en 24 bit IV (initialization vector) som er ulik for frame sending slik at framesene krypteres med ulike 64 bit nøkler. Dermed er det 2^24 ulike nøkler.
- Med algoritmen som benyttes er det 99% sannsynlig at den samme 64 bit nøkkelen er blitt benyttet etter 12000 frames, som med en frame size på 1 kbyte og overføringsrate på 11Mbps innebærer noen sekunder. Siden IV overføres i klartekst vil en lytter kunne vite når et duplikat IV er blitt benyttet (og dermed vite den hemmelige nøkkelen).

- Det finnes andre sikkerhetsbeskymringer ved WEP også, som ligger i algoritmen (RC4 og CRC), som går utover meldingsintegriteten da inntrengere kan endre innholdet uten at mottaker merker det som følge av svakheter i algoritmen.

8.9 - Brannmur

- En brannmur er en kombinasjon mellom hardware og software som isolerer en organisasjons interne nettverk fra Internettet, og tillater pakker å passere, eller blokkerer pakker fra å komme til, organisasjonens interne nettverk.
- En brannmur lar nettverksadministrasjonen kontrollere tilgang mellom den ytre verden og ressursene innen det administrerte nettverket. En brannmur har tre mål:
  - All trafikk som skal fra utsiden til innsiden og vice versa må passere gjennom brannmuren
  - Kun autorisert trafikk vil tillates å passere
  - Brannmuren selv er immun mot penetrering, dvs uautorisert adgang
- Cisco og CheckPoint er to ledende selgere av brannmurer i dag. Brannmurer kan klassifiseres innen tre kategorier: tradisjonelle pakkefiltere, tilstandsbaserte filtere og applikasjonsporter
- Tradisjonelle pakkefiletre
  - All trafikk som skal til et internt nettverk passerer gjennom en gateway ruter som knytter det interne nettverket med dens ISP. Det er ved

denne ruteren at pakkefiltrering forekommer. Filtreringsavgjørelser blir typisk basert på:
    - IP kilde og destinasjonsadresse
    - Protokolltype i IP datagram feltet: TCP, UDP, ICMP
    - TCP/UDP kilde og destinasjonsport
    - TCP flagg bits: SYN, ACK
    - ICMP meldingstype
    - Forskjellige regler for datagrammer som entrer eller forlater nettverket
    - Forskjellige regler for forskjellige rutergrensesnitt
- Tilstandsbaserte pakkefiltere (stateful packet filters)
    - Tilstandsfiltere overvåker TCP-koblingene og bruker denne kunnskapen til å ta filtreringsavgjørelser.
    - Filteringsavgjørelsen er gjort på hver pakke isolert
    - Brannmuren holder oversikt over TCP-koblingene og ser handshaking og FIN.
    - Har et felt kalt "check connection" i connection table
- Applikasjonsporter
    - For å oppnå høyere nivå av sikkerhet må brannmurer kombinere tradisjonelle pakkefiltere med applikasjonsporter. Informasjon om identiteten til interne brukere er applikasjonsnivå data og ikke inkludert i IP/TCP/UDP headere.
    - Ulempe: trengs forskjellige applikasjonsporter for hver applikasjon. Går også ut over ytelse, siden all data blir videresendt via gatewayen.

Chapter 9 - Multimedia

9.1 - Multimedia-applikasjoner

- Defineres som enhver nettverksapplikasjon som bruker lyd eller video
- Egenskaper ved video
    - Den viktige karakteristikken med video er dens høye bit overføringsrate (100kbps - 2Mbps)
    - Det er to typer overflødighet innen video, begge kan utnyttes ved videokomprimering
        - Overflødighet i rom (spatial redundancy) - Overflødighet innen et gitt bilde. Intuitivt har et bilde som for det meste består av tomrom en høy grad av overflødighet og kan effektivt komprimeres uten å måtte ofre bildekvalitet
        - Overflødighet i tid (temporal redundancy) - reflekterer repetisjon fra et bilde til det påfølgende bildet. Dersom det påfølgende bildet er helt identisk er det ingen grunn til å kode på nytt det påfølgende bildet, men heller indikere i kodingen at det er identisk.

    - Vi kan også bruke komprimering til å lage flere versjoner av den samme videoen, hver med forskjellige kvalitetsnivåer.
- Egenskaper ved lyd
    - Digital audio, inkludert tale og musikk, krever betydelig lavere båndbredde enn video. Hvordan blir analoge signaler konvertert til digitale?
        - Samples på en bestemt rate. Verdien av hver prøve vil være et tall
        - Hver av prøvene rundes av til et bestemt antall verdier (kvantifisering). Antall slike verdier er typisk en toerpotens, feks 256
        - Hver av kvantifiseringsverdiene er representert med et bestemt antall bits.
        - Denne metoden kalles pulse code modulation (PCM). Dersom et signal samples med en rate på 8000 samples per sekund og hver sample kvantifiseres og representeres med 8 bits vil signalet ha en rate på 64 000 bits/sekund.
    - En populær komprimeringsteknikk for nær CD-kvalitet stereomusikk er MPEG 1 layer 3, mer kjent som MP3. 128 kbps er den mest vanlige encoding-raten og innebærer lite lyd-degradering
    - Advanced Audio Coding (AAC) er også en kjent standard, utviklet av Apple
- Vi klassifiserer multimedia applikasjoner innen tre kategorier: streaming stored audio/video, voice/video-over-IP konversering og streaming live audio/video
- Streaming stored audio/video
    - Dette handler om forhåndsinnspilte filmer/tv-serier++ som plasseres på servere og bruker sender forespørsler for å se videoer on demand (youtube, netflix++). Streaming stored video/audio har tre forskjellige egenskaper
        - Strømming: Klienten begynner avspilling typisk rett etter videoen mottas fra tjeneren
        - Interaksjon: bruker kan pause og spole gjennom innholdet da det er forhåndsinnspilt.
        - Kontinuerlig avspilling: Når avspilling har startet skal den avspilles i samme hastighet som den ble recordet.
    - Viktigste målingen er gjennomsnittlig throughput
        - For å tilfredsstille kontinuerlig avspilling, må nettverket levere en average throughput til streaming applikasjonen som er minst like stor som bitraten til videoen selv
- Konverserende voice- og video-over-IP

    - Audio og video applikasjoner er svært delay-sensitive. For voice blir ikke forsinkelser mindre enn 150 millisekunder merket av mennesker. Forsinkelser mellom 150-400 ms er akseptable, mens over 400ms kan være frustrerende.
    - Konverserende multimedia er tapstolerante, noen glitcher kan bli akseptert
- Streaming live video/audio
    - Minner om tradisjonell kringkastningsradio og TV, bortsett fra at det finner sted på internett.
    - Gjøres med CDNs
    - Nettverket må tilby live multimedia flyt med høyere hastighet enn videokonsumeringsraten (live multimedia throughput > video consumption rate).

9.2 - Strømming av lagret video

- Kan klassifiseres i tre kategorier: UDP strømming, HTTP strømming og adaptiv HTTP strømming, der de to sistnevnte er mest vanlig i dag.
- Ved strømming kan man tillate litt forsinkelse før avspillingen begynner, mot at man bygger opp en reserve på noen sekunder av bufret, men ikke avspilt video.
    - Dette er klient buffering, fordelen med dette er:
        - Klienten kan fange opp variasjoner i server-to-client forsinkelse. Hvis data kommer forsinket, men likevel kommer før bufferet er tomt, vil ikke forsinkelsen merkes
        - Hvis båndbredden dropper dramatisk, kan man fortsatt spille av video siden det er data lagret i buffer som kan bli spilt av.
        - Se figur 9.1 (7th edition) for client-side buffering
- UDP strømming
    - Tjeneren overfører video med en rate som matcher klientens forbruk ved å klokke biter av video med en steady rate (forbruk på 2Mbps og pakker på 8000 bits vil innebære at en pakke overføres hvert 2Mbps/8000 = 4 ms). Før videobitene sendes til UDP vil tjeneren innkapsle bitene i transportpakker spesielt designet for video og audio ved bruk at av Real-Time Transport Protocol - RTP.
    - I tillegg til server-to-client video stream, klienten og tjeneren har også, i parallell, en separat control connection der klienten sender kommandoer angående session state endringer (pause, spole osv). RTP her også
    - Tre ulemper:
        - På grunn av uforutsigbare og varierende båndbredder mellom tjener og klient er det ikke sikkert konstant-rate UDP strømming

kan levere kontinuerlig avspilling (må levere høyere hastighet når mulig for å skaffe buffer-forsprang)
        - Krever en media kontrolltjener, som en RTSP tjener, til å prosessere klient-tjener interaksjonsforespørsler og overvåke klientens tilstand (hvor langt den er kommet, pausing/spoling osv) for hver pågående klient session.
        - Mange brannmurer er konfigurert til å blokkere UDP trafikk som hindrer brukerne bak disse brannmurene å motta UDP video.
- HTTP strømming
    - Video lagres på en HTTP tjener som en vanlig fil med en spesifikk URL. En TCP kobling benyttes
    - Overføringsraten ved TCP kan variere mye som følge av TCP sin opphopningskontroll, samt retransmisjon. Derfor var man skeptisk til strømming i forbindelse med HTTP. Men ved klient buffering og prefetching fungerer det bra.
    - En fordel er at video lettere passerer brannmurer som følge av TCP, og ikke UDP. Man trenger eller ikke media kontrolltjenester. Derfor bruker de fleste (Youtube og Netflix) HTTP strømming over TCP.
    - Prefetching video
        - For strømming av lagrede videoer kan klienten forsøke å laste ned video med en høyere rate enn forbruksraten og på den måten prefetche/forhåndshente videorammer som skal konsummeres i fremtiden. Disse lagres naturlig i klientens applikasjonsbuffer. Dette fungerer naturlig nok ikke med live videostrømming.
    - Klient applikasjonsbuffer og TCP buffere
        - Når klient appliksajonsbufferen blir full kan ikke send-raten (strømningsraten) være høyere enn forbruksraten. Det må frigis plass i bufferen. Ellers kan den være høyere og dermed skape et større forsprang på avspilling som er nyttig når strømningsraten faller under forbruksraten (er den høyere hele veien vil avspillingen aldri fryse).
        - Dersom forbruksraten er høyere vil bufferen gå tom og videoavspillingen etterhvert fryse. Dersom man ønsker å hoppe i videoen og har bufferforsprang vil dette gå tapt (dersom man hopper forbi forspranget, dvs $t > to + B/r$), og man vil ha sløst nettverksressurser.
        - En HTTP byte range header brukes for å indikere at man vil hoppe i videoen og starte avspillingen lenger ut i bytesekvensen.

9.3 - VOIP

- Voice-over-IP (eller Video-over-IP)
- Begrensninger ved Best-Effort IP tjenesten
    - IP er best-effort. Mangelen på garantier kan være et problem for real-time konversering som er sensitive til pakkeforsinkelser, jitter og pakketap
- Pakketap
    - Med UDP er sjansen for pakketap større enn ved TCP som bruker retransmisjon. VOIP bruker UDP fordi den ikke benytter retransmisjon som øker ende-til-ende forsinkelsen kraftig. Retransmisjon kan føre til at strømningsraten blir lavere enn forbruksraten som kan gjøre bufferet tomt. Pakketapsrater på 1 til 20 prosent kan tolereres i real-time applikasjoner, litt avhengig av hvordan lyd kodes og overføres.
    - Pakker med ende-til-ende forsinkelse over 400 ms (eller en terskelverdi i det området) går tapt.
- Pakkejitter
    - En viktig del av ende-til-ende forsinkelsen er den varierende køforsinkelsen noen pakker opplever hos ruterne. Dette gjør at ende-til-ende forsinkelsen kan variere, et fenomen som kalles packet jitter.
    - Ved å følge to mekanismer kan mottakeren periodisk spille av lydbiter i forekomsten av tilfeldig jitter. Disse mekanismene er:
        - Timestamp, senderen markerer hver bit med en tid når biten ble generert
        - Delaying playout, forsinke starten av avspillingen hos mottakeren slik at de fleste pakkene mottas før avspillingstiden deres.
    - Vi ser på to avspillingsteknikker som eliminerer jitter:
    - Fiksert avspillingsforsinkelse (fixed playout delay)
        - Mottakeren forsøker å spille av hver bit nøyaktig q ms etter at biten blir generert (hos senderen). Pakker som kommer etter det bestemte avspillingspunktet (som følge av jitter) forkastes. Jo lavere ende-til-ende forsinkelse og jitter, jo lavere q. q må være lavere enn 400 ms og kan også være lavere enn 150 ms.
    - Adaptiv avspillingsforsinkelse (adaptive playout delay)
        - Estimere nettverksforsinkelsen og variansen og tilpasse avspillingsforsinkelse tilsvarende etter begynnelsen av hver gang taleren snakker. Dermed vil talerens stilleperioder bli komprimert. En algoritme kan optimere dette og sørge for en god trade off mellom å miste pakker og å sette en for høy verdi på q (se nøyere på denne senere).

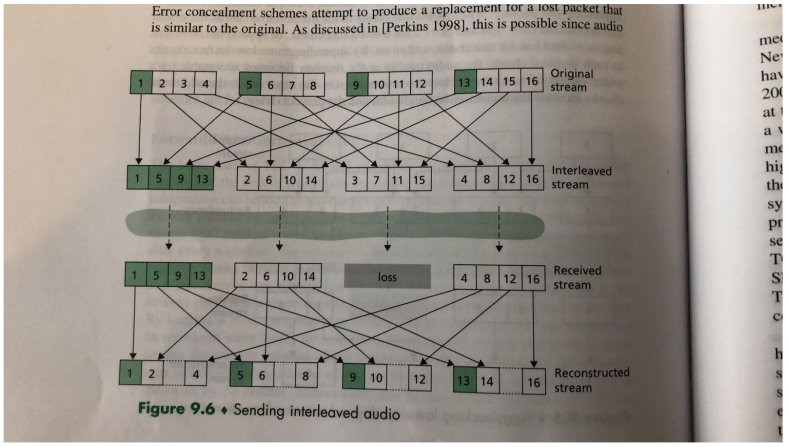
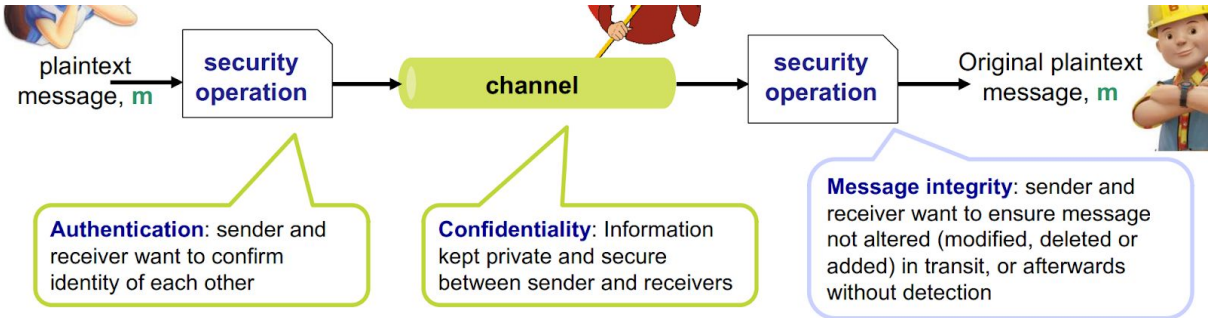
- Vi presenterer noen metoder som gjør at man kan beholde akseptabel avspillingskvalitet selv ved pakketap. VoIP bruker en form for tapsforventningsordning. To slike er:
    - Forward error correction (FEC)
        - Ideen går ut på å legge til overflødig informasjon til den originale pakkestrømmen. Kosten ved å marginalt øke overføringsraten (dårligere) kan utnyttes ved at den overflødige informasjonen benyttes til å rekonstruere approksimerte versjoner av den tapte pakken.
        - To FEC metoder
            - Enten legge ved n+1 pakke som kan erstatte en tapt pakke
            - Den ekstra pakken får man ved å XORe de originale dataene
            - Eller sende en lavere resolution audio stream som ekstra informasjon, som kan dekke for et pakketap.
    - Interleaving
        - Som et alternativ til å sende overflødige pakker, kan en VoIP applikasjon sende interleaved audio.
        - Man fordeler audio-enheter spredt fra hverandre og legges i "chunks". Hvis en chunk blir borte, interleaver man på nytt, og fordeler slik at hver chunk heller mangler én enhet, enn at en hel chunk er borte
        - Ulempen med dette er at det øker forsinkelse.
        - Fordel er at det ikke øker båndbreddekravene til en stream.
        - 

Figure 9.6 ♦ Sending interleaved audio
    - Error Concealment
        - Forsøker å produsere en erstatning for en tapt pakke som ligner originalen
        - Dette fungerer fordi audio, og særlig tale, har store deler korttids likhet.
        - Fungerer for lav tapsrate (mindre enn 15 %) og små pakker (4-40 ms).

        - Når tapslengden når lengden av fonemet/språklyd (5-100 ms) fungerer ikke teknikkene lenger
        - Enkleste formen for dette er pakkerepetisjon. Dette er at man erstatter tapte pakker med kopier av pakker som ble sendt rett før de tapte.
        - En annen teknikk er interpolation, som bruker audio før og etter tapet for å lage en passende pakke som dekker tapet. Fungerer bedre enn pakkerepetisjon, men høyere kompleksitet
- VoIP med Skype
    - Skype sender by default audio og video over UDP. Kontrollpakker sendes som TCP, og media pakker sendes som TCP der brannmurer blokker UDP.
    - Bruker FEC for loss recovery for både voice og video sendt over UDP.
    - Bruker P2P teknikker, blant annet til chat, brukerlokasjon og NAT gjennomløping.
    - Hvis to skype-ringere har NATs, kan de ikke ringe til hverandre. Dette løses ved P2P teknikk, ved å bruke super peer og releer. Super peer har ikke NAT, så hver person kobler seg til hver sin. Hvis de to (vanlige) peerene aksepterer anropet, velger super peerene en tredje ikke-NATed super peer - rele peeren - hvis jobb er så sende data mellom Bob og Alice.
    - For anrop med flere enn 2 brukere, må man motta en audio stream fra alle brukerne, til alle brukerne. Dette løses ved å sende all audio til hosten for anropet (den som initierte anropet), som deretter samler lyden til én audio og sender det ut til deltakerne igjen.
    - Når det er samme scenario med video, løses det annerledes: Hver deltaker i videoanropet sender sin videostream til en servercluster, som deretter releer til hver deltaker. Dette er da ikke P2P.
    - Sikkerhet
        - Når to personer ringer hverandre, kan man sniffe på hverandres IP og bruke dette til å finne lokasjon og ISP.
        - Man kan også se på IP adressen man får fra skype og sjekke det mot BitTorrent, og dermed se hva den andre laster ned.
- RTP - Real-Time Transport Protocol
    - Kan bli brukt til å transportere audio -og videoformater.
    - Er mellom UDP og applikasjonslaget
    - Sendersiden innkapsler en mediachunk i en RTP pakke, deretter innkapsler den i et UDP segment, og så sender videre til IP.
    - Mottakersiden gjør motsatt, pakker ut RTP pakken fra et UDP segment, og deretter henter ut mediapakken fra RTP pakken, og sender dette videre til mediaavspilleren for dekoding og rendering.

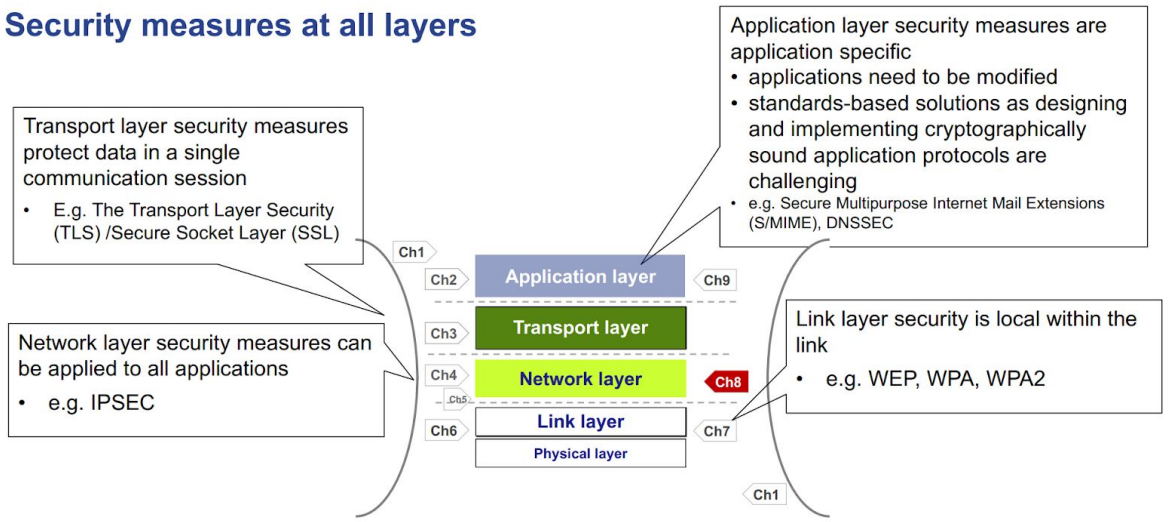
    - Har ingen mekanisme for timet levering av data eller andre QoS garantier, garanterer ikke engang levering av pakker eller unngår out-of-order levering av pakker.
    - RTP innkapsling skjer bare hos endesystemene. Rutere ser ikke forskjell på IP datagrammer som inneholder RTP pakker eller ikke
    - Har en egen stream for hver source som sender pakker. Videokonferanse mellom to personer har da fire RTP streams; to for video (begge veier) og to for lyd (begge veier).
        - Ved feks MPEG 1 og 2 blir lyd og bilde lagt inn i samme stream, og vi trenger bare en RTP stream i hver retning.
    - Er ikke bare unicast sendinger, også multicast
- RTP header
    - Sekvensnummer - 16 bits lang, øker med 1 for hver RTP pakke sendt, og brukes av mottakeren for å se etter pakketap og for å gjenopprette pakkesekvenser.
    - Timestamp field - 32 bits lang. Viser samplingsøyeblikket til den første byten i RTP pakken. Kan brukes til å fjerne pakkejitter. Timestamp kommer fra en samlingklokke hos senderen
    - Synchronization source identifier (SSRC) - 32 bits lang. Identifiserer kilden til RTP streamen. Hver stream har vanligvis ulik SSRC. Dette er ikke IP adressen til senderen, men et tall som senderen velger random når en ny stream har startet. Hvis to streams får lik SSRC (som er lite sannsynlig), velger de to en ny SSRC verdi.

Security - summary

- CIA triangle
    - Confidentiallity
    - Integrity
    - Availability
- I tillegg autentisering
- Network security model:
- 

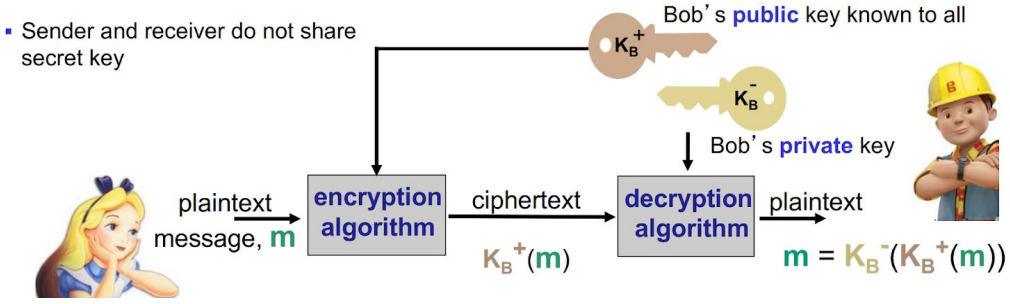
- Trusler
    - Virus, ormer, trojanske hester, spam, phishing, hijacking, lytting, DDoS

    - Første DDoS gikk på TCP
- Internet er ikke opprinnelig designet med tanke på sikkerhet
- Eavesdrop - sitter og lytter på nettet
- Man-in-the-middle - Sitter mellom de som kommuniserer
- Spoofe - falsk adresse og stjeler data
- Denial of Service - rettede angrep der man sender mange pakker til samme destinasjon
- Finnes sikkerthetshull i TCP/IP i både design og implementasjon
    - En intrenger kan enkelt lese HTTP klartekst datapakker som sendes mellom klient og server
    - DoS fører til at three-way-handshaking ikke gjennomføres og det kræsjer
- Kryptering er metode for å sikre konfidensialitet
- Autentisering - digitale signaturer
- Sikkerhet i alle lag



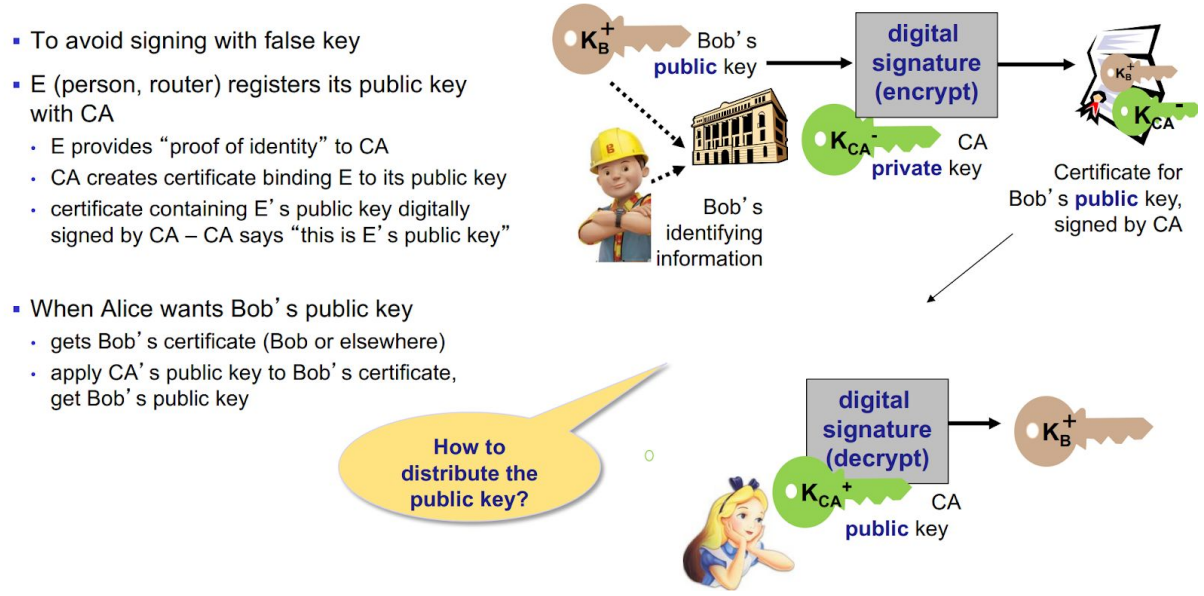
A security mechanism at a higher layer cannot provide protection for data at lower layers, because the lower layers perform functions of which the higher layers are not aware

-
- Applikasjonslaget - trenger sikker kommunikasjon, krypterer meldinger
  - Utfordring: nøkkeldistribusjon
  - Symmetriske og asymmetriske nøkler i kryptering
    - Asymmetrisk vanskelig/umulig å kryptere



  -
  - MAC - message authentication code
- Transportlageret - sikrer TCP-forbindelse
  - SSL/TLS handshaking
    - Starter med cipher bytte og server autentifikasjon
    - Number once - nonce
      - Brukes for tilfeldighet
- Nettverkslager - sikrer alt som går mellom to IP adresser
  -
- Linklageret -
  - WEP - Wired equivalent privacy, basert på symmerisk nøkkelkryptisering
  - Bruker nonce
  - CRC-32 - cyclic redundance check, sjekker etter feil bit i rammen

        - Utfordringen med dette er at det er en standard algoritme. Bryter dataintegriteten
    - WEP er utdatert, bruker nå WPA eller WPA2
    - Har kryptering på VPN-klient som feks eduroam og på https-sider
    - Port 80 overload - porten til TCP. De fleste brannmuren har port 80 åpen.
- Sikkerhet i lager under/over kan ikke håndteres i nåværende lag
- CA - certification authorities
    - For å unngå å bruke falsk nøkkel
    - Registrer sin offentlige nøkkel med CA, lager et sertifikat
    - Mottaker vil ha sender sin offentlige nøkkel, får sender sitt sertifikat, bruker CA offentlig nøkkel på sertifikatet, får offentlig nøkkel



Wireshark:
TCP går i klartekst, alt over er kryptert vha TLS.
Kvitteringsmeldinger - kvittering på at det er mottatt riktige pakker