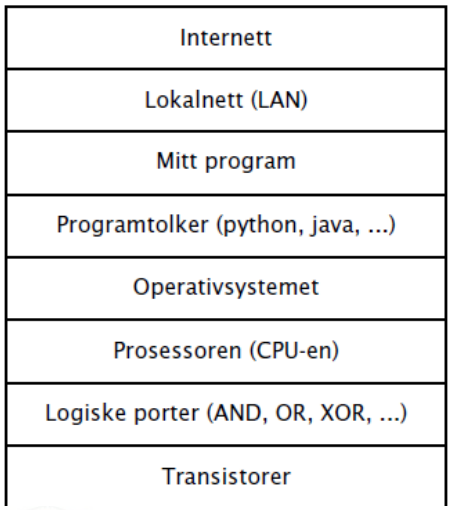
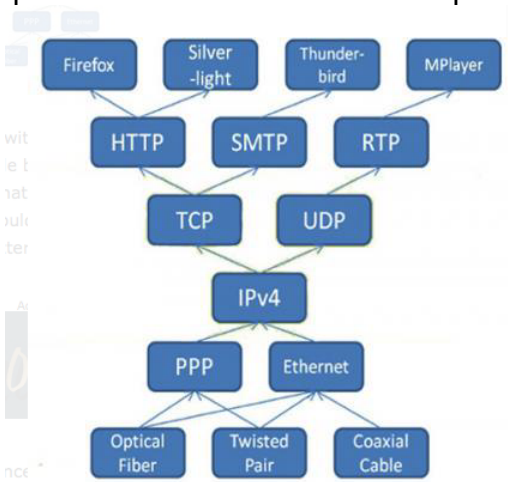
Cheat Sheet - IN1020

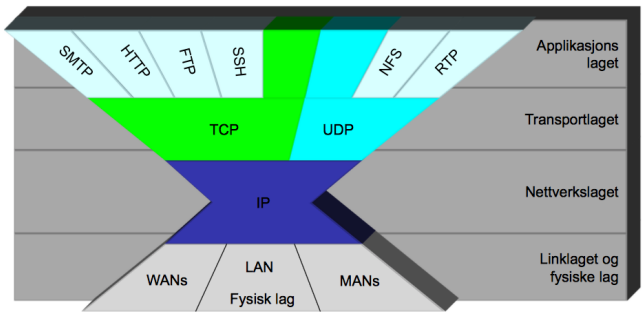


NETTVERK:

ARP: Address Resolution Protocol
CDIR: Classless Inter Domain Routing
CDN: Content Delivery Network
DHCP: Dynamic Host Configuration Protocol
DNS: Domain Name System
FIFO: first in first out
HTTP: Hypertext Transfer Protocol
ISO/OSI: Open System Interconnection
IP: Internet Protocol
ISP: Internet Service Provider
LAN: Local Area Network
MAC: Medium Access Control
MAN: Metropolitan Area Network
MSC: Messaging Service Center
NAT: Network Address Translation
NFC: Near Field Communication
PDU: Protocol data unit (beskjed)
Peer 2 peer
RTT: Round trip time
SAN: Storage Area Network
SAP: Service Access Point
SDU: Service Data Unit (nyttelasten i PDU)
SMTP: Simple Mail Transfer Protocol
SSL: Secure Socket Layer
TCP: Transmission Control Protocol
TCP/IP: det nye alternativet til ISO/OSI modellen
TLD: Top Level Domain
UDP: User datagram protocol
VPN: Virtual Private Network
WAN: Wide Area Network



Internet Protocol Stack



Kallenavn: "Timeglass-modellen"

Begreper:

- ***CIDR/classless inter domain routing***: ser til at internett har nok IP-adresser.
- ***Båndbredde*** er hvor mye data du kan flytte på en gitt mengde tid
- ***Hastighet*** er tiden det tar å gjennomføre et stadie av en oppgave.

REFERANSEMODELLER:

OSI-modellen:

- Modell for lagdelte kommunikasjonssystemer
- Inneholder 7 lag
- Brukes ikke så mye i dag på grunn av at det er for tregt å implementere

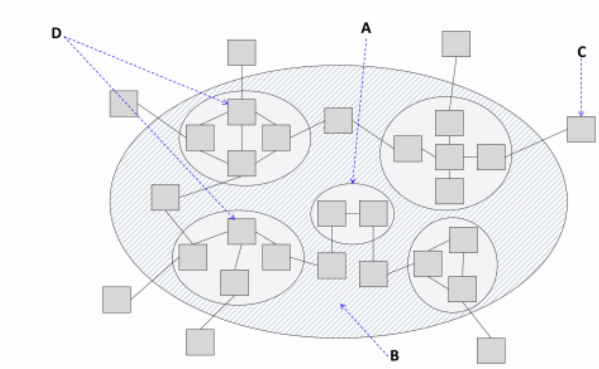
TCP/IP-modellen:

- Forenkling av OSI
  Applikasjon: der programmene i nettet kjører
  Transport: Kobler sammen ende-til-ende (TCP/UDP). Finner porten dataen skal sendes til (f.eks. nettleser, iMessage osv.)
  Nettverk: Rute data fra ende-til-ende systemer (IP) - vanskeligst å gjøre endringer. Finner riktig mottaker
  Link: Pålitelig overføring mellom to noder (MAC-adresse, Ethernet, LAN), at alt er sendt og riktig rekkefølge
  Fysisk: Sender bit ut på mediet (kablet eller trådløst)

LAGDELING I INTERNETTARKITEKTUREN
En protokoll definerer strukturen på beskjeder sendt over et nettverk. En protokoll må adressere mange kompleksiteter:
- Hvordan skal maskinvaren oppføre seg?
- Hvordan skal beskjeden finne frem?
- Er det noen garantier for levering?
- Hvordan håndtere kø, tap og andre problemer?

En protokoll handler om kommunikasjon mellom samme lag. Protokollen definerer formatet på beskjeden og en header.

Nettverkskomponenter:
- Endesystemer(C) finnes ytterst i nettverket (datamaskiner, mobiler, servere osv.)
- Intermediate systemer (D) finnes inne i et nettverk (ruter, switch, gateway osv.)
- Subnet (A), en gruppe av enheter innenfor et nettverk (B)



Nettverksstrukturer:
- Punkt-til-punkt: forskjellige typer kabler eller radiolinker som kommuniserer fra punkt til punkt (en-til-en)
- Topologi: f.eks. ring eller stjerne
- Broadcast-systemer: nettverk som deler kommunikasjonsmedium. Én sender, alle lytter (én-til-mange), eks. trådløst internett. En melding som sendes ut på en spesiell adresse og leveres til alle enheter på samme LAN. Problem: hvordan takle kø, alle skal ikke få "høre" alle sitt.

Protokoller og lag
- Utfordring: komplisert å kommunisere med fremmede maskiner på nettet og interaksjon mellom forskjellige system og/eller nettverk
- Forenkling: introduserte standardiserte abstraksjonsnivåer (lag)

Lag i nettverket (OSI)
- (N)-lag er et bestemt abstraksjonsnivå
- (N)-entity er oppgaven til et lag (typisk en prosess)
- (N)-Service Access Point er tjenesteidentifikasjon. Beskriver hvordan lag N tilbyr en tjeneste til lag N+1

- (N)-protokoll er et sett med regler for hvordan data skal overføres mellom entity på samme nivå

Dataenheter i OSI-modellen
- Data units har ulike navn ut i fra hvilket lag vi befinner oss i.
- Transportlaget: packets
- Nettverkslaget: datagram
- Linklaget: Frame

LAGENE
Applikasjonslaget: Lag med tjenester for applikasjoner, som nettleser, e-post ol.
Transportlaget: I transportlaget er det to viktige protokoller, TCP og UDP. TCP (Transmission control protocol) har et oppsett av forbindelse med 3-veis håndtrykk. UDP (User datagram protocol) er en tilkoblingsløs protokoll og har ingen garantier.
Nettverkslaget: Kobler sammen ende-til-ende systemer. Ansvarlig for ruting - hvordan pakken skal flyte gjennom nettet. IP (Internet protocol) er den mest brukte nettverkslag protokollen. En IP må være unik siden det er ende-til-ende. Dette gjør at vi begynner å få for få IP-adresser og vi går nå fra en 32 bit adresse (IPv4) til en 128-bit adresse (IPv6).
Linklaget: Har som oppgave å sikre pålitelig overføring mellom to enheter, passe på enkel flytkontroll og feildeteksjon. Hver endenode får en MAC-adresse (Medium Access Controll) på 6 byte som ofte er lagret i nettverkskortet. Dette gjør at ingen snakker i munnen på hverandre. Det vanligste linklaget er Ethernet og Wi-Fi.
Det fysiske laget: Består av signalrepresentasjon av bits. Sørger for at 1-bit blir mottatt som 1-bit. Dette gjennom kabler eller andre medium. Står for formelle regler om kommunikasjon.

Lagene spiller sammen
Internett er en sammenkobling av mindre, separate nettverk. På lokale nettverk kan man koble sammen enheter ved hjelp av en switch eller en HUB, dette fungerer på nettverkslaget. For å sende en pakke til en maskin utenfor ditt lokale nettverk må den sendes til en router som vet hvor en skal videresendes.

IP-adresse
IPv4 deler opp en IP-adresse i fire oktetter. For å definere et subnett må man ha en nettverksmaske. I nettverksmasken kan bits satt til 0 varieres for å angi IP-adresser i subnettet, mens bits satt til 1 angir delen av IP-adressen som definerer hvilket nettverk vertene tilhører. Nettverksmasken består av en sammenhengende serie “1” deretter en sammenhengende serie “0”. Det er to måter for å notere omfanget av et subnett. Ved punktnotasjon 192.168.1.0, da må man oppgi nettverksmaske. CIDR-notasjon (Classless Inter-Domain Routing) er eksempelvis 192.168.1.0/24 har vanlig punktnotasjon først og tallet etter skråstreken angir hvor mange bits nettverksmasken består av.

En IP-adresse - mange porter, hvordan fungerer dette?
- Transportprotokollene UDP og TCP implementerer ‘porter’ som muliggjør totalt 65535 (per UDP/TCP) samtidige forbindelser på en IP-adresse. Informasjon som sendes over nettet kan aksepteres av mottaker-maskinen ved å bruke TCP- eller UDP-porter. Når et program på en datamaskin sender/mottar data over nettet,

sendes denne data til en IP-adresse og en spesifikk port på den “eksterne maskinen”, og tar imot dataen på en (vanligvis random) port på lokal/mottakermaskinen.
- Ved bruk av TCP-protokollen tilkobles en TCP-port, og vice versa for UDP-protokollen.
- Når en applikasjon kobler seg til en spesifikk port vil den ikke kunne brukes av andre applikasjoner. “First come, first served”.

DHCP - Automatisk tildeling av IP-adresser
Prosedyre:
1. Det kommer en ny maskin på nettverket. Maskinen sier til alle (sender en kringkastingsadresse): finnes det en maskin (DHCP-server) med myndighet til å dele ut IP-adresser på dette subnettet?
2. DHCP-tjener (hjemmeruter) sier til alle: 192.168.1.5 er tilgjengelig. Her har du også en liste over andre viktige adresser som gateway og DNS.
3. Ny maskin: Tar i mot adressen.
4. DHCP-tjener: skriver opp at 192.168.1.5 er i bruk av den nye maskinen i en periode på 24 timer.
- Den nye maskinen må altså spørre igjen innen 24 timer om å få beholde IP-adressen hvis ikke blir den sett på som ledig.
- *Discovery* > *Offer* > *Request* > *Acknowledge*

ARP-koblingen mellom nettverk og IP
- Nettverkskortene har en 6 byte lang MAC-adresse som brukes til å identifisere maskinen innenfor et kringkastingsdomene. For at IP skal fungere må avsenderen vite hvilken MAC-adresse pakken skal sendes til. ARP kobler IP (nettverkslaget) og MAC (linklaget). ARP kan også brukes til å finne duplikate IP-adresser, dersom to maskiner svarer.
- ARP-forespørsel:
    - Sender til alle: Hvem har denne adressen?
    - Den som har den svarer
    - Den som spurte lagrer info i en liten tabell, dette kalles en ARP-tabell/ARP-cache
- Hvis kringkastingsdomenet er for stort når ARP sender ut en melding kan det stjele kapasitet som ellers burde vært brukt til å overføre data (DHCP har samme problem og avanserte switcher kan filtrere ARP og DHCP for å beskytte mot overbelastning fra kringkastet trafikk).
- ARP-tabellen lagrer informasjon om hvilke IP-adresser som assosieres med hvilke MAC-adresser. Når en pakke skal sendes til en IP-adresse sjekker systemet først om MAC-adressen finnes i tabellen; hvis ja er det ikke nødvendig og bruke ARP. Hvis nei sendes pakken til nettverket med ARP-protokollen som spør hvem som har den aktuelle adressen - maskinen med denne IP-adressen svarer med en ARP-pakke som også inneholder MAC-adressen som kan ta imot pakker for den IP’n.

Kringkastingsdomener
Om det er for mange enheter så kan det bli problemer. ARP og DHCP-forespørsler går til alle, dette vil for et stort domene stjele kapasitet som burde brukes til å overføre data.

Avanserte switcher kan fikse dette problemet. Burde unngå for store kringkastingsdomener, heller dele opp i subnett.

- Man kan skille mellom de ulike tjenestene man kan kjøre på en datamaskin ved bruk av porter. Notasjon IP : port

NAT - Network Address Translation

- NAT-routeren oversetter datatrafikk inn/ut av det private nettverket.
- Det finnes 2^{32} IP-adresser, og vi begynner å få for få. For å unngå problemet og fremme sikkerhet så kan vi bruke adressene flere ganger, dette gjennom NAT.
- NAT utnytter at transportlaget tilbyr porter som en funksjon. Routeren gir en maskin på det lokale nettet en bestemt port, og lagrer dette i en tabell som oversetter den lokale adressen til sin eksterne adresse. Når maskinen vil snakke med en annen maskin på internettet så vil den andre maskinen svare med IP + porten som er gitt for maskinen og routeren kan finne ut hvilken maskin som den skal sende svaret til.
- Dette innebærer at en unik IP-adresse er nok for flere datamaskiner.
- Ulemper med NAT: Gir en ekstra kompleksitet, og nye forbindelser må komme fra innsiden av NAT-nettverket.

Ruting i Internett

- Mål: videresende en datapakke slik at den til slutt når måladressen sin.
- ISP (Internet Service Provider) er internettleverandørene som gir deg tilgang til internett.
- ISP-peering er når ISPer inngår avtaler om å videresende hverandres trafikk. Dette fører til at økonomiske prinsipper påvirker trafikkflyt.
- BGP (Boarder gateway protocol) er protokollen som ISPene har for ruting mellom seg.

TCP

- **Metningskontroll**: Hvis en maskin merker at nettverket har alt for mye trafikk, så vil den trekke seg tilbake og bruke mindre trafikk. Det som muliggjør dette er at for hvert datasegment mottaker mottar så skal mottaker sende en bekreftelse på at det er mottatt. Så hvis pakke er gått tapt så vil hastigheten senkes og det vil da sendes pakkene som ble tapt på nytt.
- TCP har også levering i rekkefølge, dette gjør at man kan forsikre seg om at pakkene du sender kommer i riktig rekkefølge til mottaker.
- En mekanisme bygget for å forsøke å dele kapasiteten over flaskehalsen likt

Kryptering/sikkerhet i lagene

- Viktig å sikre informasjon som sendes i leddene på internett. Det er mulig å sikre informasjon i alle lagene.
    - VPN (IPSEC) kobler to subnett sammen så det fungerer som et LAN selv om de er fysisk adskilt (oppretter kryptert forbindelse og blir enige om en nøkkel som begge utveksler på en sikker måte)
    - TCPcrypt har som mål at alle TCP-forbindelser som settes opp skal være kryptert.
    - SSL (Secure sockets layer) er kryptering fra ende til ende

Mulige spørsmål/ting man bør kunne noe om:

En pakkes vei gjennom nettet

1. Du kobler maskinen til ditt lokale nettverk.
2. Linklaget mottar bits fra det fysiske laget og ser om det kan snakke med bitsene som kommer.
3. Om den kan det så vil den få en IP-adresse fra en tjener. Så kan du skrive inn et domenenavn i nettleseren din.
4. Da vil applikasjonslaget lage en HTTP-header. Dette er det som trengs for at maskinen din skal kunne snakke med den andre maskinen. HTTP er en vanlig applikasjonsprotokoll.
5. Denne protokollen inneholder en get-forespørsel som sier lever f.eks. forsiden til facebook til meg. Så vil den spørre nedover i laget.
6. Det sendes ut en ARP-forespørsel og man kan lage en linklag-header. Du har en datapakke, som legges på en http-header, så legges det på en TCP-header som sier at den jeg ønsker å kontakte på andre siden er på port 80, i tillegg til en del annen informasjon. Utenpå TCP-headeren legges det på IP-header som inneholder adressen til mottakeren.
7. Kjører DNS og får IP-adressen
8. Utenpå IP-headeren får du et linklag som er fylt ut av ARP og som har en MAC-adresse til ruteren din.
9. Dette legges ned i det fysiske laget som kjører ut en serie med bits som kan forstås i den andre enden.
10. Datapakken går fra maskinen til routeren din, der blir ethernet headeren tatt av og det settes på en ny som sendes til ISP.
11. Denne sender videre ut i fra informasjonen som er med.
12. Til slutt fjernes alle headerne og det er bare nyttelasten som leveres til web-tjeneren som leverer nettsiden.

Hva er poenget med lagdelingen?

- Lagdeling = standardiserte abstraksjonsnivåer.
- (Lagdeling av en datamaskin (fra transistorer til internett): er en forutsetning for å lage en datamaskin. Takket være lagdeling er det mulig å håndtere kompliserte konstruksjoner.)
- Lagdeling i internettarkitekturen: utfordringene utgjør at det kan være veldig komplisert å kommunisere med fremmede maskiner på nettet og interaksjon kan påvirkes av forskjellige typer system/nettverk (f eks PC/CPU vs MobilTlf/CPU), man må sørge for at maskinene snakker samme språk. Løsningen på dette er standardiserte abstraksjonsnivåer/moduler/lag. Data som skal sendes over nettet går altså gjennom flere lag og oversettes til et språk som mottakermaskinen til slutt kan forstå. Lagene kan byttes ut/endres for å tilpasse ulike oppgaver og språk/kommunikasjonsmåter (hvilket er enda en motivasjon for å bruke flere lag).
- Lagdelingen gjør det f.eks. mulig for nettverkskomponenter (som swithcher og routere) å implementere bare et delsett av lag, for å støtte bare det som trengs for å sende en pakke videre.

Kunne regne seg frem til en nettmaske og broadcast adresser.

- *Nettverksmaske*: er en 32-bit netmask/subnet mask som maskerer og separerer en IP-adresse inn i network-adressen og host-adressen (hver IP-adresse består av

disse to komponentene). Subnettmasken lages ved å sette alle network bits til “1” og alle host bits til “0”.

- Regne ut subnettet fra en IP + nettverksmaske: For å finne subnettadressen til maskinen må man gjøre en bitvis AND operasjon mellom IP-adressen og nettverksmasken.

Eks.
IP-adress til en maskin: 192.168.169.220
Binært: 11000000.10101000.10101001.11011100
Subnettmaske/CDIR: 255.255.240.0
Binært: 11111111.11111111.1111000000.00000000

Vi kjører en AND-operasjon:

11000000.10101000.10101001.11011100
11111111.11111111.1111000000.00000000
= 11000000.10101000.10100000.00000000
= 192.168.160.0

Svar: subnettet til maskinen over i CIDR-notasjonen blir altså 192.168.160.0/20. Tallet 20 på slutten her er CIDR verdien som er lik antallet positive bits i en 32 bit adresse (fra h til v) til en subnettmaske som i denne f eks. *11111111.11111111.1111000000.00000000.* Følgende adresse ville hatt en CIDR på 24: 11111111.11111111.11111111.00000000.

- For utregning av broadcast-adressen (kringkastings-adressen) til et subnett må man gjøre en bitvis OR-operasjon mellom maskinens IP-adresse og bit komplement (bitvis invers) av nettverksmasken.

Eks.
Vi bruker samme adresser som over.
Subnett-adressen vi regnet ut (binary network): 11000000.10101000.10100000.00000000
Subnettmaske/CIDR Invertert: 00000000.00000000.00001111.11111111

Vi gjør en OR-operasjon:

11000000.10101000.10100000.00000000/11000000.10101000.10101001.11011100??
00000000.00000000.00001111.11111111
= 11000000.10101000.10101111.11111111
= 192.168.175.255

Svar: Broadcast/kringkastingsadressen til IP-adressen i CIDR-notasjon er 192.168.175.255/20.

Hvordan fungerer DNS/Domain Name System?

- DNS synker domenenavn med IP-adresser hvilket gir brukere domenenavn/adresser som er enkle å huske, mens maskinene på nettet bruker IP-adressene.
- DNS identifiserer og lokaliserer datasystem og andre ressurser på nettet; når man skriver inn en webadresse/URL matcher DNS denne adresse med tilsvarende IP-adresse og kobler en til den nettsiden.

- DNS er en database som forvarer alle domenenavn og tilsvarende IP-adresser for en spesifikk top-level domain (TLD), f eks .com, .net etc.
- DNS databaser er lagret på fysiske servere i en distribuert database, dvs. mange maskiner på en måte som gjør at man kan få tak i den informasjonen man trenger hvor enn man befinner i verden.
- Hierarkisk navnetilordning:
  .com > google.com > mail.google.com
- Ulike maskiner har ansvar for forskjellige deler av navneoppslagene.
- DNS har en enkel klient/tjener arkitektur (den mest brukte tilkoblingsmåten på nettet), man setter opp en maskin/tjener som oppretter en tjeneste/lager en port f eks TCP. Alle maskiner vet da at hvis de kobler seg til akkurat den porten - kobles de til en DNS-tjeneste. UDP (vanligst frem til nå, brukes fortsatt på de fleste klientmaskiner) eller TCP port 53, fra nylig må tjener bruke TCP. Dette grunnes overgang til internett protokoll 6 hvilket medfører veldig lange IP-adresser, som ikke får plass i en DNS-pakke. Informasjonen som da skal sendes med UDP, må deles opp i to pakker hvilket medfører større risiko for å miste informasjon på veien. Klienter som bruker TCP avvises ofte.
- Klassisk metode/rekursivt oppslag: forespørselen går igjennom mange maskiner før den returneres til maskinen som først spurte. Dataflyten konsentreres rundt de sentrale tjenerne der også mange tilstander lagres. En del ressurser blir dårlig brukt.
- Ny metode/iterert oppslag: man delegerer til den første gyldige DNS-tjeneren i stedet for å gå igjennom mange maskiner på veien. Tilstanden lagres bare av den lokale tjeneren inntil svaret er levert.

Navnehierarki og rottjenere.

- Rottjenere (et set med maskiner) har lister på TLDs samt adresse til hvem det er som er ansvarlig for disse domenene. Siste leddet i en nettadresse. F eks .no, .com
- Disse TLD' forgrener seg nedover, f eks. .no > uio.no > ifi.uio.no > www.ifi.uio.no
- Rottjenerne administreres av et foretak, ICANN (frem til nylig eid/administrert i USA, er nå mer globalisert). I Norge er det UNINETT-NOR, disse godkjenner ansøkninger om bruk av nye domenenavn. Deretter er UIO administrator.
- Hver DNS-tjener har autoritet over en del av hierarkiet, de skal ha en liste over alle som sorteres under sitt sub-tre. Må kunne replikeres, derfor er det minst to tjenere på alle nivåer. Alle må kjenne til adressene til rottjenerne.
- Rottjenerne er ansvarlige for "Root Zone File"/liste over TLDer og hvem som kontrollerer dem. Det finnes 13 rottjenere hver av 6 er replikert globalt med en teknikk som heter "anycast".
- Rottjeneren kontaktes når man mislykkes med navneoppslag. I praksis mellomlagres denne informasjon i cache på de fleste system.
- DDoS-angrep: denial of service-angrep, man prøver å sende så mange forespørsler at serveren går ned (har vært nær, men ikke skjedd).
- Rottjenerne ligger fremst i USA.

TJENESTER I INTERNETT

Forbindelsesstrategier:

- **Pull**: klienten ber tjeneren om en tjeneste, og er den vanligste metoden
- **Push**: tjeneren dytter en tjeneste til en klient, dette krever at det er en forbindelse fra før eller at klienten lytter (Eks. push varsler).

- **Publish-subscribe**: variant av push der tjeneren dytter ut beskjeder til en gruppe av abonnenter.

Aksessmodeller:

Klient-tjener

- Tradisjonell kommunikasjonsmodell: Klient ber om en tjeneste (oppretter en forbindelse), tjenere leverer tjenesten (svarer på forespørselen)
- Eks. Webklient (nettleser), mail

Peer-to-peer (P2P)

- Første velkjente programmet: **Napster**
- Fildeling (brukt til f.eks. musikk og film. Eks: Popcorn time)
- Hvis en maskin har tilgang til en fil blir dette delt i et felles nettverk
- Alle noder er likeverdige
- Alle noder kan nå hverandre
- Eierskapet er distribuert

Fysisk plassering av innholdet

Man kan utvide cachehierarkiet for å gjelde også utenfor datamaskinen. Da kan man mellomlagre data på ulike avstander fra brukeren også på internett.

- Eksempelvis ha en lokal proxy på et lokalt internett, hvor den har en cache som kan levere data med kort responstid til brukerne av det lokale nettet.
- Internettleverandører har ofte ting lagret hos den for store bedrifter som trenger data i store deler av verden.
- Man kan også gå til den originale datakilden selv om dette tar mye lengre tid

CDN - Content Delivery Network: kopier av data som man ønsker å dele/levere til folk som er nærme der brukeren er. For eksempel hos internettleverandøren eller i sky-tjenester. Dette gir kortere RTT-tid og forhindrer også overbelastning på tjeneren. Minus: koster ekstra maskinvare og lagringsplass.

Proxy-cache: er en forward-proxy som står nære klientene og vil da levere til klientene. En “Reverse proxy” står nær tjenerne og mellomlagrer data fra en eller flere tjenere slik at klienten slipper å gå helt til kilden. Fungerer som en front for klienten. Dette bruker mye på Wi-Fi på fly.

Head of line blocking

- Mottaker må vente på et forsinket segment før mer data kan leveres
- Tapte pakker må sendes på nytt
- Dette fører til en forsinkelse

Videostreaming - utfordringer:

- Konstant strøm av data (til avspilling slutter)
- Nedlastning: hele innholdet lastes ned til en lokal maskin
- Streaming: Ikke lokal lagring, sparer nettverksressurser
- UDP eller TCP?
    - UDP kan fungere helt fint, fungerer raskt og merker det ikke om det mangler noen pakker.

        - Mer TCP i det siste
- Nettverksutfordringer:
    - Forsinkelser, tap, variasjoner i leveringstid (“jitter”)
    - jitter kompensasjon
    - tapskompensasjon

HTTP-streaming
Dynamisk, adaptiv streaming over HTTP (DASH)

- Bruker TCP som transportprotokoll. Måten man gjør dette på er ved å dele videoen opp i biter på cirka 2-10 sek. Hver av de bitene er en film som kan spilles av helt uavhengig. Det finnes også flere versjoner av hver bit, en med god kvalitet, en med middels kvalitet og en med dårlig kvalitet. Måten man sender dataen på er ved at videospilleren fungerer som en nettleser. Den spør serveren om den kan levere et segment og serveren sender med lavest kvalitet og mottaker mottar. Dersom dette går fint, ingen pakker blir mistet ol. så sender serveren gradvis med bedre og bedre kvalitet, helt til optimal kvalitet er oppnådd.

HTTP-protokollen (hypertext transport protocol)

- Er en applikasjonlagsprotokoll, som inneholder meldingskoder og metadata som trengs for å overføre websider oppå det som er av TCP-headere og IP-headere.
- Har en klient-tjener modell
- Bruker TCP som transport
- HTTP har ingen tilstandsinformasjon, dette er en fordel for protokoller som trenger å være effektive

Persistente og ikke-persistente forbindelser:
Ikke-persistent:

- Tjeneren leser forespørselen, svarer, lukker TCP-forbindelsen
- Hver overføring lider av TCP sin gradvise økning i senderate (slow start)
- Mange nettlesere åpner flere parallelle forbindelser for å prøve å ta mer kapasitet og få ting til å fortere, omgå et designproblem

Persistent:

- Standard for HTTP/1.1
- Over samme TCP-forbindelse: tjeneren leser forespørselen, svarer, leser ny forespørsel
- Klienten sender forespørsler for alle de objektene den trenger så fort den mottar hoveddokumentet (HTML)
- Færre RTTer, mindre slow start
- Persistent med pipelining: spør etter mange objekter på én gang (enda færre RTTer)

Cookies:

- Tar vare på tilstand
- Lager spesialtilpasset innhold for klienten, basert på lagrede HTTP-forespørsler
- Vise spesialtilpasset reklame

E-post

- Simple Mail Transfer Protocol (SMTP)

- Bruker TCP (alt må komme frem)
- 3 faser:
    - Håndtrykk
    - Overføring av beskjeder
    - Avslutning
- Ascii 7-bit

INFORMASJONSSIKKERHET:

Begreper:

- **Sikkerhet**: handler om å beskytte verdier mot skade og ødeleggelse
- **Verdi (vid betydning)**: helse, miljø, informasjon, sensitiv informasjon, penger, gjenstander, omdømme osv.
- **Skadelige hendelser (tilsiktet eller utilsiktet)**: naturkatastrofer, krig, ulykker, skade påført med hensikt osv.
- **Informasjonssikkerhet**: sikre informasjon, beskytte informasjonsressurser (IT-utstyr, infrastruktur, datafiler, programvare etc.).
- **Cybersikkerhet**: sikre ting som er sårbare via IKT (f.eks. informasjon, selvkjørende biler, pacemaker osv.).
- **Personvern** handler om retten til et privatliv og retten til å bestemme over egne personopplysninger.
- **Innebygget personvern** tar hensyn til personvern i alle utviklingsfaser av et system eller løsning. Setter en standard for hva som skal ligge til grunn for et system eller program.
- **Sikkerhetsmål**: uavhengig av spesifikk implementering og kan ivaretas med ulike sikkerhetstiltak.
- **Sikkerhetstiltak**: er tiltak for å oppdage, forebygge eller gjenopprette. De er basert på spesifikk implementering, ofte bundet til produkter.
- **DoS-angrep (Denial of Service-angrep):** hindrer tilgang til noe man vil ha tilgang til.
- **Sikkerhetsstyring** er systematiske aktiviteter for å oppnå og opprettholde et sikkerhetsnivå i overensstemmelse med de mål og krav en organisasjon har satt seg.
- **Risiko** = sannsynlighet for angrep * kostnad eller tap i kroner.
- **Trussel**: uønsket hendelse som kan inntreffe.
- **Angrep**: noen forårsaker med hensikt en uønsket hendelse.
- Informasjonssikkerhet er en kontinuerlig prosess for å oppdage og hindre trusler og å fjerne sårbarheter.

Sikkerhetsmål:

- **CIA (KIT): Confidentiality, Integrity, Availability**
    - **Konfidensialitet**: sikre at kun de med rettmessige behov har tilgang til en ressurs. Tiltak: kryptering, tilgangskontroll, perimetersikring, gode policy for hvordan data skal håndteres og hvordan kommunikasjon skal foregå. Trussel: tyveri, lekkasje
    - **Integritet**: sikre at en ressurs ikke endres eller slettes utilsiktet. Tiltak: tilgangskontroll, endringskontroll, kryptografisk sjekksumalgoritme, perimetersikring (f.eks. brannmur), digital signering også av programvare (code signing). Trussel: korrupsjon av data og systemer

    - **Tilgjengelighet**: sikre at en ressurs er tilgjengelig for rett person til rett tid. Tiltak: sikkerhetskopier, redundante tjenester, gode rutiner for hendelseshåndtering og gjenoppretting. Trussel: (D)DoS-angrep, systemfeil er også en trussel, og forekommer nok oftere enn DDoS-angrep.
- **Autentisering**: Skape tillit til ekteheten av en oppgitt identitet. Identifikasjon (hvem er du), autentifikasjon (legitimere at du er den du sier du er).
    - Autentifikasjon kan skje gjennom noe du har (kodebrikke), noe du kjenner (passord, pin) eller noe du er (biometri - fingeravtrykk, ansiktsgjenkjenning)
        - Utfordringer: lett å glemme, blir kanskje lagret i minne eller cache på maskinen. God passordhygiene er viktig.
        - Kan være behov for kombinasjon av disse slik som bankinnlogging
    - Kryptografiske autentiseringsprotokoller (f.eks. systemsertifikater som verifiseres av tredjepart) som TLS, VPN og IPSec (BankID) .
    - Kryptografiske mekanismer, PKI, digital signatur, elektronisk signatur
- **Uavviselighet**: sikre at mottaker eller avsender ikke kan bestride å ha sendt eller mottatt en melding. Bevis for at en melding er mottatt. Digitale signaturer kan løse dette.
- **Sporbarhet**: sikre at en gitt hendelse kan spores til en gitt identitet.
    - Logging av alle hendelser
    - Identifisere alle identiteter
    - Perimeterforsvar, ikke gi uvedkommende tilgang

Sikkerhetstiltak:
- Opplæring
- Kryptering (asymmetrisk, symmetrisk, sjekksum, kryptering av trafikk)
    - Sikre trygg lagring av data i utrygge lagringsenheter, sikre trygg overføring av data i f.eks åpne nett.
- Perimeterforsvar
- Tilgangskontroll
- Sikkerhetsoppdateringer
- Sikkerhetskopiering
- Gjenoppretting
- Redundans av tjenester

Sikkerhetstiltak i ulike datatilstander:
- Lagring
- Overføring
- Bruk

Sikkerhetstiltak i ulike faser:
- Preventive (forebyggende) tiltak, f.eks. kryptering
- Detekterende tiltak, varsle angrep som forsøkes eller skjedd. Eksempel: inntrengingsdeteksjon(IDS)
- Korrigerende tiltak: gjenopprette skade på dataressurser etter angrep. Eksempel: hente sikkerhetskopi av programmer og data

Ulike typer skadevare:
- Virus: spres gjennom åpning av fil av bruker eller systemet

- Ormer: sprer seg og finner nye ofre på egenhånd
- Trojaner: skadevare som utgir seg for å være et nyttig program
- Bakdør: en ubeskyttet og ukjent åpning inn i et system
- Spionvare: programvare for å spionere på en bruker for å få tak i info av nytte
- Logisk bombe: skadevare som kjøres ved et gitt tidspunkt eller ved gitt handling eller hendelse
- Tastelogger/keylogger: enhet/programvare som er koblet til tastatur eller maskin som lagrer alle tastetrykk
- Rootkit: skadevare som endrer funksjoner i OS eller applikasjoner, kan ta kontroll over maskin/system

Kryptering:
Sikrer konfidensialitet
Strategier:
- Filkryptering
- Kryptering av filsystem
- Fulldiskkryptering

Krypteringsalgoritmer:
- Symmetrisk: en nøkkel som både avsender og mottaker må kjenne til.
- Asymmetrisk: en offentlig og en privat nøkkel.
- Sjekksumalgoritmer (hash-funksjoner).

Krypteringsnøkler:
- Styrken i kryptografisk sikkerhet avhenger av: størrelse/lengde på nøkkelen, styrken i den kryptografiske algoritmen og håndtering/beskyttelse av nøklene.
- Én nøkkel har ett formål: f eks. kryptering, autentisering, generering av digital signatur.
- Nøklers klassifikasjon: offentlige, private, symmetriske og hva de skal brukes til.
- Utfordring: hvordan offentlige nøkler kan distribueres på en sikker måte.

Public Key Infrastructure (PKI):
- Sikre autentiske offentlige nøkler. Binder en offentlig nøkkel til en navngitt enhet, og bindingen kan bekreftes av en betrodd “myndighet” som utsteder et sertifikat. Et sertifikat forteller følgende: "Offentlig nøkkel K eies av X".
- Rammeverk for kryptering/autentisering og signering av dokumenter og programvare
- Asymmetrisk kryptering, med en offentlig og en privat nøkkel
- Eks: signere lån fra lånekassen: bruker Bank ID
- Hensikt med digitale sertifikater: A˚ knytte sammen en offentlig nøkkel og et individ/identitet.

Nettverkssikkerhet:
- Tradisjonell sikkerhetsmodell: stoler på endesystemet, nettverket er det som er upålitelig om noe skjer
- **Tradisjonelle trusler:** avlytting (passivt), forfalskning (aktivt), modifisering av data(aktivt)
- **To hovedområder:** kommunikasjonssikkerhet og perimeterforsvar
- **Kommunikasjonssikkerhet:** Skal beskytte data i transporten mellom virksomheter. TLS (Transport layer security) passer på sikkerhet i transportlaget og er basert på PKI (Public key infrastructure). IP security står for sikkerhet i nettverkslaget som baserer seg på kryptering, autentisering og nøkkelhåndtering.

- **Perimeterforsvar:** Sikkerhet ved hjelp av brannmur og innbruddsdeteksjon.
    - Brannmur - førstelinjeforsvar: Avgjør hva som slippes inn og ut fra et lokalt nettverk. Fungerer som en slags tilgangskontroll i nettverket og avverger uautorisert tilgang til/fra et privat nettverk. Implementeres i programvare eller maskinvare.
    - Innbruddsdeteksjon (IDS): System som detekterer mistenkelig aktivitet gjennom nettverksanalyse. Systemet kan detektere misbruk, skadevare og (D)DoS-angrep.

Sikkerhetstrusler
- Skadevare
- Sårbarheter knyttet til
    - Sosial manipulering
    - Nettverkssikkerhet
    - Applikasjonssikkerhet
    - Sikkerhet i datamaskinen
    - Lagring og avhending av informasjon
    - Bruk av skytjenester

Personvern(**loven**, vit at den finnes)
- GDPR er en stor innstramming i personvernsregelverket i hele EU/EØS og andre land som opererer innenfor EU/EØS som gjelder fra mai 2018.

DNS-sikkerhet:
- DNS-forfalskning: uvedkommende introduserer uriktige opplysninger i navnetjener
- DNS-modifisering: Man in the middle-angrep, forfalsker meldinger
- DNS-kapring: endre hvilke navnetjener offerets maskin benytter (hacke maskin)

Tilgangskontroll:
- Et subjekt vil utføre handling på et objekt, for eksempel endre et dokument.
- Tilgangskontroll = autentisering + autorisasjon.
- Autorisasjon = sjekke at subjektet har tilgang til å utføre den ønskede handlingen på objektet etter forhåndsdefinerte regler.

Referansemonitor:
- Oppgave: kontrollere tilgang fra subjekt til objekt
- Avviser eller tillater tilgang
- Følger policy som er satt
- Logger hendelser i hendelseslogg

Tilgangskontroll i UNIX:
- Brukere (UID) og grupper (GID) er grunnleggende elementer, begge er permanente identiteter.
- Brukere har brukerkonto og tilhører en eller flere grupper, en av gruppene er primærgruppe
- Subjekter i UNIX: en prosess

- Objekter i UNIX: filer (programmer), mapper og devicer (maskinvareenheter, representert som filer)

Tilgangskontroll på flere nivåer
- Brukere og passord kan kontrolleres i applikasjonslaget
- Tilgang til ulike nettverksporter kan kontrolleres i transportlaget
- Tilgang basert på IP-adresser kan kontrolleres i nettverkslaget

Sikkerhet i operativsystemet:
Kjernen i operativsystemet styrer alt. Kjernen kommuniserer med periferenhetene i datamaskinen gjennom drivere. Brukerprosessene må kommunisere via kjernen for å kunne benytte periferenhetene. Minnet i en datamaskin benyttes til mellomlagring.
- Kjerner og drivere er dataprogrammer som kan inneholder sårbarheter. Kjernen har tilgang til alt (supervisor mode) og blir dermed en sårbarhet som er svært kritisk
- Viktig å holde operativsystem og drivere oppdatert for å unngå at sårbarheter skal utnyttes.

Lagring av data
Data kategoriseres etter hvilket beskyttelsesbehov det har og hvilke tiltak som iverksettes avhenger av beskyttelsesbehovet. Tre ulike behov:
- Åpen informasjon
- Informasjon med begrenset beskyttelsesbehov
- Informasjon med sterkt beskyttelsesbehov

Sikkerhetskopiering: Sikkerhetstiltak hvor man kan gjenopprette data ved uønsket hendelse som sletter/endre data. Husk: også sikkerhetskopien må sikres.

Applikasjonssikkerhet
- Applikasjoner må designes og utvikles med tanke på å kunne brukes trygt
    - **I designfasen**: Trusselmodellering. Hvilke trusler ser vi? Hvilke tiltak kan iverksettes for å motvirke truslene?
    - **I programmeringsfasen:** Unngå å skape/tilrettelegge sårbarheter

Buffer overflow
- En sammenhengende seksjon i minnet i datamaskinen, allokert til å inneholde alt fra en streng til en array med heltall, kalles en buffer.
- En sårbarhet i et program som gjør at programmet overskrider bufferets grenser i minnet kalles buffer overflow.

SQL injection
- Spørrespråk mot relasjonsdatabaser. Brukes ofte i bakkant av web-applikasjoner for lagring av data.
- SQL-injection: Når det lures inn spørresetninger sammen med informasjon som er gitt som input i et grensesnitt. Input fra bruker valideres ikke før den sendes videre som spørresetning til databasen.

Innebygget sikkerhet oppnås når informasjonssikkerhet er:

1. innarbeidet i virksomhetsstyringen og understøtter virksomhetens mål
2. innarbeidet i virksomhetens prosesser og prosjekter fra starten av
3. tatt hensyn til i hele livssyklusen til IKT-løsninger
4. et tema alle ansatte i offentlig sektor kjenner til og vet hva innebærer for sine arbeidsoppgaver

Skytjenester

- **Fordelene** med skytjenester er blant annet redusert infrastrukturkostnader, besparelser i form av betaling for faktisk bruk og lokasjonsuavhengig tilgang.
- **Ulemper** er at dersom en virksomhet ønsker å benytte skytjenester til behandling av personopplysninger, er virksomheten juridisk ansvarlig for at personopplysningene prosesseres og lagres i samsvar med personvernregelverket.

Trusselmodellering

Flere innfallsvinkler:

- Hva er verdifullt i systemet, hvordan kan det gå tapt?
- Hva er motivasjonen for et angrep?
- Hva har gått galt tidligere?

Husk: det å beskytte ressurser skaper verdi ;)

TSD - Tjenester for sensitive data

Sensitive data: Helsedata, data om etnisitet og rase, politiske meninger, religiøse eller filosofiske standpunkt, fagforeningsmedlemskap, sexliv

Virtuelle maskiner: betyr at man har en fysisk enhet i en laptop eller server også i stedet for å si at en server tilsvarer en maskin så tilsvarer den mange maskiner. Man simulerer altså at man har flere fysiske maskiner.

DATAREPRESENTASJON OG ASSEMBLERKODE

DIGITAL REPRESENTASJON:

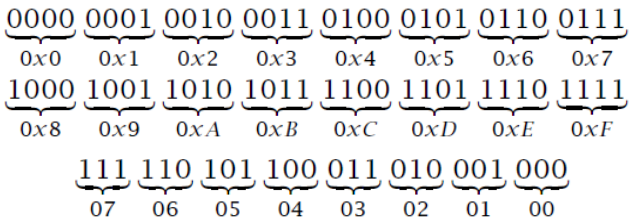
Ord og begreper:

- Binærtall: tall med grunntall 2
    - Notasjon: 1001_2
- Oktaltall: tall med grunntall 8
- Desimaltall: tall med grunntall 10
- Hextall: tall med grunntall 16 (og sifre 0-9 og A-F)
- Bit: verdi 0 eller 1
- Fortegnsbit: øverste bit som angir + eller -
- Byte: samling av 8 bit
- ASCII: 7-bits tegnsett (ett tegn per byte)
- ISO Latin-1: 8-bits tegnsett (ett tegn per byte)
- Unicode: universelt tegnsett (4 byte per tegn)
- UTF-8: lagring av Unicode med 1-4 byte per tegn
- PNG: bildeformat for rasterbilder

- JPEG: bildeformat for fotografier
- MP2: lydformat (gammel DAB)
- MP3: lydformat (MP3-spiller, iPod)
- AAC: lydformat (DAB+)

Heksadesimal notasjon
000100000010 = 0x102

Oktal notasjon
100000010 = 402



Generell omregning mellom desimaltall og andre baser:

Et tall: 1 1 1 1 1
Posisjon: 4 3 2 1 0
Utregning: 1 x grunntall^4 + 1 x grunntall^3 + 1 x grunntall^2 + 1 x grunntall^1 + 1 x grunntall^0
Eks:
Base 4: 3 0 2 1 3
Posisjon: 4 3 2 1 0
Utregning: $3 \times 4^4 + 0 \times 4^3 + 2 \times 4^2 + 1 \times 4^1 + 3 \times 4^0$
$3 \times 256 + 0 \times 64 + 2 \times 16 + 1 \times 4 + 3 \times 1$
$768 + 0 + 32 + 4 + 3 = 807$
$30213_4 = 807_{10}$

Tekst:

- ASCII
  - Enkelt å sjekke om det er et siffer eller en bokstav
  - Enkelt å konvertere fra liten bokstav til stor bokstav, ligger akkurat 32 bit unna hverandre
  - Trenger 7 bit for å lagre alle verdiene, under 128 ulike verdier
  - Inneholder kontrolltegn/styretegn, brukt til å jobbe med fjernskrivere
  - MINUS: mangler en del bokstaver og tegn, blant annet æøå
- ISO 8859-1
  - Halvparten av tabellen (128 verdier) er ASCII
- UNICODE
  - Tegnkoding som omfatter alle skriftspråk i verden
  - Plass til drøyt 1 000 000 tegn
    - Foreløpig er 136 755 tegn definert
  - Mer og mer programvare (som Java og Python) støtter UNICODE
  - 1 000 000 tegn krever 24 bit = 4 byte på alle tegn
  - UTF-8 er en måte å skrive UNICODE på som er komprimert (UNICODE er alle tegnene)
    - Bruker fra 1 til 4 byte på å lagre et tegn
    - Trenger 2 byte til æ, ø, å
    - Kommer an på tegnet hvor mange byte

Bilder:

- Hvert bildepunkt(piksel) består av tre farger: rød, grønn, blå som kan lyse sterkt eller svakt
- Hvitt -> 1
- Svart -> 0
- (Utskrift på papir: CMYK = Cyan, Magenta, Yellow, blacK)
- Ofte en byte til hver farge, dvs. 3 byte per piksel
    - 135 206 255
    - 12 x 9 piksler = 324 byte
    - 348 x 257 piksler = 363 682 byte
- Kvalitet kontra plass
- Redusere plass:
    - Lage en fargetabell for å se hva fargene inneholder, trenger kanskje ikke 3 byte per piksel
    - Run lenght-koding
        - Lagrer fargen og hvor mange slike piksler det er på rad
        - PNG og GIF bruker denne teknikken
- Fotografi litt annerledes:
    - Ikke brå overganger
    - Ekte rasterbilde tar stor plass
    - JPEG komprimerer bildet slik at det tar mindre plass, men taper informasjon(fjerner unødvendig informasjon)
        - Mennesker kan bare skjelne et begrenset antall nyanser
            - Ser hovedsakelig på overganger på himmel og hvordan huden ser ut
        - Kan ikke få tilbake til det opprinnelige, info er tapt
- Vektorgrafikk:
    - Lagrer bilde som geometriske elementer (rette linjer, buet linje)
        - Koder i det grafiske språket, matematisk definert
        - Kan derfor forstørre så mye man vil uten å tape info (geometri fungerer i alle størrelser)
        - Bruke vektorgrafikk der man kan (SVG, ESP, PDF)

Lyd:

- Lyd er trykkbølger i luft som f.eks. går inn i en mikrofon som sender elektrisk strøm som gjenspeiler lyden. En høyttaler mottar dette og kan gjenskape lyden gjennom membranen i høyttaleren
- Måler strømmen med jevne tidsperioder, flerfoldige tusen ganger i sekundet
- Skjer digitalt, må måle noe som er i nærheten av de virkelige bølgene
    - Målingen er bra, men den gjenskapte lyden vil ikke være helt nøyaktig
- Kvalitet avhenger av:
    - Hvor ofte vi måler
    - Hvor mange trinn vi benytter til målingen (hvor tett rastermønster vi har over målingene)
- CD:
    - 74 minutter spilletid med god kvalitet
    - 44 100 målinger per sekund
    - 2^{16} ? 65 536 intervaller (2 byte)
    - 2 kanaler
    - + 37 % feilkorreksjonsdata

- CD må ha plass til 783 MB
- Spare plass:
    - Lagre bare én kanal og litt informasjon om forskjellen mellom høyre og venstre kanal, i stedet for å lagre all informasjonen
        - Små forskjeller, og man reduserer nesten halvparten av dataen
    - Ikke lagre hver måling, men lagre forskjellen
        - Må lagre hele verdien innimellom pga. feil eller spoling
    - Mennesker hører ikke over 20 Hz eller under 20 000 Hz
    - Hvis vi hører en kraftig lyd med én frekvens hører vi ikke en litt svakere lyd med høyere frekvens.
    - Etter å ha hørt en kraftig lyd hører vi dårligere inntil 0,2 sekunder etterpå
- Moderne standarder:
    - mp2 (brukt i DAB)
    - mp3 (mp3-spillere, iPod)
    - ACC (DAB+, youtube, itunes osv.)
    - Tillater til dels sterk komprimering:
        - Cd – 1410 kb/s
        - mp2 – ca. 256 kb/s
        - mp3 – ca. 160 kb/s
        - ACC – ca. 128 kb/s
    - Dersom det komprimeres ytterligere vil kvaliteten gå ned, men denne komprimeringen gir god kvalitet

ASSEMBLERKODE

Ord og begreper:

- **Maskinkode** er den koden prosessoren utfører
- **Assemblerkode** er en tekstlig form av maskinkoden
- **Registre** er lagerlokasjoner helt tett på prosessoren; de er på 64 bit (dvs 8 byte)
- **Flagg** er 1-bits lagre som settes automatisk av prosessoren. Flagg er 0 eller 1 avhengig av hva som skjer i prosessoren.

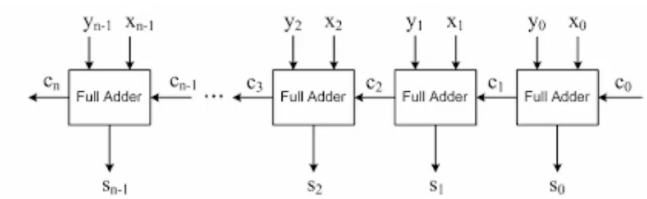
Registre:

- %RAX - her ender svaret
- %RCX - counter
- %RDI - parameter 1
- %RSI - parameter 2
- %RDX - parameter 3 (Hvor man finner “rest” etter divisjon)
- %R8 - alt mulig
- %R9 - alt mulig
- %R10 - alt mulig
- %R11 - alt mulig

Flagg:

Registre på bare 1 bit (vs 64 bit)

→ Binær informasjon (1 eller 0)



- S (“Sign” = fortegn) får en kopi av det øverste bit-et (“fortegnsbit-et”) etter operasjonen. S-flagget blir 1 dersom tallet er negativt, 0 hvis det er positivt (S(n-1)).
- C (“Carry”) får det mentebit-et som forsvinner ut (Cn, f.eks. 1 + 1 = 0, med 1 i mente).
- Z (“Zero”) settes til 1 hvis svaret er 0 (tar alle 64-bitene inn i en NOR-krets og sjekker om svaret er 0)

Instruksjoner:

- **movq** flytter data (tar kopi, registeret man flytter fra forblir uendret)
- **negq** snur fortegnet
- **addq** legger sammen
- **subq** trekker fra
- **imulq** multipliserer
- **idivq** dividerer (sær!!)
- **cqo** hjelp for idivq
- **andq** bitvis AND
- **orq** bitvis OR
- **xorq** bitvis XOR
- **notq** bitvis NOT
- **ret** returnerer

- **jmp** hopp
- **js** hopp hvis S-flagget = 1
- **jns** hopp hvis S-flagget ikke lik 1
- **jc** hopp hvis C = 1
- **jnc** hopp hvis C ikke er lik 1
- **jz** hopp hvis Z = 1
- **jnz** hopp hvis Z != 1

Divisjon:

f.s

```
# def f(a, b, c): Beregn a * b / c

        .globl  f
f:
        movq    %rdi,%rax  # Hent a og
        imulq   %rsi,%rax  # gang med b
        movq    %rdx,%r8   # Hent c og
        cqo                #
        idivq   %r8        # del a*b med c.
        ret
```

Variabler

- Gir mer plass en registrene tilbyr, da kan man legge data i minnet.
- Notasjon:

```
        .data
var1:   .quad 0
var2:   .quad 0
var3:   .quad 1
```

Her har var1 og var2 verdi 0 og var3 verdi 1.

Måter å sette 0 inn i %RAX:

```
imulq   $0,%RAX
subq    %RAX,%RAX
andq    $0,%RAX
xorq    %RAX,%RAX
```

LOGISKE PORTER OG ALU

Porter:

- Et hvilken som helst boolsk funksjonsuttrykk kan implementeres med bare NAND
  - Vi foretrekker NAND
- Et hvilket som helst boolsk funksjonsuttrykk kan implementeres med bare NOR

Minterm:

- En funksjon kan være gitt på ”sum av produkt” form
  - Eksempel: F = xy + xy’ + x
- Hvert ”produktledd” som inneholder alle variablene kalles minterm
- For to variabler finnes det 2^2 forskjellige mintermer, for tre variabler finnes det 2^3 forskjellige mintermer
  - xy + x’y + xy’ + x’y’
- Notasjon: m_x

Maksterm:

- En funksjon kan være gitt på ”produkt av sum” form
  - Eksempel: F = (x+y)(x+y’)y
- Hvert ”summenledd” som inneholder alle variablene kalles maksterm
- For to variabler finnes det 2^2 forskjellige makstermer
  - (x+y)(x’+y)(x+y’)(x’+y’)
- Notasjon: M_x

Karnaughdiagram:

Metode for å forenkle boolske uttrykk

Uttrykket må være representert ved sum av mintermer

metoden egner seg for funksjoner med 2 - 4 (5) variabler

Grupperer naboruter som inneholder 1, men antall elementer må være en potens av 2

Hjørneruter er naboruter

Topp og bunn er naboruter

Jo større ruter, desto enklere uttrykk

Ved å lese ut de tomme rutene fra diagrammet får vi F’

Vi forenkler uttrykket for å spare porter

HVA X BETYR

I et karnaughdiagram med fire variabler, slår sammen to ruter gir et uttrykk med tre variabler (blir kvitt én av variblene) Slå sammen fire ruter gir et uttrykk med to variabler. Slå sammen åtte ruter gir et uttrykk med én variabel.

Binær adder:

- Vanlige anvendelser: mikroprosessor ALU, Xbox, mikserbord, digitalt kommunikasjonsutstyr, AD-DA omformere osv.
- Basis for addisjon, subtraksjon, multiplikasjon, divisjon og mange andre matematiske operasjoner
- All form for filtrering/signalbehandling

Halvadder:
- Ingen mente inn
- To innganger, to utganger
- Adderer de to minste signifikante bittene

Fulladder:
- Adderer sammen bit A_n og B_n med evt. mente inn
- Tre innganger, to utganger

Komparatorer:
- nyttig krets for å sammenligne to bits, er den større eller mindre
- brukes for å bestemme ting, skal vi gå den veien eller den veien

Multiplekser:
- Nyttig krets i design, gjør at du er i stand til å velge hvilket inngangssignal du vil se på
- En inngang som velger hvilken av de andre inngangene den vil slippe igjennom til utgangen
- Demultiplekser: velger hvilken av utgangene den skal sende signalet til

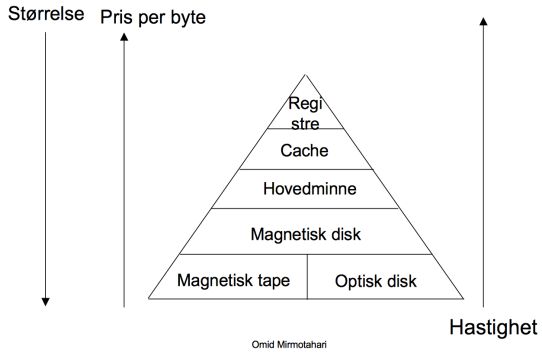
Aritmetisk logisk enhet (ALU):
- Den delen av CPU hvor logiske og aritmetiske beregninger/operasjoner utføres
    - Addisjon, subtraksjon, AND, OR osv.
- Vi trenger fulladder, multiplekser, AND, OR og NOT
- ALUer i CPU:
    - Moderne CPU kan inneholde flere ALUer
    - ALUer kan designes til å håndtere flere funksjoner per trinn og mer komplekse.
    - Alternativt kan man bruke software aktivt til å designe slik at en ALU kan utføre komplekse operasjoner over flere trinn.
    - TRADE-OFF: hvor skal man plassere kompleksiteten, i hardware eller software
    - Ulemper med å sette kompleksitet i hardware er høyere kostnader i fbm strømforbruk, areal og produksjonskost.
    - ALU designet spiller en stor rolle med hensyn på CPU sin ytelse, da det mest komplekse operasjonen setter maksimal klokkefrekvens.

Flipflop

Vi bruker flipflops fordi vi vil ha et stabilt signal gjennom en klokkesykel. Det er for at komponentene i kretsen skal kjøre synkronisert. Det finnes ulike typer flipfloper:
- D-flipflop låser Q (utgangsverdien) til D (inngangsverdien) på hver positive klokkeflanke.

- JK-flipflop har to inngangsverdier J og K, der J setter Q til 1 og K setter Q til 0. Når både J og K er 0 er Q låst, og når begge er 1 inverteres Q.
- T-flipflop er det samme som JK-flipflop, bare at det er inngang koblet til både J og K. Når T er 1 inverteres Q.

MINNEHIERARKI:



- **Register**:
  Bruksområder:
  - Intern kladdeblokk for CPU'en med en rask aksess til innholdet

  Lagringskapasitet:
  - Integrert i CPU'en, som en del av kretsene, lagret på chip, relativt få (32-128 stk/bit)
- **Cache**:
  Bruksområder:
  - Ligger i CPU
  - Hurtig mellomlager for både instruksjoner og data for å jevne ut hastighetsforskjellen mellom CPU'en og hurtigminnet

  Lagringskapasitet:
  - Mellomlagrer internt (L1) eller i nærheten (L2, L3) av CPU'en, typiske kapasiteter er fra 10 KiloByte (L1) til 1 MegaByte (L2) og flere MegaByte(L3)

Minnehierarki – aksesshastighet

Registers	< 1ns	≈ 100 Byte
L1 (på CPU) cache	≈ 1ns	≈ 10 KB
L2,L3 (utenfor CPU) cache	2-10ns	≈ 1 MB
Hovedminne (RAM)	20-100ns	≈ 1 GB
SSD/Flash	100ns-1us	≈ 1 TB
Harddisk	1ms	≈ 1 TB

- **RAM** (random access memory):
  Bruksområder:
  - Utenfor CPU'en
  - Buffer mellom ekstern lagringsmedium og CPU'en med rask lese- og skriveaksess

  Lagringskapasitet:
  - Internt på hovedkortet i nærheten av CPU'en, størrelser opptil flere GigaByte
  - Hvor mange RAM man kan ha har en sammenheng med hvor mange bit CPU'en er.
    - 32 bit CPU = 2^{32} adresselinjer i RAM
- **SSD**(solide state-disker)/**Flash/Harddisk**:
  Bruksområder:
  - Høykapasitetsmedium for program/data
  - "langt" unna CPU

  Lagringskapasitet:
  - Ekstern eller intern lagringsenhet i maskinen med kapasitet opptil flere TeraByte

Pris, strøm, hastighet og plass

Husk på Erlend:
Stor forskjell på register, cache og RAM i forhold til harddisk er at minnet slettes når strømmen skrus av!

Antall kjerner vs antall cache (hastighet og lagring) i forhold til framtidige maskiner

DATAMASKINARKITEKTUR

ALU (Aritmetisk logisk enhet) er den delen av CPU hvor logiske og aritmetiske beregninger utføres, for eksempel addisjon, subtraksjon, AND, OR osv.

Assemblerkode tolkes til et bitmønster, som har 64 0'ere eller 1'ere, som datamaskinen forstår. De ulike delene av remsen har en mening, og dette må dekodes og det gjøres av CPU. Forsinkelse i CPU skapes av at klokken til CPU'en må ta hensyn til den tregeste prosessen. Problem: hvordan effektivisere? Løsning: Pipeline.

Pipeline fungerer etter samlebåndsprinsippet, ulike prosesser kan skje samtidig. Tenk på Omid som vasker klær. Hvis man har flere subinstruksjoner som skal skje samtidig, vi kan utføre subinstruksjoner fra forskjellige instruksjoner samtidig dersom instruksjonene er ulike. Dette gjør at vi sparer tid, og at det blir mer effektivt. Det at man har en 4 trinns pipeline vil ikke si at man får 4 ganger raskere prosessering, da det alltid vil gå noe tid til administrering av instruksjoner.

- Må lagre verdiene/dataene til hver subinstruks, eller så kan dataen bli overskrevet av neste instruks og vil skape trøbbel
- Klokkesignal styrer synkroniseringen

Det kan oppstå tre hazarder ved pipelining, resource hazard, data hazard og control hazard.

- **Resource hazard:** oppstår hvis to subinstruksjoner i pipelinen ønsker å aksessere samme ressurs. **Løsninger**: design hvor dette ikke skjer, stoppe (STALL) pipelinen lenge nok til å aksessere sekvensielt, bruke lokale register som er organisert i en registerfil, bruke Harvard arkitekturen som har to separate minner for data og instruksjon (ingen er særlig gode løsninger)
- **Data hazard**: oppstår hvis to forskjellige instruksjoner trenger å aksessere samme data samtidig, den ene instruksjonen vil da trenge et resultat fra forrige instruksjon før den har produsert gyldig svar. **God løsning**: Ved hjelp av forwarding kan man lage en snarvei i pipelinen, i dette tilfellet en direkte datapath som aktiveres ved en hazard.
- **Control hazard**: Hvis vi har en pipeline som innhenter neste instruksjon fortløpende vil det oppstå problemer når vi får en JUMP instruksjon (js, jns osv). Da må vi tømme pipelinen for andre instruksjoner og lese inn den nye. Løsning: ikke hente noe annet før man vet om man skal hoppe eller ikke, forutsi om man skal hoppe, hvis vi dobler hardware så vil vi ha to parallelle løp hvor den ene ikke hopper og andre hopper og deretter bestemmes det til slutt hvilken som skal brukes.

Dette er koblingen mellom assemblerkode og maskinkode.

Når man substraherer binære tall må man addere det 2'er komplementet til det som skal trekkes fra:

2’er komplement

Setter minus foran et binært tall ved å invertere alle bittene og plusse på 1

Eksempel:

Finner -5:

invertert 5:	1 0 1 0	
	+ 0 0 0 1	
-5:	= 1 0 1 1	

7	0 1 1 1
6	0 1 1 0
5	0 1 0 1
4	0 1 0 0
3	0 0 1 1
2	0 0 1 0
1	0 0 0 1
0	0 0 0 0
-1	1 1 1 1
-2	1 1 1 0
-3	1 1 0 1
-4	1 1 0 0
-5	1 0 1 1
-6	1 0 1 0
-7	1 0 0 1
-8	1 0 0 0

Omid Mirmotahari

Eksempel: 5 + (-)3

101
- 0011 = 1101 (2’erkomplement)

=

0101
+ 1101
= (1)1010