IN1020 – Lagdeling og hjemmeruter
Gruppetime 3 (uke 37)

Gruppelærer

Institutt for informatikk, UiO

Plan for økten

Forrige uke

2. september – Lagdeling i datakommunikasjon
4. september – Hvordan fungerer din hjemmeruter?

Plan for økten

1. Protokoller

Hva er en protokoll, header og hvorfor vi trenger dem.

2. Lagdeling

Hvorfor lag, OSI vs. TCP/IP, og lagene i Internett.

3. Lagene i TCP/IP-modellen

Link, nettverk, transport og applikasjonslag – roller og adresser.

4. Hjemmeruter og subnett

ARP, MAC, broadcast, CIDR og kringkastingsadresse.

5. NAT, DHCP og IPv6

Hvordan hjemmenett får IP-adresser og snakker med Internett.

6. Transportlag (TCP/UDP)

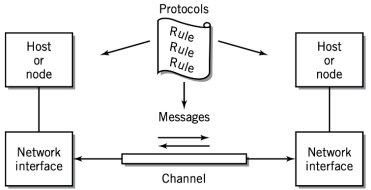
Hva tjenestene er, og når vi velger UDP vs. TCP.

Vi bruker øvingsoppgavene som ramme for diskusjon og repetisjon.

Hva er en protokoll?

Definisjon

- En **protokoll** er et sett av regler som styrer kommunikasjon mellom maskiner:
  - hvilke beskjeder som kan sendes
  - i hvilken rekkefølge
  - hvordan de tolkes og hva som skjer når de mottas
- Vi trenger protokoller fordi:
  - maskiner fra hele verden skal kunne forstå hverandre
  - forskjellige teknologier (LAN, WiFi, fiber) skal kunne samarbeide
- Analogt til **språk og høflighetsregler** i menneskelig kommunikasjon.



Hva er en protokoll?



Header og protokollstakk

Header-lagdeling

- En **header** er metadata protokollen trenger for å levere tjenestene sine:
  - adresser, portnummer, lengde, feildeteksjon, m.m.
- Når data sendes nedover i protokollstakken:
  - hvert lag legger til **sin** header på nyttelasten.
  - f.eks. applikasjonsdata $\rightarrow$ TCP-header $\rightarrow$ IP-header $\rightarrow$ Ethernet-header.
- På mottakersiden:
  - headerne tas av i **motsatt rekkefølge** når data går oppover i stakken.

«Løkskall»-modell: hvert lag pakker inn data med sin header.

Oppgave 1: Protokoller

Diskuter i grupper (5 min)

1. Protokoller

(a) Hva er en protokoll, og hvorfor trenger vi det i datakommunikasjon?

(b) Hvorfor trenger protokoller en header?

(c) Hvordan blir header lagt til av de forskjellige lagene i en stack, og hva skjer på mottakersiden?

Bruk illustrasjonen fra forrige slide + forelesningen når dere svarer.

Fasit: Oppgave 1 (Protokoller)

Forslag til løsning

(a) En protokoll er regler for hvordan kommunikasjonen mellom maskiner skal foregå.

- Nødvendig for at ulike maskiner og teknologier på tvers av verden skal kunne kommunisere.

(b) For å håndtere og overlevere metadata som trengs for å levere tjenestene protokollen tilbyr:

- adresser, porter, lengde, feildeteksjon osv.

(c) Header blir lagt til lag for lag når pakken går nedover stakken:

- applikasjonsdata får TCP-header, så IP-header, så Ethernet-header.
- routere kan ta av og bytte ut headere for linklaget.
- på mottakersiden tas headerne av lag for lag i motsatt rekkefølge oppover mot applikasjonslaget.

Lagdeling: hvorfor organisere i lag?

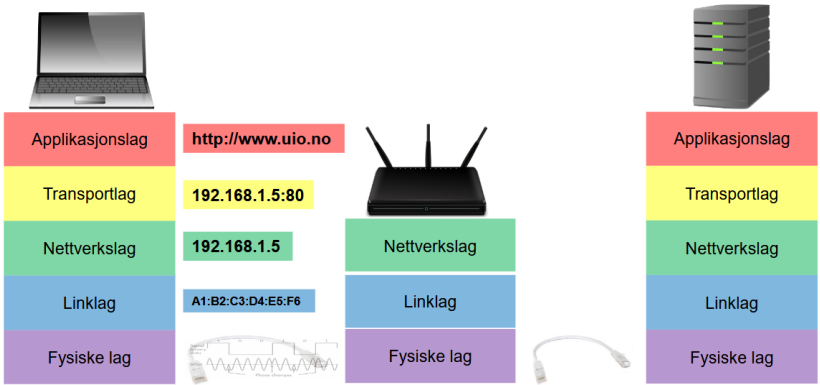
Lag		Funksjon
5	**Applikasjon**	Applikasjonsrelaterte tjenester (**HTTP**, **Mail**)
4	**Transport**	Kobler sammen systemene ende-til-ende (**TCP/UDP**)
3	**Nettverk**	Sende data fra ende-til-ende systemer (**IP**)
2	**Link**	Pålitelig overføring mellom to noder (**LAN/WiFi**)
1	**Fysisk**	Sender bit ut på mediet (kablet eller trådløst)

Lagdeling: hvorfor organisere i lag?

Motivasjon

- Hvert lag leverer et **unikt sett med tjenester** til laget over:
  - link: pålitelig overføring på lokal link
  - nettverk: levere pakker mellom vertsmaskiner
  - transport: ekstra tjenester (pålitelighet, porter, rekkefølge)
  - applikasjon: tjenester som web, e-post, DNS
- Fordeler med lagdeling:
  - kan bytte ut teknologi på ett lag uten å endre alle lag
  - komponenter (switcher, rutere) kan implementere bare noen lag
  - gir **klar struktur** og lettere feilsøking

Lagene i Internett (TCP/IP-modellen)



Lagene i Internett (TCP/IP-modellen)

Oversikt

- Typiske lag i Internett (TCP/IP-modellen):
  - Fysiske lag
  - Linklaget
  - Nettverkslaget (IP)
  - Transportlaget (TCP/UDP)
  - Applikasjonslaget
- Hvert lag har en tydelig oppgave:
  - f.eks. linklaget: frames, MAC, lokal overføring
  - nettverk: IP-adresser og ruting
  - transport: TCP/UDP-tjenester
  - applikasjon: protokoller som HTTP, SMTP, DNS

Oppgave 2: Lagdeling

Jobb 2 og 2 (7–8 min)

2. Lagdeling

(a) Hvorfor er nettverket organisert i lag?

(b) Hvilke lag finner vi vanligvis i Internett?

(c) Hva er oppgaven til hvert av de ulike lagene?

(d) Hvilke av lagene i TCP/IP-modellen kommuniserer kun ende-til-ende?

(e) Hva er en MAC-adresse, og på hvilket lag brukes den?

(f) Hvilke lag i nettverksstakken kan vi kryptere, og hva er fordeler og ulemper med dette?

Bruk figuren med lagene i Internett og tenk på eksempler som HTTPS, VPN og WiFi-kryptering.

Fasit: Oppgave 2 (Lagdeling) – Del 1

(a) Hvorfor er nettverket organisert i lag?

(a) **Hvorfor lag?**

- Nettverket må løse mange forskjellige oppgaver:
  - finne riktig endesystem og program
  - finne vei mellom nettleverandører
  - håndtere problemer i nettet, forsinkelser, høy trafikk
  - sørge for at mottaker forstår data, ivareta personvern og sikkerhet
- **Lagdeling** deler dette i abstraksjonsnivåer:
  - tydelig ansvarsfordeling per lag
  - lag kan utvikles/endres uavhengig av hverandre
  - like problemer løses én gang i riktig lag

Fasit: Oppgave 2 (Lagdeling) – Del 2

(b) Hvilke lag? og (c) Oppgaven til hvert lag

(b) **Hvilke lag finner vi vanligvis i Internett (TCP/IP-modellen)?**

- Fysiske lag
- Linklag
- Nettverkslag (IP)
- Transportlag (TCP/UDP)
- Applikasjonslag

(c) **Oppgaven til hvert lag**

- **Fysisk lag**: signalrepresentasjon av bits, kabler/medium, spenning osv.; sørger for at en 1 blir mottatt som 1.
- **Linklag**: deler bits i frames, pålitelig overføring mellom noder på samme link, feildeteksjon, enkel flytkontroll, MAC.
- **Nettverkslag (IP)**: leverer pakker mellom vertsmaskiner på tvers av nettverk (ruting i Internett).
- **Transportlag (TCP/UDP)**: kommunikasjon mellom prosesser (porter), pålitelighet, rekkefølge, flyt- og metningskontroll (TCP) eller lettvekts «best-effort» (UDP).

Fasit: Oppgave 2 (Lagdeling) – Del 3

(d) Ende-til-ende, (e) MAC og (f) Kryptering

(d) **Hvilke lag kommuniserer kun ende-til-ende?**

- **Applikasjonslaget** og **transportlaget** – de ser «hele veien» mellom endesystemene, mens IP og link håndteres også i rutere/switcher.

(e) **Hva er en MAC-adresse, og på hvilket lag brukes den?**

- MAC-adresse identifiserer et nettverksinterface (NIC) i f.eks. Ethernet/WiFi.
- Brukes på *linklaget* for å adressere noder på samme LAN.

(f) **Hvilke lag kan vi kryptere, og fordeler/ulemper?**

- Mulig på flere lag:
  - Applikasjon (TLS/SSL, f.eks. HTTPS)
  - Transport (f.eks. QUIC over UDP med innebygd kryptering)
  - Nettverk (IPSEC/VPN)
  - Link (WPA2/WPA3 på WiFi)
- Fordel: bedre konfidensialitet og sikkerhet.
- Ulempe: mer overhead og kompleksitet, kan gjøre feilsøking vanskeligere.

Linklaget (Lag 2): frames og MAC-adresser

Rolle og adresser

- Oppgave: **pålitelig overføring** av frames mellom noder på samme link.
- MAC-adresser:
  - identifiserer et nettverksinterface (NIC) på linklaget.
  - brukes til å adressere andre maskiner på samme lokale nettverk (LAN).
- Frames inneholder:
  - MAC-kilde og MAC-destinasjon
  - lag 3-pakke (IP) som nyttelast
  - felter for feildeteksjon, m.m.

Switch jobber primært på dette laget og bruker MAC-adressetabell.

Nettverkslaget (Lag 3): IP og CIDR

IP-adresser og subnett

- Nettverkslaget (IP) leverer pakker til **vertsmaskiner** på Internett.
- IP-adresser skrives ofte i **CIDR-notasjon**, f.eks. 192.168.0.165/28:
  - første del: IP-adresse
  - andre del: nettverksmaske (hvor mange bits som er «nett» vs. «vertsdel»)
- Subnett og kringkastingsadresse:
  - subnett: IP-adresse AND nettverksmaske
  - kringkastingsadresse: IP-adresse OR invers av nettverksmaske

IP-lekeapp i Python

Hvor finner dere programmet?

- Vi har laget en liten **IP-lekeapp** i Python som hjelper dere å regne på subnett- og kringkastingsadresser (oppgave 5).
- Programmet ligger i **mappen til gruppen vår på emnesiden** for IN1020.
- Bruk den til å sjekke fasit etter at dere har prøvd å regne for hånd.

Eksempel på bruk

```
$ python ip_utregning.py
CIDR > 192.168.1.50/24
CIDR > 192.168.100.10/26
CIDR > 10.10.100.130/23
```

Oppgave 4: Nettverkslaget og CIDR

Regneoppgave (10 min)

4. Nettverkslaget

(a) IP-en til en maskin er gitt i CIDR-notasjon ved: `192.168.0.165/28`

i. Hva er IP-adressen med hver oktett (8) notert binært?
ii. Hva blir nettverksmasken når hver oktett er notert binært?

(b) For å finne subnettadressen til maskinen må du gjøre en bitvis AND-operasjon mellom IP-adressen og nettverksmasken.

i. Hva blir subnettet til maskinen over i CIDR-notasjon?
ii. Hvor mange adresser er det i vertsdelen av dette subnettet?

(c) For å finne kringkastingsadressen til et subnett, må du gjøre en bitvis OR-operasjon mellom maskinens IP-adresse og bitkomplementet (bitvis invers) av nettverksmasken.

i. Hva er kringkastingsadressen til subnettet fra forrige oppgave?

Skriv opp IP og maske binært før dere gjør AND/OR – da går det lettere.

Fasit: Oppgave 4 (Nettverkslaget og CIDR) – Del 1

(a) IP-adresse og nettverksmaske i binær

Gitt IP-en 192.168.0.165/28.

(a.i) IP-adresse i binær

- IP: `192.168.0.165`
  - $192 = 11000000, 168 = 10101000, 0 = 00000000, 165 = 10100101$

$\Rightarrow$ `11000000.10101000.00000000.10100101`

(a.ii) Nettverksmaske i binær

- Prefiks /28 betyr at de første 28 bits er nettverksdel, resten vertsdelen (nettverksmaske med 28 stk. 1-bits og 4 stk. 0-bits) <ref: index=61981247 firstWord=1 lastWord=20/>.
- Maske: `255.255.255.240`
  - $255 = 11111111$ (tre første oktetter)
  - $240 = 11110000$

$\Rightarrow$ `11111111.11111111.11111111.11110000`

Fasit: Oppgave 4 (Nettverkslaget og CIDR) – Del 2

(b) Subnettadresse og vertsdelen

(b.i) Subnettadresse

- Subnett = bitvis AND mellom IP-adressen og nettverksmasken (oktett for oktett) <ref: index=61981252 firstWord=9 lastWord=23/>.
- `192.168.0.165` AND `255.255.255.240` ⇒ `192.168.0.160/28`
- Dette er *første* adresse i subnettet og identifiserer selve subnettet.

(b.ii) Antall adresser i vertsdelen

- Prefiks /28 gir $32 - 28 = 4$ vertsbits.
- Totalt antall adresser i subnettet: $2^4 = 16$.
- 2 reserveres (subnett + kringkasting) ⇒ $16 - 2 = 14$ vertsadresser.
- Intervallet går fra subnettadressen til kringkastingsadressen, med vertsadresser imellom <ref: index=61981254 firstWord=1 lastWord=20/>.

Fasit: Oppgave 4 (Nettverkslaget og CIDR) – Del 3

(c) Kringkastingsadresse

(c.i) Kringkastingsadresse

- Kringkastingsadressen finnes ved:
  - bitvis OR mellom IP-adressen og bitkomplementet (invers) av nettverksmasken <ref: index=61981255 firstWord=7 lastWord=23/>.
- Invers maske til `255.255.255.240` er `0.0.0.15` (00000000.00000000.00000000.00001111).
- `192.168.0.165` OR `0.0.0.15` ⇒ `192.168.0.175`
- `192.168.0.175` er *siste* adresse i subnettet; trafikk hit kringkastes til alle i subnettet.

Oppgave 5: Treningsoppgaver IP-adresser

Jobb to og to (8–10 min)

5. Regning med IP-adresser – treningsoppgaver

- Finn **subnettadressen** og **kringkastingsadressen** for:
  - `192.168.1.50/24`
  - `192.168.100.10/26`
  - `10.10.100.130/23`

Tenk: «nett» + «vertsdel» og hvor mange adresser vertsdelen gir.

Fasit: Oppgave 5 (IP-treningsoppgaver)

Forslag til løsning

- `192.168.1.50/24`
  - Subnett: `192.168.1.0`
  - Kringkastingsadresse: `192.168.1.255`
- `192.168.100.10/26`
  - Subnett: `192.168.100.0`
  - Kringkastingsadresse: `192.168.100.63`
- `10.10.100.130/23`
  - Subnett: `10.10.100.0`
  - Kringkastingsadresse: `10.10.101.255`

Transportlaget (Lag 4): TCP og UDP

Tjenester

- Transportlaget tilbyr **kommunikasjon mellom prosesser** (porter) på endesystemer.
- UDP:
  - multiplexing over IP-adresse (porter)
  - kontroll av nyttelast (checksum)
  - liten overhead, ingen garanti for levering/rekkefølge
- TCP:
  - samme som UDP +:
    - pålitelighet (garantert levering)
    - levering i rekkefølge
    - flytkontroll (ikke sende for fort)
    - metningskontroll (dele nettverksressursene med andre datastrømmer)

Oppgave 8: Transportlaget

Kort diskusjon (5 min)

8. Transportlaget

(a) Hvilke tjenester tilbyr UDP?

(b) Hvilke tjenester tilbyr TCP?

(c) Hvorfor ønsker man noen ganger å bruke UDP, til tross for at TCP tilbyr flere tjenester?

Tenk særlig på sanntidsvideo, spill og telekonferanse.

Fasit: Oppgave 8 (Transportlaget)

Forslag til løsning

(a) UDP tilbyr:

- multiplexing over IP-adresse (porter)
- kontroll av nyttelasten via checksum

(b) TCP tilbyr:

- samme som UDP (porter + checksum)
- i tillegg: pålitelighet (garantert levering), levering i rekkefølge, flytkontroll, metningskontroll.

(c) Hvorfor UDP?

- mindre overhead (mindre ekstra data)
- applikasjonen kan implementere egen kontroll (flyt/metning) hvis ønskelig
- sendetidspunkt styres ikke av metningskontroll, nyttig for sanntidsvideo, telekonferanse, spill.

Hjemmenett og kringkastingsdomener

Broadcast og ARP

- Hjemmenett består typisk av:
  - endesystemer (laptop, mobil, printer)
  - intermediate systemer (hjemmeruter, switch, access point)
- **Broadcast-nett**:
  - én sender, alle i kringkastingsdomenet kan lytte.
  - brukes bl.a. av ARP: «Hvem har IP X?»
- For stort kringkastingsdomene:
  - mye ARP-trafikk, kan ta kapasitet og gi problemer.

NAT og IPv6

NAT

- **NAT** = Network Address Translation:
  - skriver om IP-adresser når pakker passerer en NAT-enhet (f.eks. hjemmeruter).
  - gjør at mange private IP-adresser kan gjenbruke én offentlig via portnummer.
- NAT-enheten holder oversikt over hvilke interne forbindelser som hører til hvilke eksterne porter.

IPv6

- IPv4 er i ferd med å slippe opp for adresser, selv med utstrakt NAT-bruk.
- IPv6 har 128-biters adresser (vs. 32-biters i IPv4) $\rightarrow$ langt flere adresser.
- Vanskelig å bytte IP-protokoll:
  - alle enheter som behandler IP på veien mellom avsender og mottaker må støtte IPv6.
  - dette tar tid og krever stor global koordinering.

Oppgave 6: NAT og IPv6

Diskusjon (5–7 min)

6. NAT og IPv6

(a) Hva er NAT, og hvordan fungerer det?

(b) Hva er hovedgrunnen til at man forsøker å gå over til IPv6?

(c) Hvorfor er det vanskelig å bytte IP-protokoll?

Bruk gjerne eksempelet med hjemmeruter og lokal/privat IP-adresse.

Fasit: Oppgave 6 (NAT og IPv6)

Forslag til løsning

(a) NAT:

- skriver om en IP-adresse til en annen når trafikk passerer en NAT-enhet.
- brukes mest til å gjenbruke IP-adresser i lokale nett (privat adressering innad, én offentlig utad).
- hver forbindelse kobles mot et portnummer på den offentlige adressen, og NAT holder en tabell over koblingen.

(b) Hovedgrunn til IPv6:

- vi går tom for IPv4-adresser.
- IPv6 har 128-biters adresser og gir enormt mange flere adresser.

(c) Hvorfor vanskelig overgang:

- alle enheter langs leveringsveien må støtte IPv6, ellers stopper trafikken.
- krevende å få «hele Internett» til å støtte en ny protokoll samtidig.

DHCP: automatisk IP-tildeling

Hva og hvorfor?

- **DHCP** = Dynamic Host Configuration Protocol:
  - brukes til å automatisk tildele IP-adresser til enheter på et LAN.
- Hvorfor:
  - manuelt oppsett av IP-adresser er tungvint og feilutsatt.
  - gjør at brukere kan koble seg til nettverk uten å tenke på IP-detaljer.
- IP-adresser «frigis» etter en tid:
  - adressene til enheter som ikke har vært aktive på en stund kan gjenbrukes.
  - man slipper store subnett som «holder av» adresser til inaktive enheter.

Oppgave 7: DHCP

Kort oppgave (5 min)

7. DHCP

(a) Hva er DHCP, og hvorfor bruker man denne protokollen?

(b) Hvorfor er det vanlig å «frigi» en IP-adresse som er utdelt via DHCP etter en viss tid?

Tenk på situasjoner med mange enheter som kobler seg av og på (f.eks. campus-WiFi).

Fasit: Oppgave 7 (DHCP)

Forslag til løsning

(a) DHCP:

- Dynamic Host Configuration Protocol.
- automatisk IP-adressetildeling til enheter som kobler seg til et nett.
- brukes for å slippe manuell konfigurasjon for hver bruker.

(b) Frigi IP-adresse etter en tid:

- gjør det mulig å gjenbruke adresser til enheter som ikke lenger er aktive.
- reduserer behovet for svært store subnett.

Oppsummering: lagdeling og hjemmeruter

Lagdeling

- Protokoller: regler + header + stakk ned/opp.
- Lagdeling: hvert lag har tydelige tjenester.
- TCP/IP-lagene: fysisk, link, nettverk (IP), transport (TCP/UDP), applikasjon.

Lagoppgaver

- Link: frames, MAC, lokal overføring.
- Nettverk: IP-adresser, subnett, kringkastingsadresse.

Hjemmeruter + IP-adresser

- NAT: oversetter mellom privat og offentlig adresser.
- DHCP: automatisk IP-tildeling og gjenbruk.
- IPv6: løsning på adresseknapphet, men krevende overgang.

Transportlag

- UDP: enkel, lav overhead, nyttig for sanntid.
- TCP: rik tjenestepakke med kontroll, rekkefølge og pålitelighet.

Spørsmål?

Hva vil dere
ta en ekstra runde på?

Lagdeling, IP-regning, NAT/DHCP eller TCP/UDP?