Operativ Systemer 7

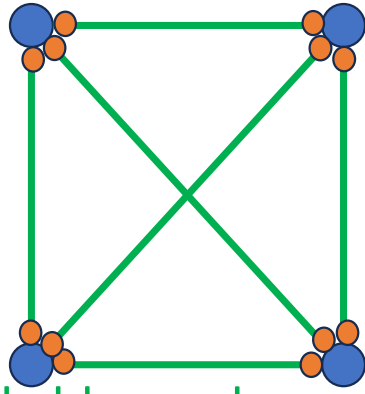
Lavet af: Vivek Misra

Topologi

- Topologi fortæller, hvordan Devices er forbundet med hinanden.
- Eksempler på Topologi er følgende:
  - Mesh
  - Star
  - Bus
  - Ring
  - Hybrid

Mesh-Topologi

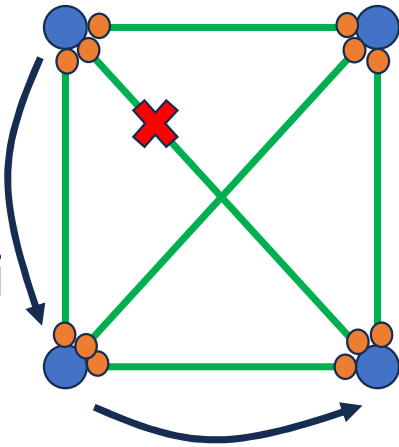
- Mesh-Topologi: Det er der, hvor alle Devices er forbundet til hinanden.
  - Vi kan se, at punkterne som er betegnet som Devices er forbundet med hinanden.
  - Når man snakker om Mesh-Topologi, så er det følgende:
    - Antal af kabler,
    - Antal af Porter
    - Reliabilitet
    - Sikkerhed
    - Kost og Penge



- Kabler siges, at jo flere Devices som er forbundet med hinanden, desto flere kabler er der. Men det betyder, at man udefra noder kan finde ud af hvor mange kabler der er brugt til at danne forbindelse. OBS: NODER = Antal af Devices.
- Porter er der, hvor Devices er forbundet til hinanden. Eksempel er vist med orange prikker. Hvor det kan ses at de små prikker kendetegner det koblingspunkt, hvor porterne er forbundet med hinanden.

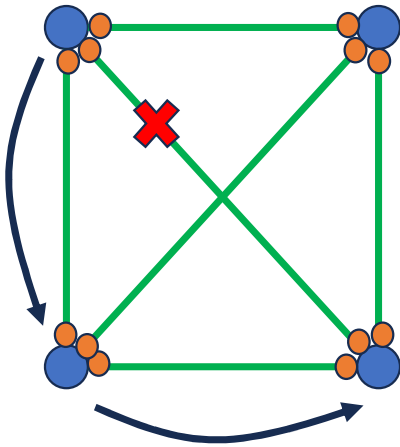
Mesh-Topologi

- Vi kan se, at Reliabilitet er der hvor vi ser om systemet kan funktionere videre, selvom der er fejl ved Beskedens Afsendelse.
- Eksempelvis hvis vi sender en Besked fra A til D og der så sker fejl ved krydset, kan vi så sende beskeden gennem en Genvej?
- Ved kost, kan det ses at den er afhængig af de forskellige kabler. Vi ved i forvejen at vores Devices er fiks-placeret og kan ikke rykkes. Vi kan se, at der er en sammenhæng og det er at jo flere Devices der er desto flere Kabler er der. Og jo flere Kabler der er desto mere koster det, at kunne sætte ting oppe i forbindelse.



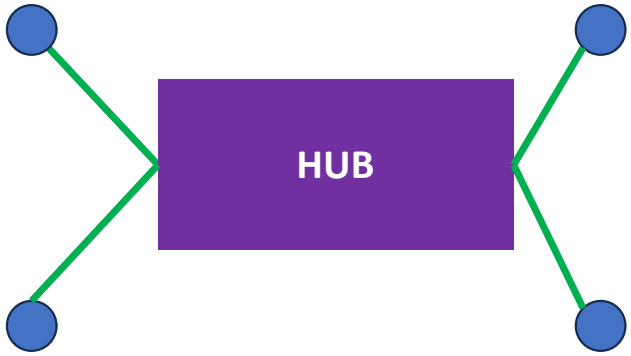
Mesh-Topologi

- Ved Sikkerhed kan det ses, at når to Devices kommunikerer med hinanden, så ved de andre Devices ikke at der foregår kommunikation mellem to Devices. Derfor er der sikkerhed i Mesh-Topologi!
- Vi kan se, at Point-to-Point går ude på, at man i et Kabel kan have flere end 2 Devices sat på. Vi kan se, at det er meget lav i Mesh, og det er fordi der er ikke så mange Devices, der er forbundet i et Kabel. **Man kalder det for en Dedikeret Kommunikation!**



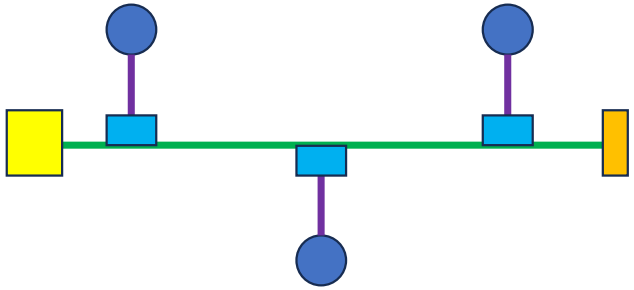
Star-Topologi

- Vi kalder STAR-Topologi også for Hub. Vi kan se, at det er en Centraliseret Device, som er også en Multiport Device. Vi kan se, at de forskellige andre Devices er forbundet med hinanden gennem Hub. Det betyder, at de andre Devices er ikke forbundet direkte med hinanden. Men at de er forbundet gennem en hub.
  - Kost: Ned
  - Kabler: Ned
  - Porter: Ned
  - Reliabilitet: Ned
  - Point-To-Point: Op
  - Sikkerhed: Op



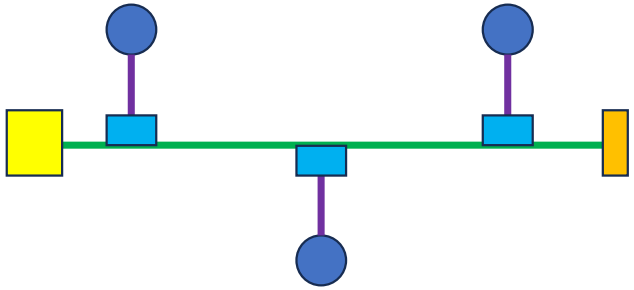
Bus-Topologi

- Det er der, hvor Devices er forbundet gennem en Kabel på en Linje.
- Vi kan se, at vi har en Backbone Linje som er den vandrette linje, som er markeret med grøn.
  - Denne linje er også kendt som Thick Ethernet Kabel, hvor man plejer at holde sin Bandwidth på Høj Niveau, grundet bedre forbindelse til Devices hængende på Backbone Kablet.
- Vi kan se, at vi har mange lilla linjer, som er kaldt for Droplines og de lyseblå kasser er kaldt for tap.
  - Tap er Elektriske Devices, som forbinder Dropline med Droplines gennem Backbone Kablet.



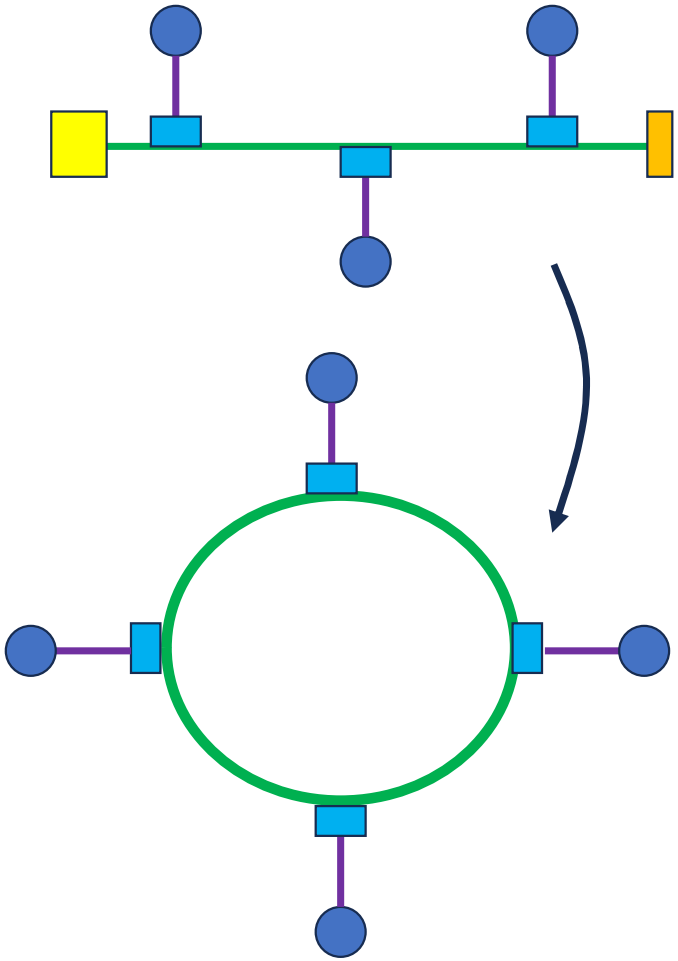
Bus-Topologi

- Vi kan se, at vi har en Terminator i slutningen af Backbone. Man har også en Repeater, som sidder det samme sted ved Terminatoren.
  - Grunden til at man sætter en Repeater er fordi hvis man ønsker at øge Backbone Kablen, gennem en Repeater. Det gør, at Repeater styrker den elektriske signal.
  - Det gør, at signalet kan køre videre i det nye ledning.
  - OBS: Det er vigtigt, at huske, at Bus-Topologi er en Multipoint Cyklus som gør at der kan se Kollision.



Ring-Topologi

- Kigger man på højre side, så er der ikke så meget forskel mellem Ring- og Bus-Topologi.
- Det kan ses, at hvis man folder Bus-Topologi om til en cirkel så er dette kaldt for en Ring-Topologi.
- Vi kan se, at Ring-Topologi har den samme arbejde som Bus-Topologien. Dette betyder, at Ring-Topologien er svag og stærk på samme punkt ligesom Bus-Topologien ift. Kabler, Sikkerhed, Porter, Reliabilitet og Penge.



Forskellige Computer Netværk Devices

Her bliver vi introduceret til Devices som er med til at danne Computer Netværk.

Forskellige Devices i Computer Netværk

- Generelt plejer vi, at bruge mange Devices i Computer Netværking. Og vi kan se, at nogen Devices består kun af Hardware og andre af en blanding mellem Software & Hardware.
- Vi giver her en generaliseret introduktion til de forskellige netværks-devices.

1. Kabler
2. Repeaters
3. Hubs

Det er rent Hardware, og der findes intet Software i disse Devices.

4. Bridges
5. Switches
6. Routers

Meget vigtige Devices.
Der findes både Hardware og Software i disse Devices.

7. Gateway = Dette er ikke en Device, men et sted hvor Devices kan tilgås og operere ved.
8. IDS
9. Firewall

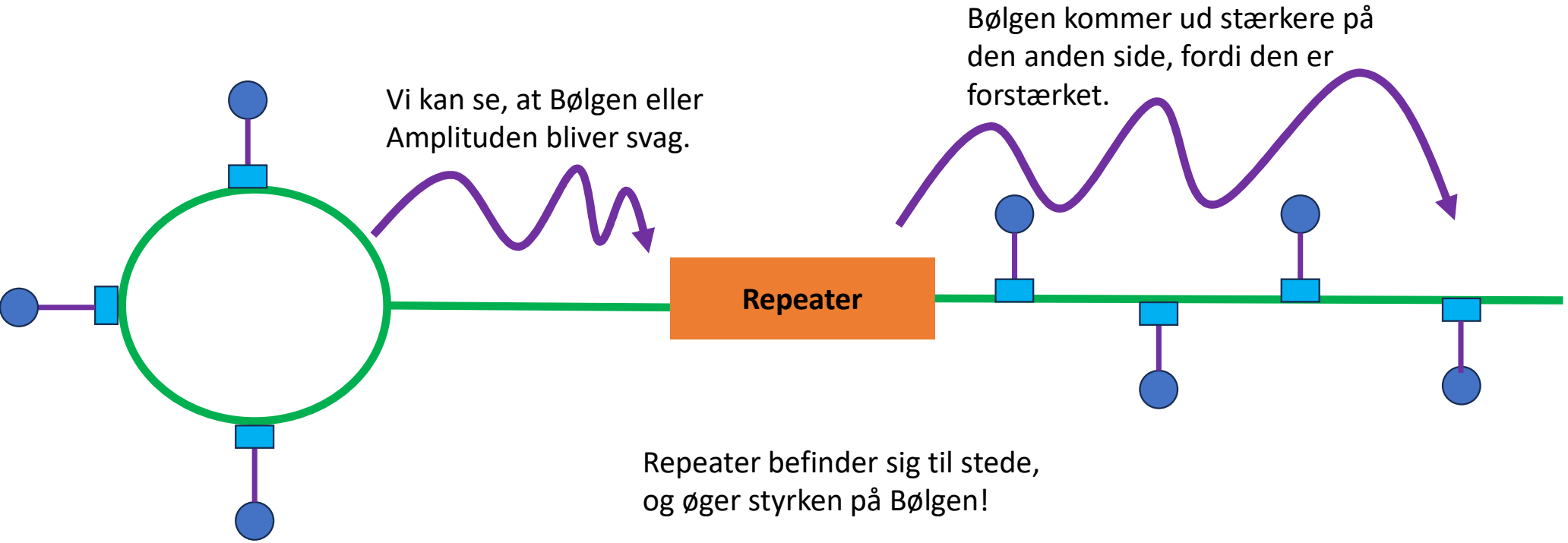
IDS og Firewall er generelt brugt som sikkerhedsværktøjer i et system.

10. Modem = Konvertere Software til Analog (Hardware) og vice versa.

Hvad er Repeaters?

- Vi har snakket om det tidligere i en Bus-Topologi Netværk.
  - Men hvorfor bruger vi egentligt Repeaters?
  - OBS: Repeater er en Hardware Device!
- Hensigten med Repeater, er at kunne føre strømmen videre til den anden Backbone Lager.
- Det er sådan, at når noget strøm går igennem en Backbone Kabel, så kan det være at strømstyrken bliver svag og endda går tabt. Tilføjes der en Repeater, så resulterer det at strømmen nemlig at Bandwidth øges og det gør at strømmen går videre.
- En Repeater findes ved det Fysiske Lager.
- Kig venligst på næste side, for at få bedre overblik.

Hvad er Repeaters?

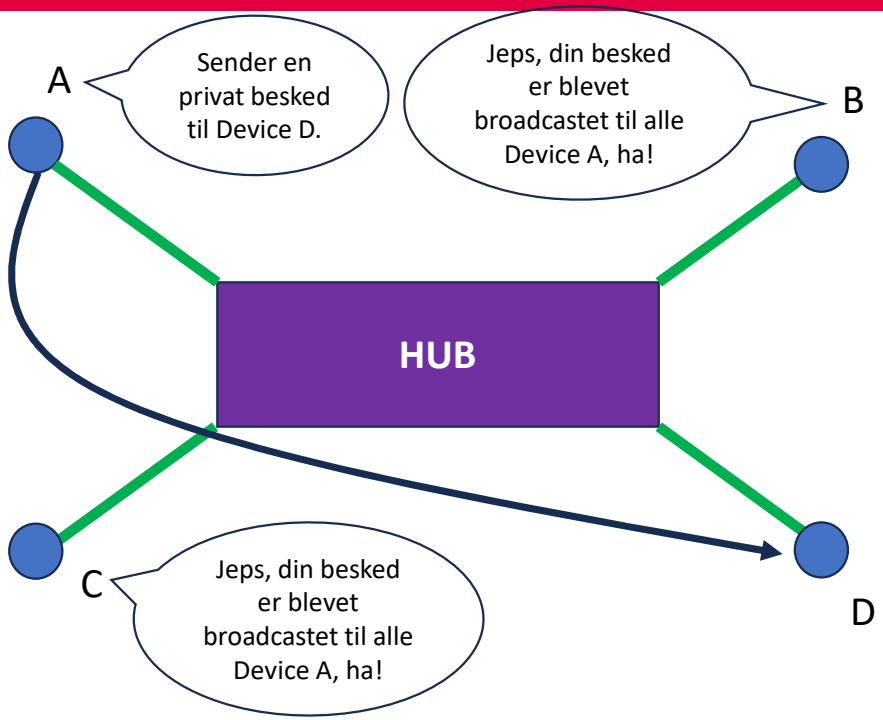


Repeater befinder sig til stede, og øger styrken på Bølgen!

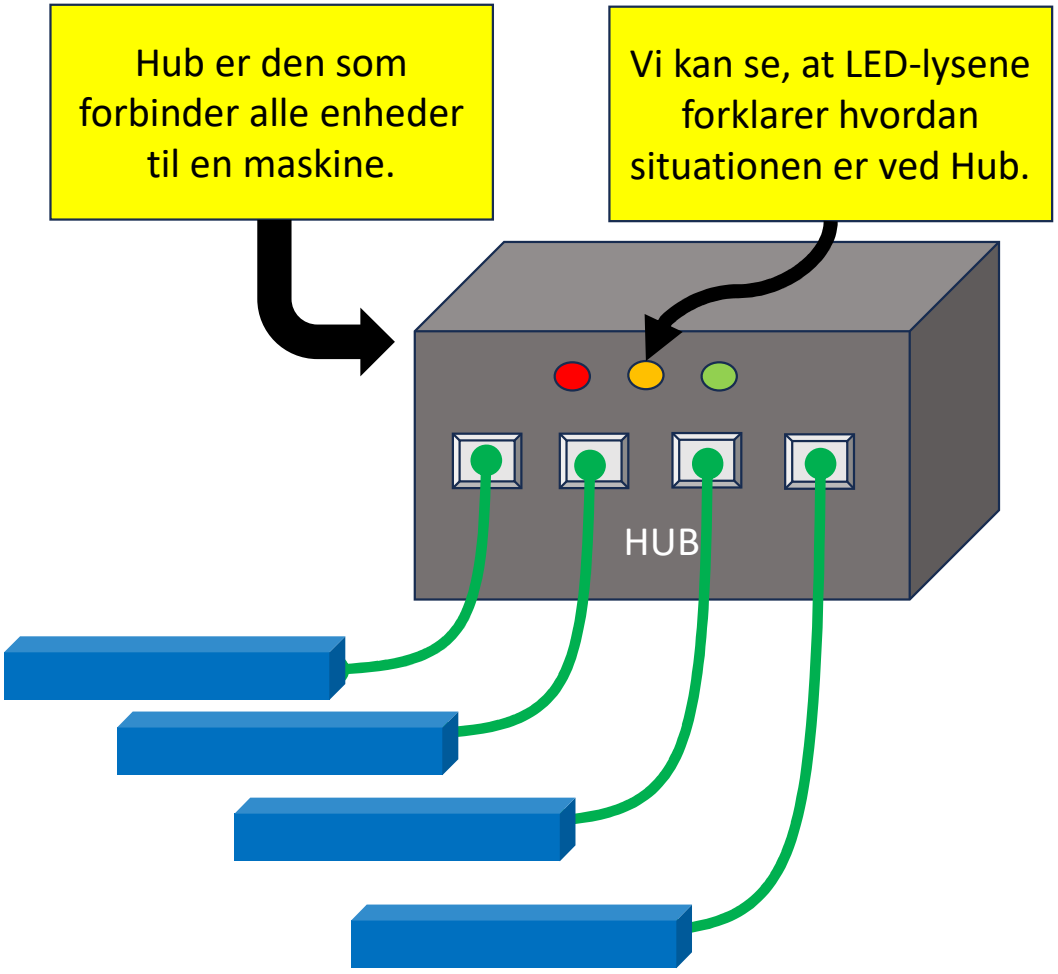
Hvad er Hub?

- Vi har også snakket om det tidligere i Star-Topologi Netværk. Men hvad bruges det til?
- Hub er en Multiport Repeater og er en Fysisk Device.
  - Grunden til at Hub er en Multiport, det er fordi man kan forbinde forskellige Device med hinanden.
- Hubs har indbygget små lyse, som sidder ved siden og fortæller om der er problemer ved lysets farver.
  - Hub laver Forwarding, fordi hvis der kommer data til den. Så vil den sende det videre.
- Man kan se i en Hub, at der er ingen Filtrering.
  - Filtrering betyder, at hvis noget er sendt fra A til B, så er det på samme tid også sendt til C og D gennem Kabler. Derved kan man sige, at der er heller ingen Security, med høj Trafik.
- Man kan se, at der er også høj Kollision, fordi hvis man sender Informationer ud på samme tid, så kolliderer de med hinanden.

Hvad er Hub?



Vi kan se, at A sender en besked til D, men der ingen Security og det betyder at C og B har allerede fået beskeden.



Hvad er Bridges?

- I en Computer Netværk bruges Bridges til at forbinde to LAN's med hinanden.
  - HUSK: LAN = Local Area Network fra KAN MAN LADE VARM VAND VÆRE I EN PAND.
- Husk, at det skal understreges, at det er to forskellige LAN's som forbindes med hinanden.
- Dette betyder, at man kan have en Ring-Token på den ene side og Bus-Token på den anden side.
- Når det kommer til Forwarding. Så kan det tydeligt ses, at hvis Bridges kan både arbejde i det Fysiske Lag og i Datalink-Lageret. Specielt i Datalink-Lageret kan Bridges tjekke MAC-Addressen.
- Ved MAC-Addressen kan det ses om Pakken skal sendes videre eller ej.
  - Eksempelvis, kan det ses at Device M1 sender en Pakke til Device M6.

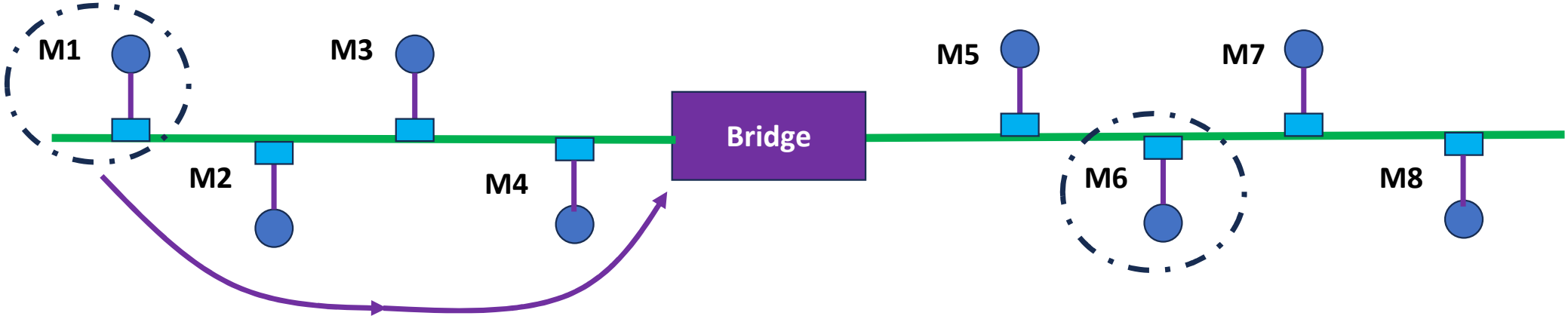
Hvad er Bridges?

- Da vi sender en Pakke gennem en Bridge. Så kan vi se, at en Bridge er istand til at afgøre om Pakken skal sendes videre udefra MAC-Adressen.
  - Dette betyder, at Pakken indeholder en source mac-adresse og en destination mac-adresse.
  - Hvis MAC-Adressen af Pakken kan læses til at være på højre side af destinations MAC-Adresse, så vil Bridge sende Pakken videre.
  - Men spørgsmålet kommer her, og det her om hvordan man kontrollere MAC-Adressen i tilfældet af Kontrollering?
    - Man bruger i tilfældet Filtrering, som betyder at Bridge kan blokere Pakken!
    - Det betyder, at hvis M1 sender til M3 og begge er på venstre side, så kan Bridge stoppe Pakkeovergangen. Men hvordan sker Blokeringen egentligt?

Hvad er Bridges?

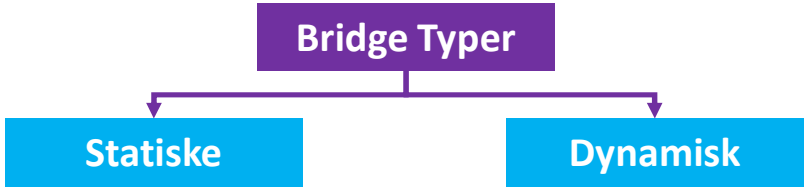
- Her viser vi et billede over, hvordan Bridges er opbygget.

Når en Device sender en Pakke, så har Pakken en MAC-Adresse som indeholder source-adressen og destination-adressen.



Source MAC	Destination MAC

Typer af Bridges



Statiske Bridges betyder at Bridges er i et fikst-placering. Det betyder, at der opstilles en tabel hvor de forskellige devices står i en side og portnummeret står i den anden side.

Netværks administratoren skriver manuelt de kendte portnummer ned ved de forskellige devices og som tjekkes af Bridge når en Pakke afsendes fra en source-devicet.

Dynamiske Bridges betyder, at der opstillet en tabel i forvejen men hvor der ikke skrives noget information om Devices placering på forhånd.

Eksempelvis hvis M1 sender besked til M6, så når Netværks Administratoren kommer til Bridge, så ved den at M1 er afsenderen. Derfor skriver den Port1 i tabellen. Men hvad med M6?

Når Net-Admin broadcaster pakken til alle og M6 sender en Pakke tilbage, så kan Bridge og Netværk Admin regne ud at M6 er placeret ved Port2 fordi den er afsenderen af en ”anden” pakke.

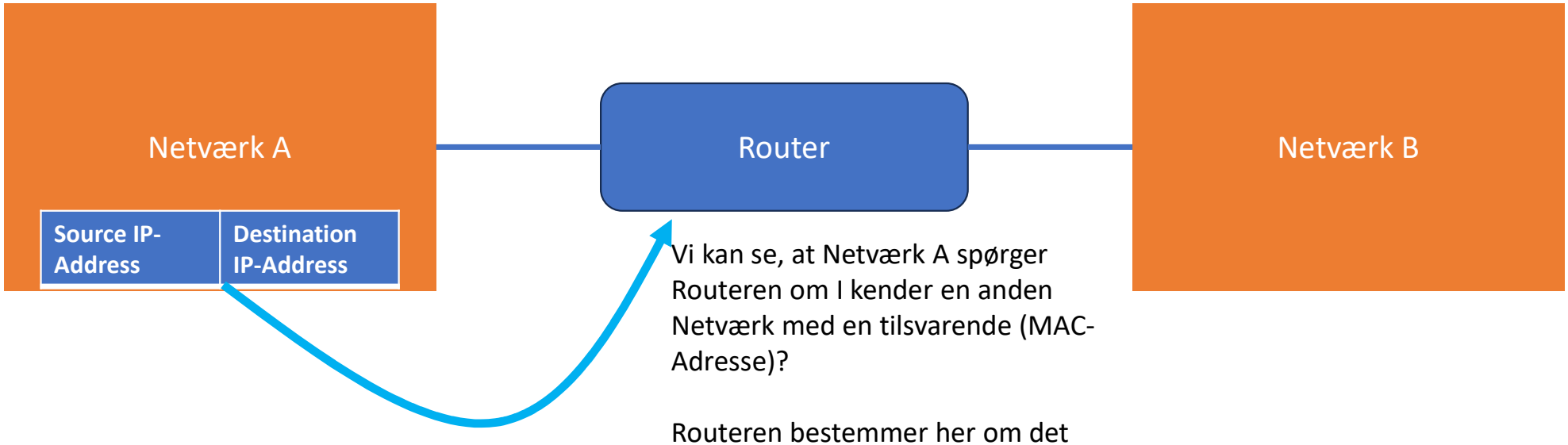
Hvad er Routers?

- Routers bruges til at forbinde to forskellige Netværker.
- Når vi snakker om Routers, kommer ”Internettet” i vores tanker.
  - Internet er en samling af Netværker.
  - Vi kan se, at Router beskæftiger sig med WAN som er Wider Area Network.
  - Routeren arbejder på den Fysiske Lager, Datalink Lageret og Netværkslageret.
- Vi kan se, at Router er en Fysisk Genstand som kan tjekke MAC-Adressen ved Datalink-Lageret. Så har vi også Netværkslageret, og det er her hvor IP-adressen tjekkes.
- Forwarding: Når IP-adresse informationen er sendt videre til Router. Så danner Routeren en Routing-Tabel, hvor det bliver afgjort hvor Pakken skal sendes videre i Netværket.

Hvad er Routers?

- Floading: I tilfældet af at Routeren ikke bestemmer, hvor Pakken skal sendes henne. Så laver den Floading og det betyder at den Broadcaster til alle Netværker.
- Filterering: Eksempelvis hvis nu Netværk A ønsker, at vide MAC-Adressen på Routeren. Det er sådan, at når vi ønsker at forbinde os til Internettet gennem hvilken som helst Netværk. Så kan vi se, at Netværk-Provideren har en Standard-Gateway med IP-Addresser. Men det gælder dog ikke for Standard-Gateway for MAC-Addresser. Det er her, hvor ARP-Request kommer på spil!
  - ARP-Request er der, hvor Netværket A spørger Routeren, om de har en MAC-Adresse tilsvarende til det IP-Adresse som netværket har på forhånd.
  - Bagefter bestemmer Routeren om der skal Forwarding eller Filtrering på Request. Men oftest er det sådan, at ARP-Request virker kun inden for et Netværk.
- Kollision: Routeren bruger Store-Forward Metoden, som betyder at data gemmes midlertidigt, hvor den processeres og derefter sendes videre. Derfor kolliderer Pakkerne ikke i Trafikken.

Hvad er Routers?



Vi kan se, at Netværk A spørger Routeren om I kender en anden Netværk med en tilsvarende (MAC-Adresse)?

Routeren bestemmer her om det skal sendes videre eller ej.

Dette er kaldt for ARP-Request.

De 3 Casts

- En cast er en definition af antal folk som vi sender data til.
  - I en netværk sender vi en samling af data gennem filer.
- Unicast eller Uni betyder en, som betyder at en person sender noget til (en) anden person.
- Broadcast: Det er der, hvor alle modtager en data.
  - Limited Broadcast: Det er der, hvor vi inde i selve Host-Netværket A sender data ud til alle Devices i Netværk A.
  - Direct Broadcast: Det er der, hvor vi inde i Netværk A, men sender alt data ud til Netværk B's Devices.'
- Multicast: Det er der, hvor man sender data ud i grupperet form.
  - Eksempelvis er Nyhedskanaler et eksempel på Multicast.
  - Grupper på sociale medier, kan være eksempel på Multicast fordi man "targetter" disse grupper.

ARP-Request

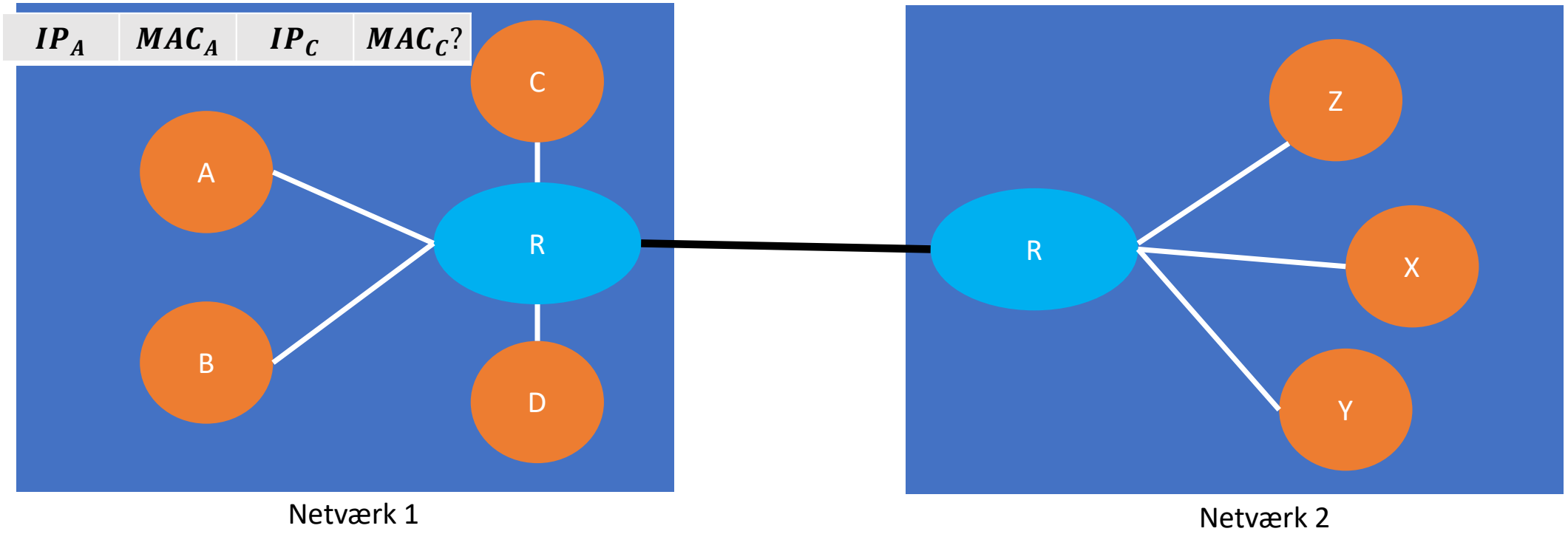
Vi kommer til at introducere nu til Address Resolution Protocol.

ARP-Request

- Vi har egentligt i de sidste slide snakket omkring, hvordan en Netværk spørger om MAC-Adressen hos den anden Netværk. Det er det som vi nærmere skal kigge på her.
- Det som vi egentligt gøre her er, at vi konverterer den logiske adresse til den fysiske adresse.
  - ARP-Protokollen tilhører niveau 3, når man snakker om OSI eller TCP/IP Modellen.

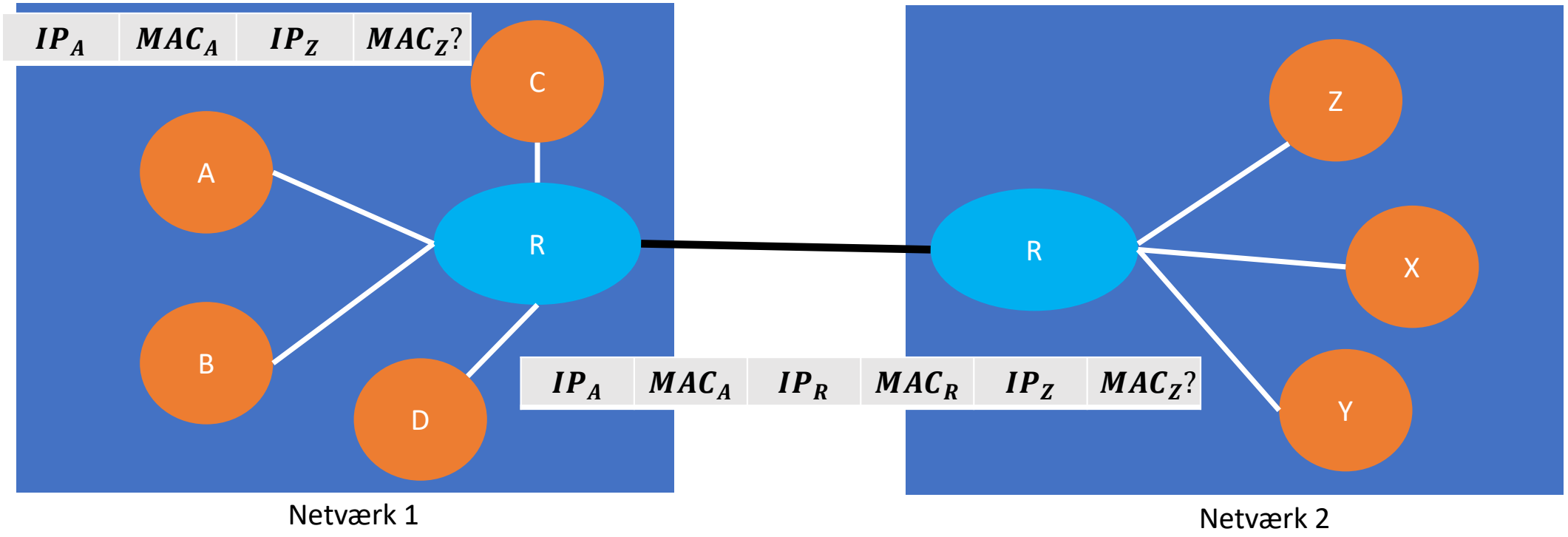
ARP-Request

- Her viser vi tilfældet hvor Device A ønsker at kontakte Device C indenfor Netværk A (Unicast).
  - Netværk A sender en besked til Routeren, med dets IP og MAC-adresse og Device C's IP-adresse.
  - Herefter modtager Device C oplysningerne og derefter sender dets MAC-adresse tilbage til Device A gennem Routeren.



ARP-Request

- Her viser vi tilfældet, hvor Device A kontakter fra Netværk 1 til Device Z i Netværk 2.
  - Device A sender dets IP-adresse og MAC-adresse til Routeren sammen. (Den kender ikke INFO om Z).
  - Routeren modtager oplysningerne og indsender sit INFO til Routeren i Netværk 2. Netværk 2 sender dets MAC-adresse tilbage.
  - Device A INFO er opsamlet hos Router ved Netværk 2, og dette sendes videre til Device Z som så sender sit MAC-adresse tilbage.

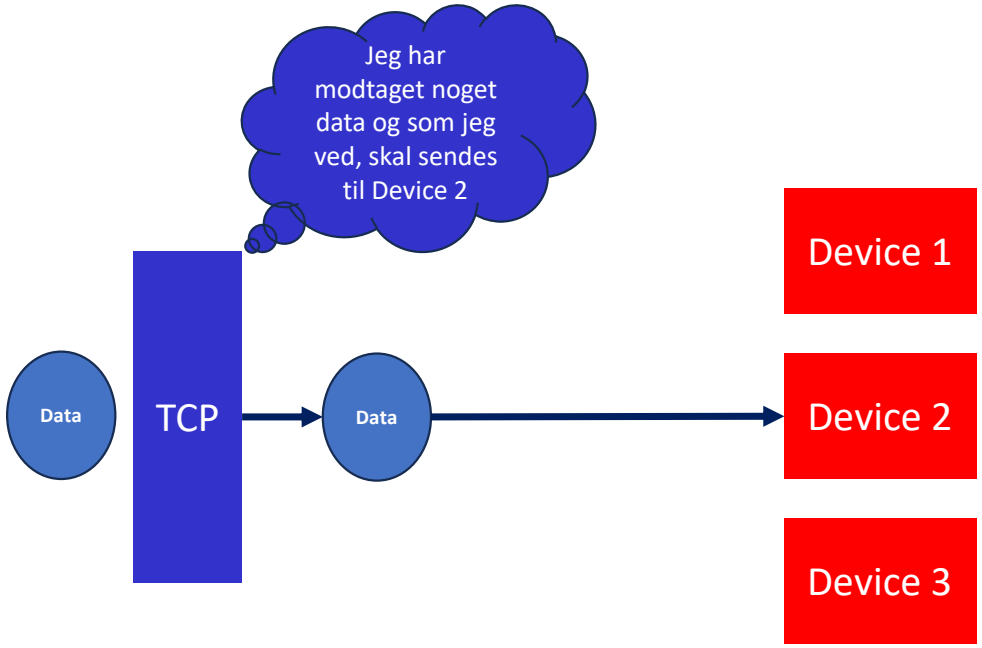


TCP & UDP

Her kommer vi til at introducere til forskellen mellem dem.

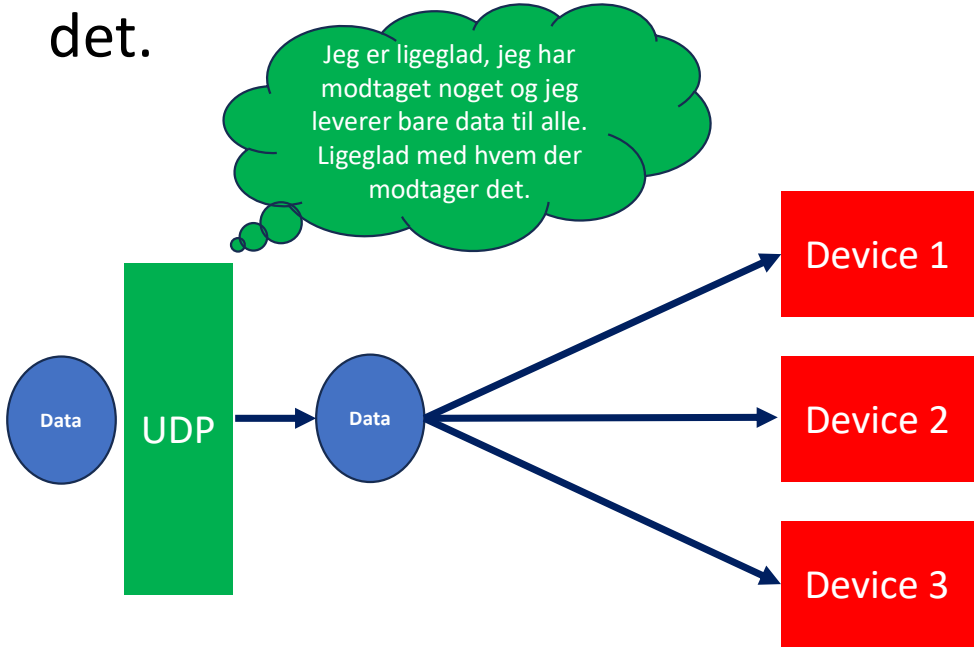
TCP

- TCP eller Transmission Control Protocol er der, hvor TCP modtager noget data og derefter danner forbindelse til et andet Netværk eller Enhed og sørger for at den er leveret.



UDP

- UDP eller User Datagram Protocol er der, hvor UDP modtager noget data og derefter ”broadcaster” til alle Netværker og Enheder derude, uden at tage hensyn til hvem der modtager det.



TCP, DNS & HTTP

Vi kommer i denne afsnit til at introducere til lidt flere detaljer om TCP.

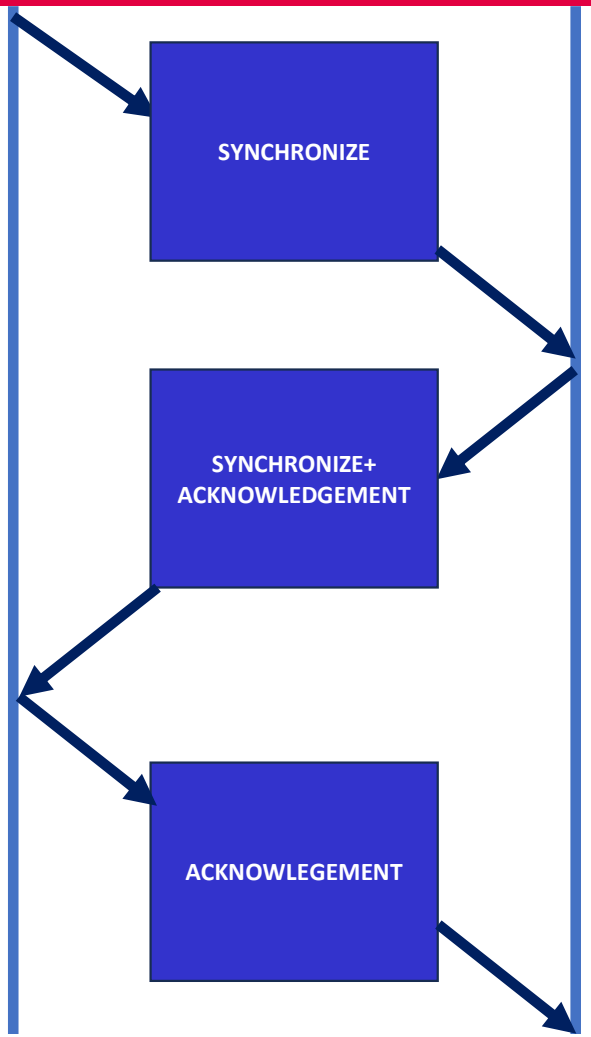
Så introducerer vi også DNS som er Domain Name Server og HTTP-Protokollen.

TCP i lidt detaljer

- Byte Streaming: Vi kan se, at når vi modtager noget data så forsøger vi at inddele det i segmenter af bytes.
- Connection Oriented: Her findes reliabilitet, som betyder at det data som afsendes, skal modtages henne ved modtageren i sidste ende uden afbrydelser. Sker der afbrydelser, skal data sendes igen på den korrekte måde.
  - 3-WAY HANDSHAKE ACKNOWLEDGEMENT.
- Full Duplex: Når Handshake er blevet etableret, så kan afsenderen også være istand til at modtage data på samme tid som den afsender.
- Piggybacking: Når dataoverførslen er færdig, siger vi ”færdig” på terminalen. Det betyder, at en Anerkendelse er sendt med noget data.
- Error-Control: Hvis den afsendte data er blevet ændret, så skal modtageren vide fra TCP at der findes Ændringer i data i form af Error.
- Flow-Control: VI skal sørge for, at holde en god flow når data afsendes. Fordi eller skaber det problemer i modtagerens modtagelseskapacitet. Dette gør, at modtageren måske ikke kan håndtere data ordentligt og at dataet nok sandsynligvis går tabt.
- Congestion Control: Ligesom man tager hensyn til Kapacitet hos Modtageren, så tager man også hensyn til Netværkstrafik således at der ikke opstår Congestion i dets Kapacitet.
- **PÅ NÆSTE SIDE VISER, HVORDAN 3-WAY HANDSHAKING METODE ER GENSKABT!**

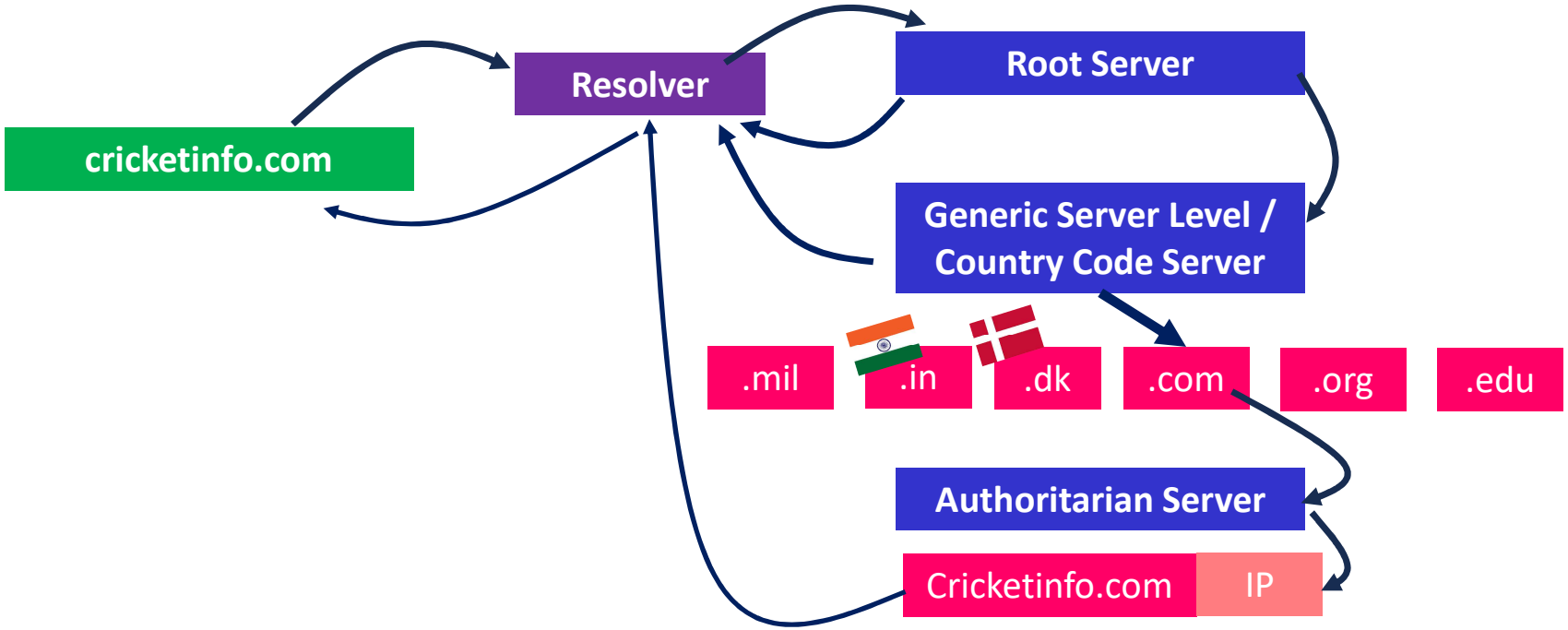
TCP – 3-WAY HANDSHAKING METHOD

- Når vi plejer, at sende data gennem TCP, så plejer den at lave forbindelsesetablering som er 3-vejs håndgivning.
  - Vi kan se, at den første er det som man kalder for Forbindelsesetableringsfasen. Vi kan se, at Klienten sender en Synkronisering.
  - Synkronisering betyder, at man danner en forbindelse til Serveren.
  - Så sender Serveren en Acknowledgement og en Synkronisering tilbage, så den også etablere en forbindelse tilbage.
  - Så modtager Klient Synkronisering og Acknowledgement og dermed sender en Acknowledgement og siger at forbindelsen er godtaget og vi starter samtalen.



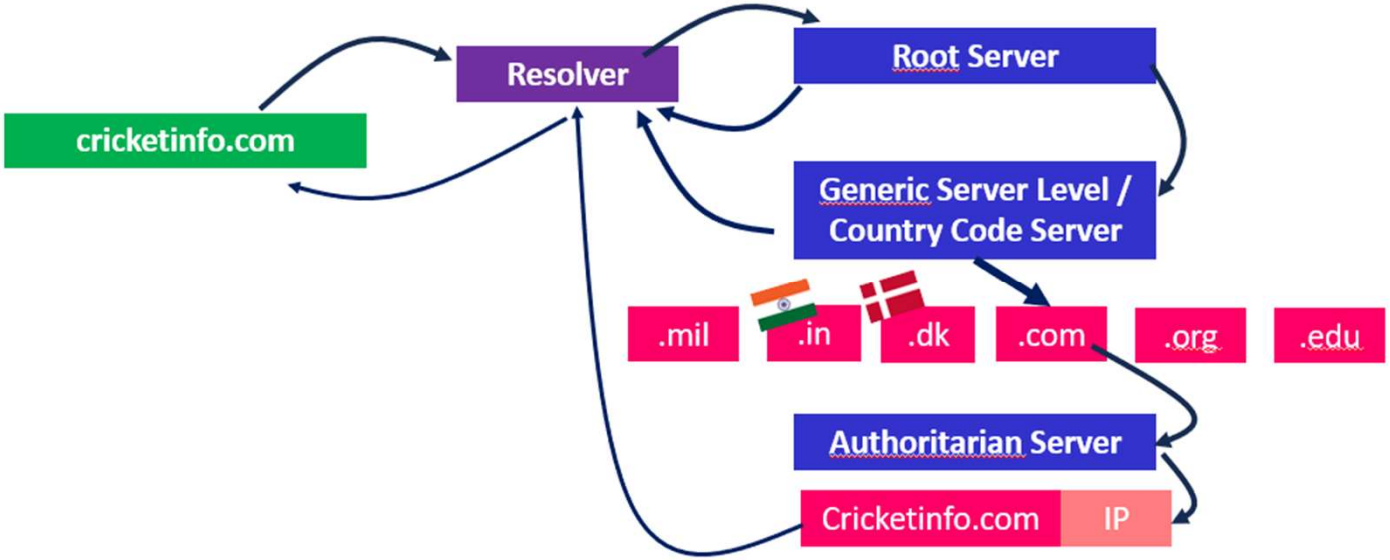
DNS i lidt detaljer

- **Her forklarer vi, hvordan Domæne Navnsystemet virker.**
  - **Vi kan se, at en generelt mennesker bruger domæne navn til at surfe på en webside.**
  - **Ved brug af domænenavne har brugeren nemmere ved at huske linket til websiden, hvor IP-adresser med tal er meget svære at huske.**
- **Vi kommer her til at tage et udgangspunkt i en webside med domæne: cricketinfo.com**



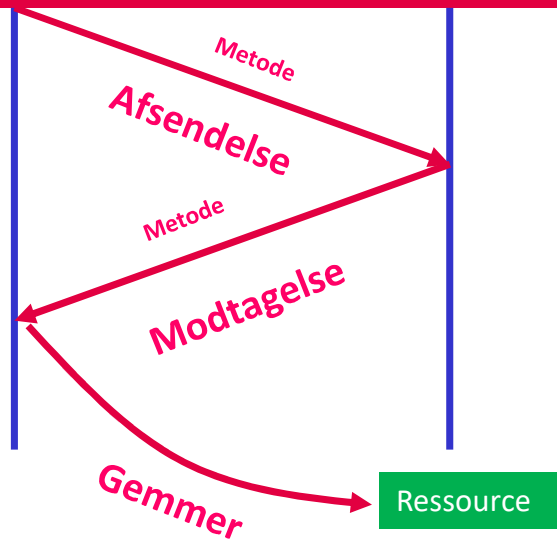
DNS i lidt detaljer

- **Set på sidste side, så kan det tydeligt ses at vi har en webside ved navnet cricketinfo.com**
- **Denne info sender en Request til Root-Server gennem Resolver som er vores tjenestemand, der gør arbejdet.**
- **Så sender Root Server anmodningen videre til Country code, hvor vi så får udleveret en country-code til vores webside.**
  - **Denne country-code bliver sendt til Resolveren, mens anmodningen går videre til Autoritative Server.**
- **Så bliver anmodningen sendt videre til Autoritative Server, som udleverer en fikst IP-adresse til os.**
  - **Denne IP-adresse bliver sendt til Resolveren, som betyder domæne navnet får en IP-adresse og som bliver sendt til cricketinfo.com**



HTTP i lidt detaljer

- **HTTP er en Protokol som brugs i sammenhæng med TCP, hvor man afgiver og modtager data mellem afsender og modtager.**
- **Vi kan se, at hensigten med HTTP er at kunne danne en kommunikation mellem to maskiner, hvor den ene spørger om noget information og den anden giver response på informationen.**
- **Oftest er det sådan, at når man ønsker noget respons på noget information, så har man en metode der fortæller hvordan informationen skal bearbejdes efterfølgende.**
- **Der hvor man bearbejder eller gemmer selve informationen er kaldt for ressource.**
- **Headers kan være eksempel på de protokoller som man ønsker at snakke i.**



FLERE INFORMATIONER

Nu introducerer vi til nogle facts for Autonome Systemer osv.

Autonome Systemer i lidt detaljer

- **Autonome Systemer, er der hvor vi snakker om klynger af kæmpe store netværker.**
- **Vi kan se, at klynger af de forskellige netværker som har interesse i netværks sfæren.**
- **Og alle disse klynger er med til at udgøre hele netværket.**
- **Routing mellem de forskellige autonome systemer er, at man har listen af de forskellige ip-adresse som autonome systemer ejer.**
- **Her forklarer man, hvordan de forskellige ting er forbundet med hinanden.**

RIP og OSPEF i lidt detaljer

RIP:

- **Den første Information er kaldt for RIP, hvor man inde i et autonome system ønsker at forbinde/danne den korteste forbindelse.**

OSPEF:

- **Det betyder, at man ønsker at danne en kort rute over antal af maskiner som man skal til.**
- **Så har man en OSPEF, hvor man ønsker at finde den hurtigste og korteste rute.**

BGP:

- **Så kommer vi til, hvordan vi finder den hurtigste og korteste forbindelse inde i en klynge. Vi starter med, at se hvordan kommunikationen er.**
- **Her kan det ses, at vi har BGP, hvor hvert firkant kendetegner deres eget autonome systemer.**
- **Vi kan se, at det kendetegner to ting, hvor den ene er hvilken autonome system (ip-adresser) ejer jeg, og den anden er hvilken autonome systemer er man forbundet til.**

SLUT 7

Lavet af: Vivek Misra