Kommunikationssystem – Lektionsanteckningar

4/9 – Föreläsning 2

Kapitel 2 och 3 – Information och bitar, och signaler som skickas på en länk

Binärdata är information som datorer kan lagra, i form av ettor och nollor. Text, ljud och bild måste omvandlas till binärdata för att en dator skall kunna förstå. Olika regler bestämmer hur datan ska lagras, så kallade protokoll. Tanken bakom är att informationen ska begränsas till värden, som sedan kan representeras av ett binärt tal. En **bit** är en siffra och åtta siffror kallas för en **byte**. 1024 bytes (=) är en kilobyte.

Ex: 8 bitar (en byte) = 1001 1011

Omvandling av vår information till att datorn lagrar informationen kallas för digitalisering. Ett sätt kallas ASCII.

2.2 Digitalisering

Text, ljud, bild och video måste allt göras om till binärdata för att en dator skall kunna förstå informationen. **Text** är ganska simpelt att omvandla eftersom den är uppbyggd av en begränsad mängd data. Man kan säga att texten redan är digitaliserad. ASCII är en vanlig kodningsstandard för text där varje skrivtecken representeras av en sjubitarskod. En kodningsstandard som också är vanlig heter UTF-8 och är den dominerande teckenkoden för world wide web och de senaste Linux-distributionerna.

Ljud kan ses som en som signal som varierar i amplitud. Ljudet måste därför diskretiseras. Denna omvandling sker i tre steg:

- **Sampling**

Ljud mäts i Hertz (Hz), och Hz talar om vilka frekvenser en kontinuerlig signal har. Innehåller en signal frekvenser mellan 800 Hz – 1 000 Hz betyder det att bandbredden är 200 Hz.

Vad man gör vid sampling är helt enkelt att man mäter signalen så ofta att den högsta frekvensen kommer med. Detta gör man för att kunna återskapa den senare. Samplar man 1 gång per sekund är samplingsfrekvensen 1 Hz.

- **Kvantisering**

I Kvantiserinssteget försöker man att minska felmarginalen. På något ställe i proceduren kommer man behöva avrunda, och för att få minsta möjliga avrundningsfel (kvantiseringsfel) så gäller det att kvantiseringssteget är tillräckligt litet, annars blir inte signalen återskapad tillräckligt bra.

Man avrundar varje mätvärde från samplingen till en specifik amplitudnivå för att kunna representera varje nivå med ett binärt tal. För att kvantiseringsfelet ska bli så litet som möjligt skall steget mellan varje amplitudnivå vara så lite som möjligt.

- **Kodning**

När man kodar lagrar man signalen i binär form. Varje amplitudnivå har nu blivit representerat av ett binärt tal. Kodar man signalen med 8 bitar betyder det att det finns 256 () möjliga amplitudnivåer.

Telefoni: 8 bitar – 256 nivåer

CD: 16 bitar – 65 536 nivåer

2.3–4 Från information till binärdata och datakomprimering

Att omvandla **bild** fungerar med samma procedur som ljud. Bilden måste samplas på något sätt innan den kan omvandlas till binärdata. Man delar då upp bilden i små bildelement som man kallar för pixlar. Varje pixel representeras av ett bildvärde. Ju fler pixlar – ju bättre upplösningen. Om man bilden skall vara i färg innehåller varje bildvärde en kombination av färgerna blå, grön och röd. Detta gör att varje bildvärde består av tre komponenter.

Video kodas på samma sätt som bilder, eftersom en filmsekvens är ett antal bilder efter varandra. Bildfrekvensen (frame rate) är antalet bilder per sekund. Varje bild består av en vertikal respektive horisontell uppdelning vilka kallas linjer respektive pixlar. Bildupplösningen är antalet pixlar per linje och antalet linjer per bild.

Datakomprimering är en idé för att utnyttja överföringskapacitet bättre och ta bort redundant (onödig) information. Mindre data – samma info.

Överföringshastighet – bitar per sekund.

3.1–2 Digital kommunikation

Vid **digital transmission** läser mottagaren av spänningsnivån (eller energinivån) på länken vid olika tidpunkter, och kan då se vilka bitar (ettor och nollor) som har skickats. Exempel på denna teknik (linjekodning):

NRZ – Här låter man ettor och nollor vara olika amplitudnivåer. Oftast motsvarar en nolla en positiv spänningsnivå och en etta motsvarar en negativ spänningsnivå. Denna metod är väldigt enkel, men kan skapa problem då flera nollor eller ettor skickas samtidigt. Spänningsnivån kommer att vara konstant och det kan bli problem vid synkroniseringen mellan sändare och mottagare då den senare inte vet när nästa bit kommer.

Manchester – Denna metod löser problemet med NRZ genom att vid varje bitövergång byta amplitudnivå. En nolla motsvarar en övergång från positiv till negativ spänningsnivå, och skulle flera nollor komma på rad byter spänningen amplitud för att sedan skifta till negativ igen. En etta motsvarar tvärtom, en övergång mellan negativ till positiv spänningsnivå, och fungerar på samma sätt. Problemet med manchesterkodning är att den ställer högre krav på användarutrustning, då signalen har dubbelt så hög frekvens som vid NRZ. Mottagaren kan också ha svårt att hinna med om signalen byter amplitud för ofta.

Differential Manchester – Fungerar som manchesterkodning, att signalen inverteras i tidsintervall. Skillnaden är att i den här kodningen byter signalen spänningsnivå även i början av tidsintervallet. Skulle den byta nivå är det en nolla, medan om den är kvar på samma är det en etta. Man kan tro att det inte är någon skillnad mellan denna kodning och den tidigare, men differential manchester-kodning har faktiskt lägre frekvens när den byter tillstånd.

Vid **analog transmission** låter man sinusvågar variera och variationen berättar om det är en etta eller en nolla som beskrivs. Vågen beror på tre storheter – fasmodulering, frekvensmodulering och amplitudmodulering, där amplitudmodulering är störningskänslig, frekvensmodulering är bandbreddsbegränsat och fasmodulering kräver att mottagaren ska kunna identifiera små skillnader.

3.3–5 Analog kommunikation

bruh

3.6 Multiplexering

All information som man skickar över en **länk** når samtliga datorer inkopplade på länken. Informationen har alltid en begränsad storlek då en signal som skickas över en länk både dämpas efter hand, men också tar tid på sig att ta sig till andra änden av länken. Signalen kan förstärkas genom att man bygger in en **repeterare** på länken. Denna återskapar signalen (vilket man hör på namnet), men löser inte tidsproblemet. När två datorer skall utbyta data över en

länk sätter de först upp en regel om i vilken riktning datan skall ta. Simplex betyder att bara en datorerna skickar data, halv duplex betyder att signaler kan färdas i båda riktningar, men bara en väg i taget, och om **full duplex** råder kan signaler färdas i båda riktningar samtidigt. För att inte signalerna ska kollidera under full duplex kan man dela upp en länks **kapacitet** i olika multiplex (kallas mulitplexering):

1. Rumsmultiplex
2. Frekvensmultiplex
3. Tidsmultiplex
    - Synkron
    - Statistisk
4. Koduppdelad multiplexering

Idén är att man skall utnyttja länkens kapacitet till fullo och vad man gör är att man helt enkelt delar in länken i olika logiska vägar, som kallas **kanaler**. Båda datorerna kan nu utnyttja olika kanaler, och de logiska vägarna gör så att signalerna som skickas inte krockar med varandra. Länken (kanalen) har en viss bandbredd, vilket gör att den bara kan skicka signaler över ett visst antal frekvenser. Antingen är länken fysisk, vilket betyder att den sitter direkt mellan sändaren och mottagaren, eller logisk. En logisk kanal tilldelar frekvenser till en viss överföring. På så sätt kan flera sändar-mottagar-par använda länken utan att information krockar.

Eftersom det finns flera datorer på en länk, behöver det finnas ett sätt för sändaren att nå fram till rätt mottagare. Varje dator har därför en egen **adress**. När sändaren skickat paketet över en länk når paketet fram till alla datorer, men bara datorn med rätt adress läser in paketet. De andra datorerna kastar det.

6/9 – Föreläsning 3

Kapitel 4 – Tillförlitlig dataöverföring

4.1–2 Protokoll och Feldetektering

I varje paket har man ett sätt för mottagaren att kunna kontrollera om allt i paketet har kommit med. Man upptäcker helt enkelt **bitfel** genom att lägga till extra bitar. Felaktiga paket kastas.

Det finns några olika sätt. Ett sätt är att sändare och mottagare kommer överens om:

- Jämnt antal ettor – jämn paritet

- Ojämnt antal ettor – ojämnt antal nollor

Till exempel har de bestämt sig för jämn paritet. Det betyder att om det är ett jämnt antal ettor i paketet lägger sändaren till en nolla. Eller om det är ojämnt antal ettor läggs en etta till i slutet. Mottagaren vet då att sista siffran är en så kallad **paritetsbit**. Enda problemet med denna metoden är att mottagaren inte märker om det har blivit ett jämnt antal fel.

Ett annat sätt är att man använder sig av en **kontrollsumma**. Detta upptäcker fler fel än en paritetsbit, men tar längre tid att utföra. Det går till så att man delar upp bitströmmen i större segment, till exempel 8 bitar per segment. Man summerar (adderar) segmenten och får allt som oftast några bitar större än de tidigare segmenten. Dessa överskjutande bitar läggs till och skapar en ny bitström som läggs till på slutet av paketet. När mottagaren tar emot bitströmmen skall komplementet av den nya bitströmmen vara = 0. Då är bitströmmen korrekt.

Den absolut vanligaste kontrollen man använder sig av kallas **CRC**. Ettorna i paketet översätts till x, som bildar ett polynom. Sändaren utför en polynomdivision med hjälp av polynomet och ett generatorpolynom. Paketet skickas och mottagaren översätter polynomet. En ny polynomdivision utförs med översättningen av polynomet och generatorpolynomet. Resten skall = 0.

4.3 Felhantering

Efter att man skickat ett paket är grundprincipen att mottagaren **bekräftar** att paketet kommit fram. Det finns två metoder som oftast används; Stop-and-wait och Go-back-n.

Stop-and-wait fungerar så att efter att man skickat ett paket, väntar man på en bekräftelse innan man skickar nästa. Fördelen är att det är en simpel metod och lätt att göra, men det tar ganska lång tid. Ett effektivare sätt som tar vara på tiden är **Go-back-n**. Skillnaden mellan metoderna är att man i go-back-n skickar flera paket samtidigt. Antalet paket som kan skickas bestäms av ett sändfönster. Varje paket får en egen klocka, och om det inte kommer ett ACK inom tidsramen skickas paketet igen. Mottagaren skickar ACK i rätt ordning som de får paketen, beroende på vad ACK säger flyttas sedan ”fönstret” så att sändaren kan fortsätta skicka nästa paket i listan.

9/9 – Föreläsning 4

Kapitel 5 – Lokala nät, Ethernet, stora nät

5.1–3 Accessmetoder och större länkar

LAN (Local Area Network) kan vara sammankopplade med **bryggor**, som förstår **länkprotokoll**. Länkprotokollet ser till så att det blir enkelt för mottagaren att se vart varje paket ligger i bitströmmen som skickas från **nätadaptern**. Sändaren använder en applikation som skickar ett brevliknande paket till **länkhanteraren**. Länkprotokollet hjälper länkhanteraren att rama in varje paket med flaggor, som är lättare för mottagaren att detektera. Det är i länkprotokollet man brukar använda ordet **ram** istället för paket, då flaggorna ”ramar in” paketet. Länkhanteraren skickar vidare datan till nätadaptern som skickar över bitströmmen via den fysiska länken till mottagaren. Mottagarens länkprotokoll ser direkt med hjälp av flaggorna de olika paketen. Flaggorna är en del av vad man kallar för pakethuvudet och paketsvansen.

På varje lokalt nät (och kanske mer) har man en **accessmetod** för att datorerna ska veta i vilken ordning och hur de ska skicka sina paket.

- **Polling** à En master bestämmer när och i vilken ordning de andra datorerna får skicka data. Datorerna kallas för slavar. Inte speciellt effektiv metod, speciellt inte på större skala.

- **ALOHA** à Två gemensamma kanaler är uppsatta till och från en centraldator. En **klient** (dator) kan skicka all sin data på samma gång, och vänta på ACK. Väldigt effektivt vid lite trafik. Totalt 2 gånger sändningstiden och chans till kollision. (18%)

- **Slotted ALOHA** à Fungerar precis som ALOHA, men uppdelat i tidsintervall. Detta minskar chansen för kollision och sändningstiden är hälften så stor. (36%)

- **CSMA/CD** à En väldigt vanligt förekommande accessmetod. Datorn väntar och lyssnar på länken först. Är det ledigt, skickar den sitt paket. Är länken upptagen väntar man tills den är ledig (först till kvarn). CD (Collision Detection) betyder att efter att paketet har skickats lyssnar datorn på sin egen signal. Märker man att signalen inte fungerar avbryter man direkt, så att signalen inte kolliderar med en annan signal.

- **Token Ring** à En dator har en token, skickar sitt paket, och lämnar sedan vidare token. En äldre accessmetod som användes oftare förr.

5.4–5 IEEE 802.x och Ethernet 802.3

LLC och **MAC** är två exempel på länkhanterare. De har olika protokoll för att hantera länken. LLC-protokollet ansvarar för den ”logiska vägen” mellan sändare och mottagare. MAC-protokollet håller koll på adresseringen i ett lokalt nät, och tillgång till mediet som används (till exempel en kabel). MAC-adresser består av 48 bitar och skrivs på hexadecimal. Används därför att det är kortare att skriva än binära tal. **Exempel**: 00:00:0C:1A:E4:BD

Sedan 1985 finns det olika standarder för lokala nät, och ett av dem kallas för **Ethernet**. Ethernet började användas som standard redan 1976, och det vi använder idag kallas för

Ethernet II. Internationella standard (**IEEE 802.x**) bygger på Ethernet. Vad IEEE-standarder är kända för är att dela upp länkhanteraren i två skift (LLC/MAC). Det som skiljer IEEE och Ethernet II åt är att ramarna ser lite annorlunda ut. I 802.3 (IEEE) återfinns ett längdfält som talar om hur mycket data som skickas i ramen (paketet). På motsvarande plats i Ethernetramen finns ett typfält, som berättar vilket typ av protokoll ramen för över.

När man bygger ett LAN med Ethernet som standard använder man en **Ethernet-switch** i centrum. Ethernet-switchen arbetar på **länknivå**, och ser till att rätt data kommer till rätt mottagare med hjälp av MAC-adresser. Switchen ser även till att paketen inte kolliderar. Switchen är nödvändig för att kunna använda högre överföringshastigheter i nätet. Varje bildar ett ”eget” litet LAN tillsammans med switchen (egen kollisionsdomän). **Fast**-, **Giga**- och **10Gigabit** Ethernet är några olika exempel på utvecklade Ethernet-standarder. Man vill helt enkelt öka överföringshastigheten. Fast- och Giga Ethernet fungerar både på optisk fiberkabel och tvinnad parkabel. 10Giga Ethernet fungerar enbart på fiberkabel.

Bryggan, som är sammankopplar LANet, förstår länkprotokollet. Den har en adressbok som berättar vilken dator som är kopplad till vilken länk. Bryggan tillhöger **lager** 3 (2?), som kallas internetlagret. Alla enheter på samma länk ingår i en **kollisionsdomän**. Att kunna nå alla enheter på en länk kallas att **broadcasta**. En **broadcast domän** är alla enheter som kan nås av samma meddelande på lager 2, länklagret. Här ingår bryggor, länkar och repeterare i samma LAN. Alla LAN har samma länkprotokoll.

Exempel på olika **stora nät** är SONET och ATM. SONET är till för telefoni och överföringshastigheten är 64 kbps. ATM använde sig av föregångaren till IP, alltså ett internetprotokoll. Alla paketen hade samma storlek och det var ett paketförmedlande nät.

Vägväljare skiljer ett LAN från ett annat LAN. Det enda som vägväljarna gör är att de får paket skickats till sig, och skickar vidare till rätt adress.

16/9 – Föreläsning 5 och 6

Kapitel 6–7 – ARP/NDP, TCP, IPv4/IPv6 (ICMP)

6.1–2 Vägväljare och nätarkitektur

Det finns olika sätt för vägväljaren på olika OSI-nivåer att hitta en lämplig väg paketet. Ett sätt är att ta reda på den kortaste vägen mellan noder. Detta kallas **least-hop-path**. Ett annat sätt är att ta reda på den väg som kostar minst. En kostnad kan bero på olika saker. En förbindelse som utnyttjas av många kan ha en hög kostnad, samtidigt som en längre länk kan ha en högre kostnad**. Least-cost-path** tar den väg som kostar minst. **Flooding** är ett tredje sätt som används när vägväljaren inte vet vart adressen finns. Då sätts en TTL upp (för att inte paketet

skall hoppa runt hur länge som helst) och paketet skickas ut till alla länkar. Paketet kastas sedan när TTL är slut.

Vägväljare skiljer ett LAN från ett annat LAN. Det enda som vägväljarna gör är att de får paket skickats till sig, och skickar vidare till rätt adress.

6.3 Kommunikation över fler nät

Nätarkitektur menar hur olika kommunikationsnät är uppbyggda och hur de är sammankopplade. Hur transporten av data fungerar mellan en sändare och en mottagare. Grundläggande brukar man prata om två olika slags nät: **accessnät** och **stamnät**. Alla värddatorer är kopplade till ett accessnät, som ger datorn ”access” till nätet. Detta är ofta ett LAN. LAN är sedan anslutna till stamnät, som sammankopplar näten. Detta stamnät kan i sin tur vara accessnät till ett annat stamnät.

Olika typer av nät:

- **LAN**

Här återfinns noder som bryggor, länkar och repeterare.

- **MAN –** Metropolian Area Network (stad)

- **WAN –** Wide Area Network, större än MAN.

För att olika standarder skall kunna kommunicera med varandra behövs ett gemensamt protokoll. Detta protokoll är **nätprotokollet**. Vägväljare sitter på nätnivån (lager 3) och nätprotokollets uppgift är att ge vägväljaren möjlighet att bestämma vilket nät som ramen skall skickas till. Ett exempel är Internet Protocol (IP).

Transportprotokollet ska underlätta för alla applikationer, men också skapandet av nya applikationer. Nya applikationer tar inte lika lång tid att utveckla nuförtiden då **ISO**-modellen för protokoll redan är byggd. Applikationsprotokollet är egentligen det enda som behöver skapas. Transportprotokollet hjälper applikationerna med felhantering och underlättar förbindelser med andra applikationer. Detta gör det genom portadresser, som berättar för paketen precis till vilken port som för dem till vilken applikation. De flesta applikationer har en **standardport**. När en applikation vill starta en dialog skickar den ett meddelande till mottagarapplikationens standardport. Portar adresserar vägar för att flera applikationer skall kunna fungera samtidigt. Vissa är fördefinierade och hjälper till exempel mejl att gå till mejlapplikationen direkt.

I transportprotokollet kallas paket för **segment**.

Två exempel på ett transportprotokoll är Transmission Control Protocol (**TCP**) och User Datagram Protocol (**UDP**). TCP är förbindelseorienterat, vilket är en förlitlig förbindelse. Alla paket kommer fram garanterat, felhantering och kontroll sker i dialogen mellan sändare och mottagare, vilket gör att alla paket kommer fram till slut. UDP är ett exempel på ett förbindelsefritt transportprotokoll. Mindre väntan och ingen dialog med bekräftelser (ACK) sker, vilket gör att dataöverföringar går snabbare. Nackdelen är att här inte finns någon felhantering eller kontroll, vilket gör att många paket inte når fram. Vissa applikationer kräver inte att alla paket kommer fram korrekt, vilket gör att ett förbindelsefritt protokoll är mest effektivt (Skype).

När data ska levereras över flera nät är det över tre generella nivåer. Länkprotokollet ansvarar för att data levereras till rätt nod inom ett nätverk (**node to node**). Länkprotokollet samarbetar med nätprotokollet, som i sin tur ansvarar för att datan ska ta sig till rätt värddator (**host to host**). Väl framme hos rätt värd ansvarar transportprotokollet för att rätt applikation får rätt data, med sina portnummer (**process to process**). Det behövs alltså tre adresser för att datapaket ska hitta rätt.

6.4 Protokollskikt

Alla protokoll som används vid en datorkommunikation arbetar alltså för sig själva, utan att veta vad de andra protokollen gör. Detta ser utarbetade gränssnitt till. Tänker man att kommunikationen sker i flera steg, så har man delat in dessa steg i skikt (**layer=skikt**). Mellan varje skiktövergång läggs det på ett för skiktet eget pakethuvud. Protokollen behandlar alltså de andra protokollens headers som ren data. När de tar emot ett paket tar de sedan bort sitt eget huvud och skickar vidare, beroende på vad det står i huvudet.

Skulle ett skikt få för mycket information från ett annat skikt, sker något som kallas för **segmentering**. Det betyder att ett skikt delar upp paketet i mindre segment, och sätter på ett likadant huvud på varje segment. Segment kallas även för Protocol data unit (**PDU**). Till exempel kan transportprotokollet ta emot information från en applikation, och inse att det är för mycket data för att skickas i ett *datagram*. Datan delas upp i *segment* och en transportheader sätts på varje segment. Nästa stopp är nätprotokollet, som tar emot segmenten och inser att det ena segmentet måste delas upp i mindre *paket*. Nätheadern sätts på alla tre paket och skickar vidare till länkprotokollet. Här ser det sista protokollet, i det lägsta skiktet, att paketen kan skickas vidare utan att segmenteras mer. Paketen ramas in med flaggor, bildar *ramar*, får en länkheader och skickas ut på länken. Under hela processen får de olika slags paketen siffror så att mottagaren vet i vilken ordning PDU:er ska sitta ihop. Varje skikt ansvarar för att foga samman rätt uppdelning, och skicka vidare rätt information i rätt ordning.

OSI-modellen beskriver en referensmodell för hur kommunikationsnät bör vara uppbyggda. Varje skikt består av en eller flera standardiserade protokoll.

1. **Fysiska skitet** innehåller regler för hur en bitström representeras över ett transmissionsmedium (länk). Ser även till att sändare och mottagare är synkroniserade, och att överföringshastigheten blir rätt för länken och noder. På OSI-nivå 1 kan man säga att repeterarna arbetar.

2. **Länkskiktet** ansvarar för node-to-node, det vill säga dataöverföring mellan två noder. Här finns alltså information om adresser till noder på en länk, men också tillförlitliga funktioner som flödeskontroll på länken och felhantering av ramar. Har även regler för hur datan paketeras i ramar, så att mottagaren förstår informationen. På OSI-nivå 2 arbetar switchar. De är intelligentare än repeterare och har flera portar. Kommunicerar inom ett LAN.

3. **Nätskiktet** ansvarar för host-to-host, alltså att datan tar sig mellan två värddator, oavsett vilket nät mottagaren tillhör. För att kunna ta sig genom nät finns här också funktioner för att hitta den lämpligaste vägen. På OSI-nivå 3 hittar man routern. Det är en vägväljare som kopplar samman flera nät. Notera att olika nät kan arbeta enligt olika protokoll på OSI-nivå 1 och 2, men på tredje måste näten ha samma protokoll för att kunna samarbeta. Broadcast som skickas på ett nät kommer inte längre än till routern, som utgör gränsen (broadcast kan förekomma på OSI-nivå 3 också). Detta kallas för en broadcastdomän.

4. **Transportskiktet** ansvarar för process-to-process. Detta betyder att OSI-nivå 4 ska skicka rätt information till rätt applikation. Därför finns det funktioner för flödeskontroll, felhantering och adressering. Adresseringen fungerar med hjälp av portnummer till applikationerna. Transportskiktet skall också kunna dela upp information i segment ifall det kommer för mycket information från en applikation. Detta skall även kunna de-segmenteras. Om ett förbindelseorienterat protokoll används måste skiktet även kunna sätta upp, underhålla och stänga ner en förbindelse. OSI-nivå 4:s syfte är att flera applikationer skall kunna vara igång samtidigt. På denna nivå och högre arbetar gateways, som kopplar samman nät och översätter mellan olika protokoll och applikationer

5. **Sessionsskiktet** – Ser till att synkroniseringen mellan datorer fungerar, hur dialogen sätts upp och underhålls. Finns regler för hur datorerna sänder till varandra (vem som sänder).

6. **Presentationsskiktet** – Här kodas informationen som kommer från applikationen till binärdata. Finns funktioner för komprimering av data samt kryptering.

7. **Applikationsskiktet** – Det som användaren, som kan vara människa eller programvara, ser. Användaren ska kunna komma åt nätet. Innehåller användargränssnitt.

7.1–2 TCP/IP-modellen och dess nätstruktur

Internet består av flera olika nät, som är sammankopplade. Det enda som alla har gemensamt är att de har samma nätprotokoll, Internet Protocol (**IP**). För att kommunicera med varandra behöver alla enheter en unik adress, dessa kallas IP-adresser för IP.

Internets protokollmodell kallas för **TCP/IP-modellen** och utvecklades oberoende av OSI-modellen. Om man jämför dem så skiljer det sig speciellt på två punkter. Skikt 5–7 av OSI-modellen är endast ett skikt i TCP/IP-modellen. Denna delen kallas för applikationsskiktet. Under OSI-nivå 3 definieras inget speciellt för TCP/IP-modellen, utan här kan varje nätoperatör bestämma själv vilka datanät som skall användas för Internetöverföringen. Internet använder sig av **förbindelsefri dataöverföring**, vilket betyder att det inte finns några garantier att paketen kommer fram eller i rätt ordning. Mottagaradressen är det enda som styr hur routrarna i olika nät skall skicka paketet, då överföringen mellan sändare och mottagare är oberoende av varandra. Detta kallas för ”**best effort**”.

Internet har inget nät som leds, drivs och underhålls av en organisation, utan Internet är alla **sammankopplade nät** som använder sig av TCP/IP för transport av data. Det finns ingen organisation som driver Internet, men det finns en viss global hierarki. Internet Service Provider (**ISP**) levererar vad man kallar för **internetaccess** till nationella och regionala operatörer, som i sin tur levererar internetaccess till företag, organisationer och lokala nät.

Nätsstrukturen för Internet delas in i **ryggradsnät** och **accessnät**, där små nät ofta har alla enheter kopplade till samma switch, och switchen i sin tur är ansluten via en uplink till företagets ISP. ISP ansluter nätet till en accessrouter. Större nät har istället flera accesnät kopplade till ett ryggradsnät, där ett accessnät kan bestå av flera routrar. **Full mesh** = alla är anslutna till varandra. **Internet Exchange** (IX) = en knytpunkt där man utbyter information.

Internetprotokoll består enbart av:

- Nätprotokoll: **Internet Protocol** v4 och v6
- Transportprotokoll: Transmission Control Protocol v4 och v6 (**TCP**), och User Datagram Protocol (**UDP**)
- På applikationsnivån finns en mängd olika protokoll, men de vanligaste är Hyper text transfer protocol (**http**), Simple mail transfer protocol (**smtp**) och File transfer protocol (**ftp**).

7.3 IPv4

Alla datorer och vägväljare måste under en dataöverföring använda samma version, annars måste man översätta mellan versionerna. Alla **IPv4**-adresser är utdelade, därav behövde man en ny version. Den nya versionen heter **IPv6**, och innehåller tillämpningar för realtidsapplikationer och stöd för autentisering och kryptering. Nätverksadresser är uppbyggda så att man vet var de finns.

En **IPv4-adress** består av 32 bitar. Skrivs som 4 decimala tal, där ett tal motsvarar en byte (8 bitar). **Nät-id** kommer först, sedan värd-id. Nät-id hjälper routrar att skicka paket till rätt nät.

Värd-id är unikt för varje dator. Routrar har **tabeller** med adresser för varje nät som är kopplat till routern, samtidigt som den har en tabell för värd-id på sitt eget nät. Routrar kan kommunicera med varandra och utbyta information som kan hjälpa en annan router att hitta rätt mottagare. Skulle en dator byta nät, måste den alltså byta IP-adress. Ett värd-id 0 refererar till nätet självt, medan ett värd-id med bara ettor innebär att paketet skall skickas inom det nät som specificeras i nät-id:t.

Mask – Se labbar (klasslös adressering).

Hade från början olika adressklasser, A, B och C. C har färre bitar till värd-id och A tvärtom. Detta gör att klass C passar bäst för någon som inte behöver många adresser. Värd-id:t för klass C har 8 bitar à (256) stycken adresser. Adresserna för C börjar med 110 följt av nät-id och till sist värd-id.

Viktiga delar i ett IPv4-**huvud** är **TTL**, sändaradressen och destinationsadressen. TTL står för Time-to-live och berättar det maximala antalet routrar som **datagrammet** (paketets namn för IPv4) får passera. Alla berörda routrar tar bort 1 från TTL när datagrammet passerar, routern som får 0 slänger datagrammet. TTL finns för att inte datagrammet ska cirkulera obegränsat i nätet. TTL består av 8 bitar och börjar på 65:e (9:e byten) biten in i pakethuvudet. Sändaradressen börjar på 13:e (97:e biten) byten och destinationsadressen börjar på 17:e (129:e biten). IPv4 är uppbyggd med ett huvud på 20–60 bytes (en byte består av 8 bitar och en bit är en siffra) och data på 0–65 515 bytes.

Eftersom TCP/IP inte definierar något länk- och fysiskt skikt kan ett datagram stöta på olika protokoll, som har olika regler för ramformat. Skulle datagrammet vara för stort sker något som kallas för **fragmentering**. Detta kan ske hos sändaren eller vilken router som helst på vägen. Maxlängd på en ram kallas för maximum transfer unit (MTU). Eftersom IP är förbindelsefritt garanteras det inte att fragmentet kommer fram, eller ens tar samma väg till mottagaren.

7.4 IPv6

7.5–9 ARP/DNS/ICMP och omvandling mellan IPv4 och IPv6

ARP är en förfrågan (**request**) som en dator skickar ut på ett lokalt nät för att få reda på mottagarens MAC-adress. Eftersom alla datorer i det lokala nätet får meddelandet är det en broadcast. Mottagaren svarar med att skicka sin MAC-adress, vilket inte behöver vara en broadcast då sändarens adress är känd. Behöver man en MAC-adress på ett annat LAN skickas en request om länkadress till vägväljaren som skickar vidare. Detta är för IPv4.

För IPv6 använder man något snarlikt som kallas **NDP** (Neighbor–Soliciation Message). Grundidén är att NDP också skickar en förfrågan och får ett svar med MAC-adress.

DNS översätter namn på webbservrar för att man istället för att skriva in IP-adresser skriver namn. Domäner för varje namn finns hierarkiskt i ett namnsystem, och kan ha underdomäner som också har varsitt unikt label. Detta för att undvika sammanblandning.

ICMP kompenserar för IP:s svagheter. IP är förbindelsefritt och saknar felhantering. ICMP kapslas in i IP-datagram med ett meddelande som ansvarar för att nå rätt destination. Protokollet har annars inga funktioner för transport. Själva ICMP-huvudet innehåller bitar för vilken typ av **meddelande** och varför det skickats tillsammans med kontrollsumma, rester av huvudet och data. Rester av huvudet och data har med vad för typ meddelande som skickats. Det kan vara antingen ett felmeddelande eller förfrågningar. I vilket fall har inte ICMP någon funktion för att hjälpa till, den bara **meddelar**.

Föreläsning 7

Kapitel 12.1, 15.5 – Applikationer och SUNET/LUNET

Föreläsning 8

Kapitel 13, 14.1–14.3 – POTS och mobila nät

POTS – PLAIN OLD TELEPHONE SERVICE

Förväntningarna på telenätet och det mobila nätet är att det alltid ska vara användbart. Systemet är något av de mest komplexa människan någonsin skapat. 6 minuter varje får systemet vara nere för uppgraderingar, men inte mer än så.

Telenätet kan delas in i två nätkategorier. Accessnät ansluter abonnenten (den som ringer) till en lokalstation, där signalen digitaliseras. Det betyder att signalen mellan abonnenten och lokalstationen alltid är analog. Väl i lokalstationen digitaliseras signalen så att trunknätet, den andra kategorin, kan transportera talsamplen. Det fasta telenätet är kretslopplat, vilket betyder att det enbart är ditt samtal som utnyttjar nätets kapacitet.

Alla abonnenter är kopplade via en egen ledning till en lokalstation (kopparledning). Talet som skickas via ledningen till lokalstationen är analog på ett frekvensband mellan 300–3 400 Hz. Att det är just detta intervallet beror på att man tidigt kom på att man hör talet klart och tydligt, men

också för att man förr i tiden hade haft problem med att signalen dämpades på långa avstånd. Man insåg att om man kopplade in pupinspolar med jämnt intervall kunde man minska dämpningen kraftigt på signaler på under 4 000 Hz. När signalen tagit sig till lokalstationen kodas talet med PCM, alltså sampling med 8kHz och kodas med 8 bitar. Varje telefonsamtal genererar en bitström på 64kbps i varje riktning, och består av 8-bitars sampel. Från och med nu tar trunknätet över, och dess uppgift är att överföra den nu digitala dataöverföringen till den andre abonnentens lokalstation, där den är ansluten.

Trunknätet har som uppgift att transportera talsampel både genom länder och mellan länder. Det kallas för ett transportnät och har en hierarkisk struktur, där lokalstationen är längst ner. Den ska kunna koppla samtal mellan egna abonnenter, och uppåt i hierarkin. Alla abonnenter är kopplade till en lokalstation, så högre nivåer har inga abonnenter kopplade till sig utan bara stationer. Lokalstationerna är kopplade till förmedlingsstationer, som har som uppgift att koppla vidare. Högst upp finns internationella stationer som sköter koppling mellan olika länder, men också stationer som utgör gateways till mobila nät.

Lokalstationerna ligger inom ett riktnummerområde, och är kopplade till en högre hierarkisk station. De kan dock i praktiken koppla samtal mellan områden, på grund av ekonomiska och tillförlitlighetsmässiga skäl. När alla tidsluckor mellan två lokalstationer är upptagna kan de koppla samtalet mellan flera förmedlingsstationer, vilket kallas för en hierarkisk dirigering.

Telefonen har några viktiga delar för att kunna ta emot, och ringa ett samtal. **Mikrofonen** används till att generera en analog talsignal, som ska ge ljud i telefonens **högtalare**. När ett samtal kommer in till telefonens basstation, ringer **ringsignalen**. När luren lyfts, bryter **klykomkopplaren** kretsen till ringsignalen, samtidigt som den sätter ihop en ny krets så att ström kan passera genom telefonen. **Knappsatsen** sänder olika kombinationer av frekvenser beroende på vilken siffra som trycks, och **talkretsen** innehåller kretsar för att kunna reglera ljudnivå för högtalaren och signalnivå för mikrofonen.

Varje lokalstation innehåller två delar: styrdel och kopplingsdel, där kopplingsdelen innehåller all hårdvara (materiell) för att kunna koppla upp samtal, och där styrdelen ”styr” (kopplar) kopplingsdelen. Den består av flera datorer. I lokalstationen finns också en linjekrets som ser till att de analoga signalerna, som sänts från abonnenten, digitaliseras och genererar sedan talsampel som skickas vidare över nätet.

Linjekretsen har några större funktioner: sändning av ringsignal, generering av 8-bitars sampel, krets för skydd vid åsknedslag och test av abonnentledning, bland annat. Från varje abonnent går två kopparledningar, en för abonnentens tal och en för mottagarens tal. I linjekretsen delas de till fyra ledningar som gör omvandlingen till digital data.

Kopplingsdelen har två viktiga delar för hantering av talsampel: koncentratorn och gruppväljaren. Den fungerar ungefär som en switch, då den får in sampel både från koncentratorn och från andra stationer, och skickar vidare dem till rätt mottagare.

Styrdelen består av processorer, som innehåller olika telefonstationers programvara. Processorerna kan organiseras på olika sätt, men fungerar likadant. Deras uppgift är att samla in data, bearbeta den och skicka den rätt. Ingenting får försenas så att funktionaliteten för styrdelen äventyras, därför jobbar processorerna i en jämn snabb takt. Skulle en processor gå sönder, finns en back-up som tar över jobbet tills den ordinarie är lagad.

Föreläsning 9

Kapitel 17.1, 18 – Prestandamått, nätdrift, terminologi