

Diadoker

Datakommunikation

Innehållsförteckning

Kapitel 1: Inledning

Kapitel 2: Nivå 1, den fysiska nivån

Kapitel 3: Nivå 2, data länk nivån

Kapitel 4: Nivå 3, nätverk

Kapitel 5: Nivå 4, transport

Kapitel 6: Nivå 5, session

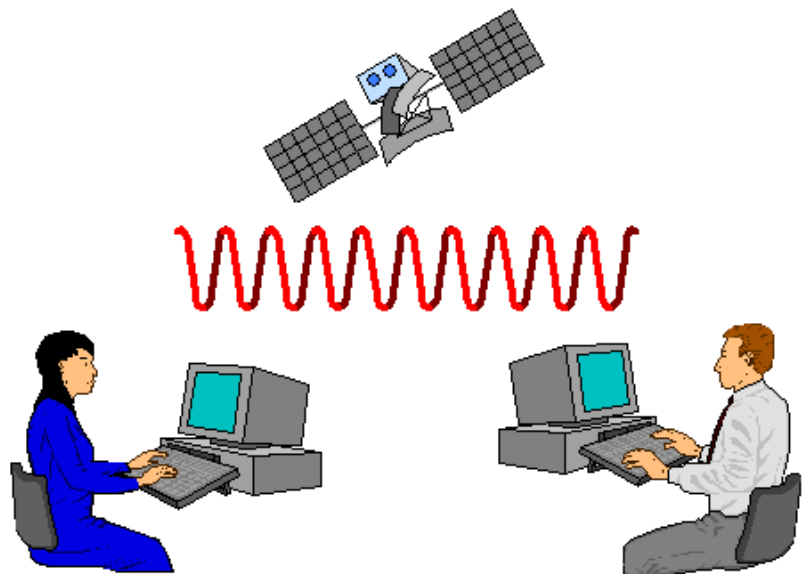
Kapitel 7: Nivå 6, presentation

Kapitel 8: Nivå 7, applikation

Kapitel 9: Design av kommunikation

Kapitel 10: Referenser

[Tillbaks till Diadoker](#)



Förord

Detta dokument har som syfte att skapa en bra förståelse för de behov och problem som man ställs inför vid kommunikation mellan olika typer av enheter. Detta dokument är också i huvudsak orienterat mot PC-världen och dess

kommunikationsbehov. OSI-modellen används som bas för beskrivningen eftersom OSI-modellen är strukturerad och beskriver just det som detta dokument vill förmedla.

I datakommunikationsvärlden finns en oerhörd mängd av uttryck och förkortningar som gör det svårt att ta det hela till sig. Detta beror främst på att det finns väldigt många aktörer på kommunikationsmarknaden som tillverkar produkter. Dessa aktörer vill sälja sina produkter på bekostnad av någon annans, och att ge saker och ting klatschiga namn tros främja detta mål. Standardisering tar av nödvändighet relativ lång tid, speciellt i jämförelse med utvecklingen av produkter inom IT-världen. Det gör att företag som har stora marknadsandelar kommer att skapa defacto-standarder just p.g.a. sin ledande marknadsställning.

Implementationen av de funktioner som behövs för att datakommunikation skall fungera, görs oftast med hjälp av program och blir därför väldigt abstrakt, inte mycket att ta på. Varje leverantör har sina produkter, och det är inte meningen att i detta dokument redogöra för produkterna, utan för idéer, behov, funktioner, standarder och defacto-standarder, och olika lösningar som kan användas. Utvecklingen fortskrider även på detta område, men det som oftast utvecklas är att hastigheten ökar på förbindelserna. De protokoll som finns etablerade utvecklas inte i samma takt. Efterhand som Internet utvecklas kommer nya behov att dyka upp och därför blir detta dokument en spegling av den tid som det är skrivet i.

I det inledande kapitel 1 beskrivs datakommunikation allmänt och en del termer och begrepp används redan där, men förklaras inte i detalj. Dessa förklaringar kommer i senare kapitel och där dessa termer och begrepp sätts in i ett sammanhang som ger mening åt dessa.

Kapitel 1: Inledning

1.1. Hur möter du datakommunikation idag?

Datakommunikation är i dagens samhälle nästan lika osynligt som telefonen, det vill säga att vi inte lägger märke till att det just är datakommunikation som används.

När du skall ut och resa och bokar plats och tid på flyg eller på tåg, så görs detta genom att det kontor du anlitar är kopplat mot en bokningscentral som håller reda på alla bokningarna. Om du skall flyga till Paris till exempel, så anlitar du en resebyrå som kontrollerar vilka avgångar som finns vid den önskade tidpunkten, kontrollerar om ledig plats finns, och om du sedan vill boka, reserverar plats på den avgång som du skall åka med. Eller så använder du Internet och gör dina bokningar själv. Varje flygbolag har en egen bokningscentral, men denna är i sin tur uppkopplad mot en ännu större central som täcker flera bolag. Det är komplicerade program som behövs för att det skall fungera, men det är datakommunikation som gör det möjligt.

Om du skall ta ut pengar på en bankautomat, så är denna kopplad med datakommunikation till bankens centraldator som håller reda på alla konton och som avgör om du kan få några pengar eller inte.

När du går och handlar i till exempel en matbutik, så är kassorna kopplade mot en dator som registrerar varan med hjälp av streckkoden och antalet artiklar som kassörskan anger. Kassan uppdaterar lagerlistan, räknar fram om det skall beställas hem mera av denna vara, och skriver priset i kassan och på kvittot. Du kanske betalar med betalkort som Eurocard, där man läser kortet i en speciell läsare som kopplar sig mot Eurocard som kontrollerar att betalkortet är okej.

1.2. Hur började det?

Kommunicera på långa avstånd har varit svårt för människan från tidernas begynnelse. Men efterhand har olika tekniker uppfunnits, som till exempel röksignaler, eller två flaggor som i bestämda förhållanden till varandra har bestämd mening och så vidare.

Telegrafien med en telegrafist som sitter med sin Morseapparat och tickar fram telegrammen är en teknisk uppfinning som förändrade både avstånd och noggrannhet vid meddelandeöverföringen. Telegrafien använde kabel och senare även radio, och skickade morsekod som en kombination av pulser med närvaro av spänning och frånvaro av spänning samt tiden som detta varade. Morsekoden är uppkallad efter Morse som skapade en kod som man skulle använda för att förmedla våra skrivtecken elektriskt. Dessa pulser kom efter varandra i serie, och det kallas att man skickar signalerna seriellt.

Efter hand som textmängden ökade, så ökade kraven på effektivare överföring och Telex apparaten uppfanns. Denna använde ett tangentbord och en skrivmekanism för att förmedla informationen. Detta medförde att det blev nödvändigt att använda en annan kod än Morsekoden för att överföra våra tecken elektriskt. En man som heter Baudot lanserade en 5 bitars binär kod, som fått namn efter honom. [denna kod](#) kan överföra 32 olika kombinationer av binära tecken, vilket räcker för vårt alfabet. Men en annan kod kom snart i användning, [ASCII](#) (American Standard Code for Information Interchange), en kod som har 7 bitar, men finns nu utökad till 8 bitar.

De apparater som används av användaren eller operatören kallas för terminaler.

1.2.1. Asynkron överföring

När man skall skicka ett tecken så skrivs detta med tangentbordet, och detta tecken måste först omvandlas till en kod, till exempel [ASCII](#), och sedan till elektriska pulser, som skickas seriellt iväg direkt. En människa kan inte skriva med jämn takt, så för att mottagaren skall kunna säkerställa att samma tecken som skickats även mottages, införde man att varje tecken fick en startbit och en stoppbit som avgränsare på varje tecken. Detta gjorde att mottagaren kunde synkronisera sig på startbiten och sedan visste att när stoppbiten kom var tecknet slut. Varje bit har en bestämd längd, som beror på den hastighet som man vill sända med. Så genom att mottagare och sändare vet vilken hastighet som skall användas, kan varje tecken identifieras.

Detta sätt att överföra tecken för tecken kallas asynkron överföring, mottagare och sändare behöver inte vara i synkronisation med varandra. Synkroniseringen sker vid varje tecken. Denna kommunikation sker i regel mellan två maskiner som står i förbindelse med varandra, och kallas punkt till punkt förbindelse. Det behövs ingen adressering, mottagaren är välkänd och endast en.

Telexmaskinen, TTY, stod som modell när terminaler började kopplas till en dator. När man kopplade den till datorn så fungerade det så: När en tangent trycks ned så genereras ASCII koden för detta tecken, och omvandlas till elektriska pulser som skickas seriellt till datorn. Där går pulserna dels vidare in i datorn dels så skickas de tillbaks till TTY och tecknet skrivs på skrivardelen. Genom detta förfarande: först går signalen till datorn, sedan skrivs den på terminalen, blev det en kontroll för operatören att datorn hade uppfatta rätt tecken, man kallar det eko. Kontrollen blir ju att om operatören skriver ett A, och skrivaren skriver ett A så måste datorn ha uppfattat det på samma sätt, eftersom det ju har varit och vänt vid datorn. Men för att få en säkrare och automatiserad kontroll att rätt tecken skickas, infördes en extra bit till varje tecken, en paritetsbit, udda eller jämn paritet kan användas, det vill säga udda paritet är att antalet bitar som har värdet 1 tillsammans skall vara udda.

Asynkron överföring är en tecken-för-tecken överföring, en startbit och stoppbit tillförs på varje tecken. Detta blir en ganska ineffektiv överföring, 20% av bitarna är inte information av det som överföres på linjen.

1.2.2. Synkron överföring

För att öka effektiviteten på linjen så använder man synkron överföring, vilket betyder att sändare och mottagare måste synkroniseras på annat sätt än via start/stopp bitar. Detta görs så att antingen mottagaren eller sändaren eller en extern källa används för att generera klockpulser för synkronisationen, det vill säga att bitarna överförs i takt med dessa klockpulser. Tecknen skickas i ett block som måste ha en definierad början och slut, och som har paritetskontroll på hela blocket. Speciella synkroniseringstecken måste skickas före varje block så att mottagaren får tid att synkronisera sig, dessa tecken var i början 4 st. SYN (16h), men numera räcker det med 2 st. Om data som sänds är stort kan det behövas synkroniseringstecken insprängda i datadelen för att säkerställa synkroniseringen. Sändare och mottagare måste ha ett buffer för att kunna hålla ett helt block tillgängligt för att kontrollera/generera paritet. Synkron överföring etablerades när bildskärmar började användas istället för skrivardelen på terminalen, och när det skall sändas trycks en speciell sändtangent.

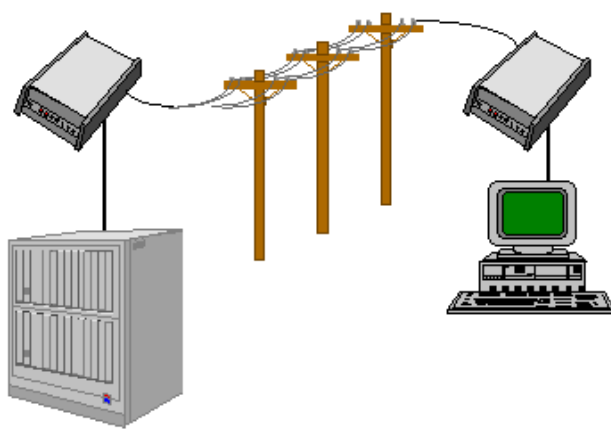
1.2.3. Protokoll

Eftersom paritetskontrollen på hela blocket infördes måste man kunna hantera situationen att ett block mottages felaktigt, har fel paritetstecken. Regler införs hur detta skall hanteras, och man kallar reglerna för ett protokoll. Varje block skall kvitteras hur det har mottagits, om det är

felaktigt så får blocket sändas igen. Kvittensen är ACK (06h), som betyder mottaget okej, respektive NAK (15h), som betyder återsänd.

Nu har kommunikationen blivit lite mera komplex, trafiken måste ske så att mottagare och sändare är i takt även på meddelande nivå. Trafik eller överföring i endast en riktning kallas simplex överföring, att kunna överföra i båda riktningarna kallas duplex överföring. För att kunna överföra i båda riktningarna samtidigt kallas full duplex och kräver att det finns kablar för detta, det vill säga dubbla kablar. Detta beskrivs mera utförligt i kapitel 2. Överföring i båda riktningarna men en i taget kallas halv duplex, och kräver bara en överföringskabel. Protokoll har utvecklats för att handha dessa komplexa situationer, och för att dessa protokoll skall vara effektiva, och även vara lätta att skapa och underhålla har det skapats protokoll som bildar en form av hierarki. Detta beskrivs mera i kapitel 1.3.1 med OSI modellen.

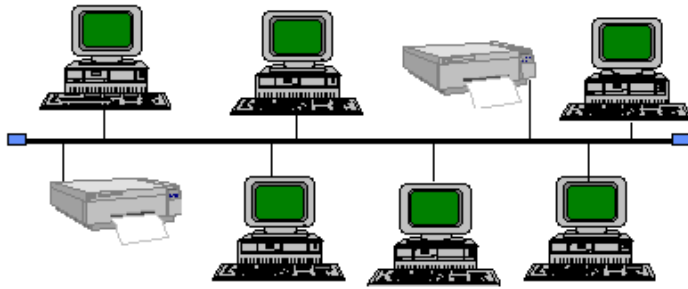
1.2.4. Modem



Terminalerna var i början kopplade direkt till datorn via en kabel som var begränsad i längd p.g.a. dämpning, men behov att vara på längre avstånd fanns, och då började man utnyttja det allmänna telefonnätet. Telefonen är också en kommunikation som använder kabel, för att överföra mänskligt språk genom att omvandla detta till elektriska signaler. Telefonen arbetar analogt, eftersom vårt sätt att prata är analogt. Datorn arbetar digitalt, så en omvandling måste göras så att telefonnätet kan utnyttjas. Denna omvandling sker i ett modem, modulator/demodulator. Sändarmodemet omvandlar digitala signaler till analoga, och i mottagarmodemet omvandlas analoga signaler till digitala.

Telefonnätet är väl utbyggt i hela världen och det gör att man kan koppla upp sig mellan två punkter i princip varsomhelst på jordklotet, till exempel mellan Ödåkra och Los Angeles. Finns det rätt utrustning på båda ställen och rätta protokoll, så kan dessa två punkter kommunicera på ett kontrollerat och säkert sätt.

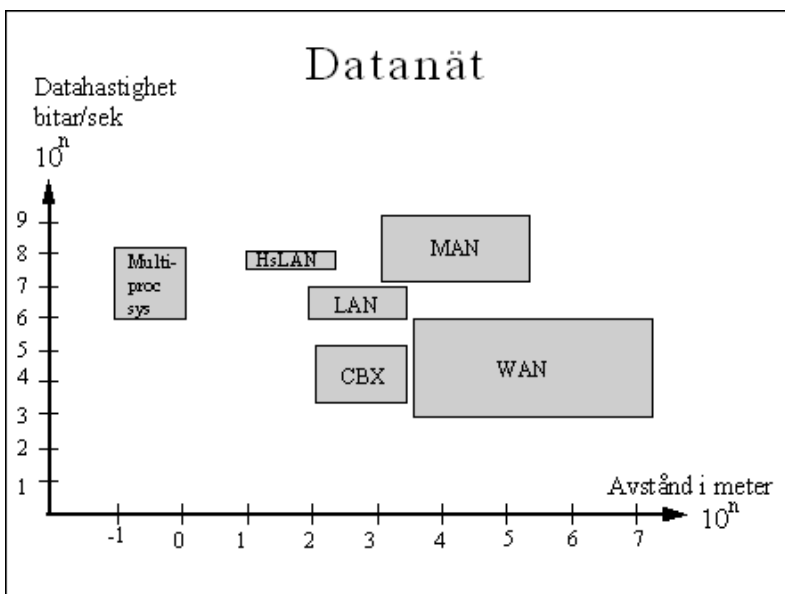
1.2.5. LAN



I början fanns det lokalt endast terminalnätverk, som hade låg överföringskapacitet, men när PC:n kom blev det möjligt att distribuera datorkraften på ett nytt sätt. Detta skapade nya problem främst genom att den data som man skulle arbeta med fanns utspridd och var inte på samma nivå på alla dessa olika ställen som den fanns på. Behov av att nå och flytta stora mängder data uppstod och lösningen blev i okala snabba nätverk med en filserver som lagrade alla filer med data, och alla som hade behov kunde nå dem via det lokala nätverket. Det uppstod olika typer av lokala nätverk, dels beroende på teknik, dels på topologin. Det mest använda LAN-nätverket i dag är Ethernet.

1.2.6. Datanätverk

När man utnyttjar telefonnätet för datakommunikation benämns detta att man använder ett nätverk eller ett datanätverk. Telefoninätet kallas för WAN (Wide Area Network), eftersom det sträcker sig över stora avstånd, globala avstånd. WAN använder flera typer av media för att nå de stora avstånden, till exempel kabel, radio och satellit.



När man kopplar ihop datorer lokalt inom en byggnad eller fabriksområde, till ett eget litet nätverk kallas det för LAN. Det har inte definierats som att detta skall ske med ett viss kommunikationssätt, men det förknippas oftast med höghastighetsnätverk som Ethernet och Token ring. Dessa LAN har utvecklats till olika typer med olika protokoll och topologier. Topologi menas hur nätverket utbreder sig i rummet, som till exempel buss, ring eller

stjärnätverk.

Stora företag och organisationer har sin verksamhet utspridd på stora områden, och har behov av att koppla samman sina LAN. Detta kan göras med WAN men detta ger inte så stor kapacitet och därför har en ny typ av nätverk tillkommit, MAN (metropolitan Area Network). MAN är inom ett större område än LAN men mindre än WAN.

Kapaciteten på de olika nätverken är beroende på avståndet mellan de kommunicerande enheterna. Bilden är en principbild över hastigheten kontra avståndet och skall inte ses som någon definition av vilka maximala hastigheter som finns, eftersom detta förändras kontinuerligt.

I multiprocessor system och inom processorsystem, är avstånden mycket korta och därför kan kapaciteten vara hög, samtidigt som skall framhållas att kommunikationen inom dessa typer av system är parallell. WAN är för stora avstånd, globala avstånd, och därför har detta nätverk inte så stor kapacitet. Hs-LAN är speciella hög-hastighets LAN, som är utvecklade av speciella leverantörer av datorsystem. MAN har ganska hög kapacitet, och det är därför att dessa skall "serva" ett stort antal mindre LAN. CBX är telefonväxlar, som ju har en ganska liten utbredning i avstånd men har kapacitet ungefär som WAN.

1.3. Modell för kommunikation

Protokollen som skapades på olika hierarkiska nivåer löser olika typer av problem. Genom hierarkin så kan bara vissa protokoll kommunicera med varandra och genom att dessa protokoll utvecklades av organisationer som var oberoende av varandra blev det stora problem att koppla ihop utrustningar av olika slag. Varje datorleverantör av rang hade utvecklat sina protokoll för att passa sina maskiner och behov, och när dessa skulle kommunicera med en annan leverantörs maskiner och protokoll, så gick det inte. Genom internationella organisationer har dessa problem försökt att bli lösta genom att skapa standarder som alla skall följa och därmed möjliggöra kommunikation alla till alla. ISO (Internationella Standard Organisationen) tog fram en modell över de funktioner och behov som datakommunikation ger upphov till, OSI-modellen.

Datakommunikation sker med information som beroende på vilken nivå i OSI-modellen som man talar om, har olika s.k. ramar. Dessa ramar är i form av meddelanden, segment, paket, tecken eller block, och bitar. Bitar är binära tecken, bit (binary digit), som kan ha värdet 1 eller 0, men beroende på hur man kombinerar dessa bitar så kommer bitarnas position att spela betydelse. Normalt kombinerar vi bitarna i grupper om 8, en oktett, och denna grupp kallas byte på engelska, by eight, detta gör vi för att kunna representera vårt alfabet binärt. Denna representation har standardiserats och i ett senare underkapitel redovisas några standarder i detalj.

1.3.1. OSI modellen

OSI (Open System Interconnection), ISO 7498, CCITT X.200

OSI är en referensmodell för kommunikationstjänster, och är en hierarkisk indelning av de funktioner som behövs för att två enheter skall kunna kommunicera. OSI är ett ramverk för att standardisera och funktionsuppdelade kommunikationen i olika nivåer, där varje nivå är fristående från de andra nivåerna. Nivåerna kommunicerar med varandra genom att en nivå anlitar tjänster som nivån under erbjuder, och den som erbjuder en tjänst ger ett resultat tillbaks på hur detta blev utfört. På varje nivå samlas likartade funktioner och reglerna inom en nivå kan bytas ut eller förändras, utan att påverka nivåerna ovanför eller nedanför. En nivå behöver inte känna till hur en underliggande nivå är utformad, utan endast av vilka tjänster denna kan utföra, och gränssnittet för detta. Varje nivå erbjuder tjänster till nivån ovanför, och

utnyttjar de tjänster som nivån under erbjuder.

OSI modellen

7	Applikations nivå
6	Presentations nivå
5	Sessions nivå
4	Transport nivå
3	Nätverks nivå
2	Datalänk nivå
1	Fysisk nivå

OSI beskriver kommunikationen mellan noder, där en nod är en adresserbar kommunikationsenhet, och som har protokoll för kommunikation mellan två närliggande noder och mellan noder som har andra noder mellan. Varje nivå kommunicerar med samma nivå på den andra noden genom utbyte av styr- och kontrollinformation som fogas till det ursprungliga meddelandet i form av "header" och "trailer", man säger att varje nivå har sitt eget protokoll.

OSI modellen är uppdelad i 7 nivåer som alla har specifika uppgifter. Eftersom modellen beskriver en datakommunikationstjänst så är den översta nivån ingången till tjänsten för applikationsprogrammen, applikationsnivån. I nästa nivå hanteras hur informationen skall presenteras till applikationsprogrammen, presentationsnivån. För att kommunicera måste ett "möte" etableras så att utbyte av information kan ske, detta möte måste kontrolleras så att det inte avbryts, bestämma hur dialogen skall ske och vem som skall delta, sessions nivå. Det måste etableras en kommunikationslänk mellan de deltagande och denna måste kontrolleras så att alla meddelanden kommer fram och är korrekta, transportnivån. Deltagarna måste adresseras och en väg genom nätverket etableras så att kommunikationen kan ske, nätverksnivån. Den valda vägen i nätverket skall kopplas upp och kontrolleras, datalänk nivån. Det behövs en fysisk länk mellan de kommunicerande, fysiska nivån.

Genom den hierarkiska modellen hanteras kommunikationen på de olika nivåerna på olika sätt och med olika detaljnivå. I nivå 1 arbetar modellen med databitar, den kan sägas endast förstå och hantera bitar det vill säga 1 och 0. Informationen som finns representerad av bitarna förstås inte på denna nivå. I nivå 2 arbetar modellen med dataramar (frames), och dessa kan vara tecken, block eller paket. På nivå 2 kan informationen förstås och därför ligger den första av kontrollfunktionerna på denna nivå som säkerställer att informationen har anlänt korrekt. I nivå 3 hanteras informationen som paket/datagrams, och i nivå 4 med datagrams/segments. På nivå 4 säkerställs att alla datagrams/segments har anlänt och kan bilda ett helt meddelande, som nivåerna 5, 6 och 7 arbetar med, det är den andra kontrollfunktionen. Den tredje kontrollfunktionen sker på nivå 5 som säkerställer att alla meddelanden som skall hanteras på "mötet" har anlänt.

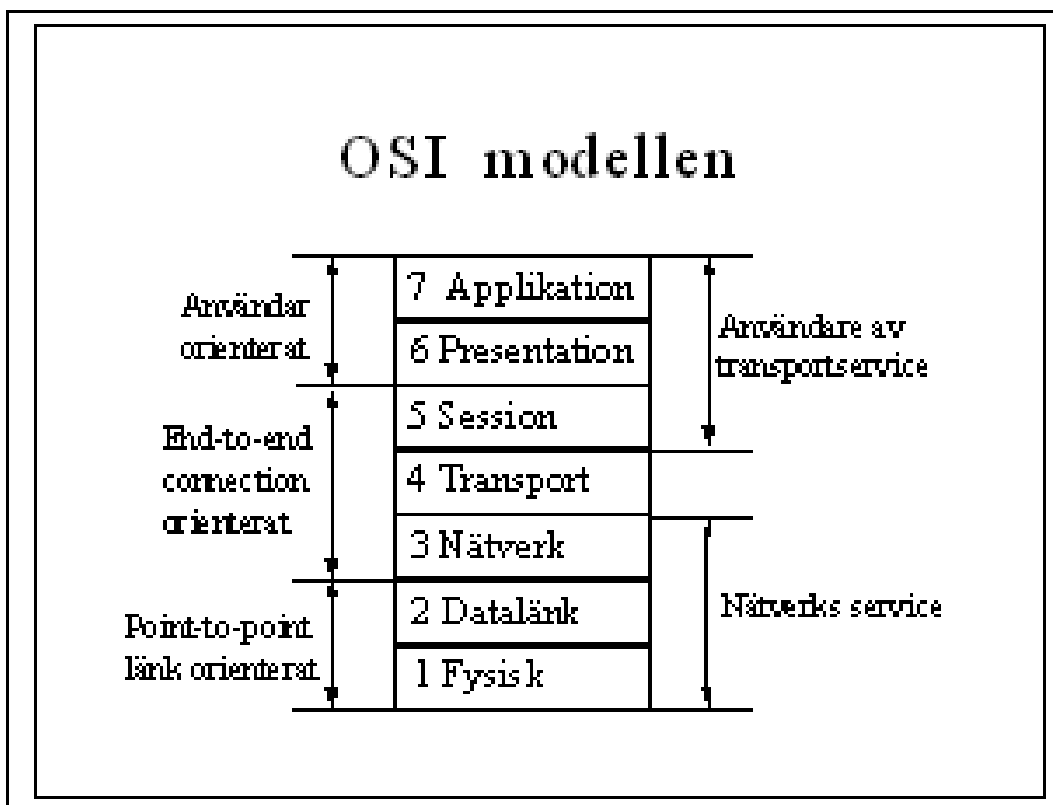
De tre kontrollfunktionerna för säker kommunikation kan också ses på detta sätt:

på nivå 2 mellan kommunicerande enheter: rätt data? (data all in one piece?).

på nivå 4 mellan slutenheter: kom all data fram? (did the data get there?).

på nivå 5 mellan sessioner: pågår mötet? (are we still talking?).

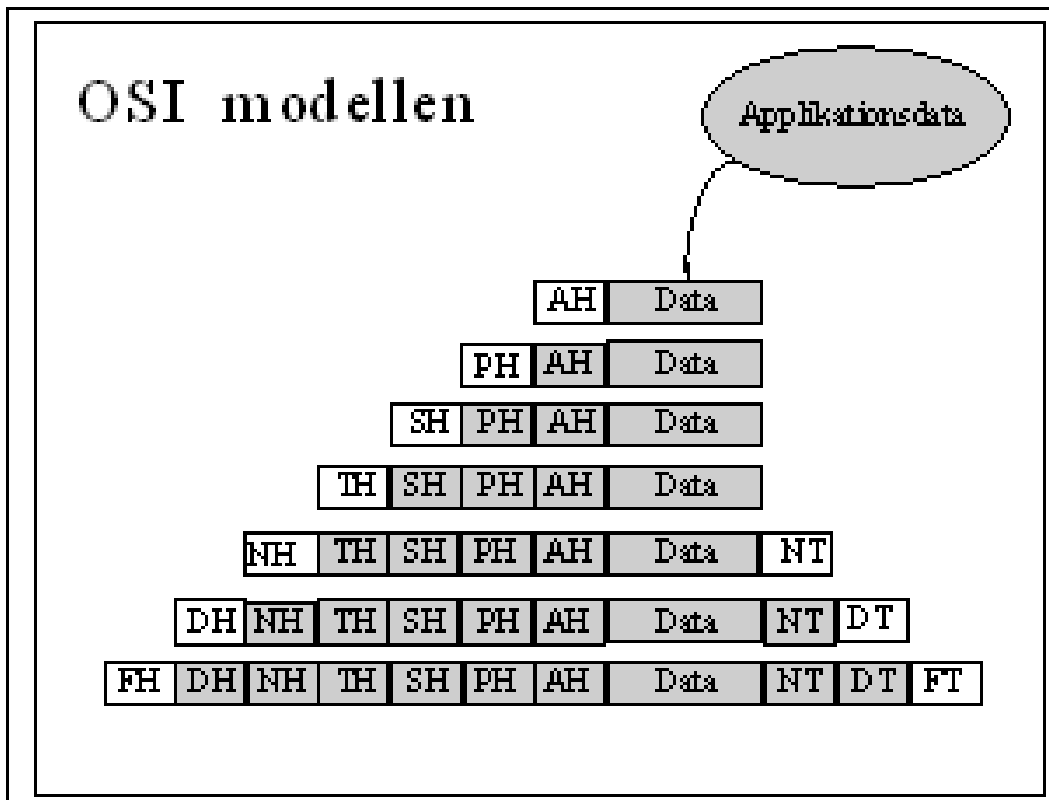
Kontrollfunktionerna skall fungera så att respektive nivå har som uppgift att lösa sin del, men om inte det går att lösa så måste detta rapporteras till nivån ovanför. Tjänsterna som erbjuds av respektive nivå utnyttjas genom att kommandon ges till den som skall utföra tjänsten och den som utför tjänsten sedan rapportera hur den har utförts. Denna rapportering används för att meddela om det inte har gått att leverera sin del av jobbet.



Modellen kan ses från olika perspektiv, till exempel att nivåerna 5, 6, och 7 är användare av transportservice för att kommunicera med sin motpart, och transportservicen använder nätverksservicen för att möjliggöra den faktiska uppkopplingen mellan de noder som kommunicerar. Eller att nivå 6 och 7 är orienterat mot användaren, dessa nivåer är ju helt orienterade mot applikationen som användaren använder. Nivåerna 3, 4 och 5 ser till att en användare kan kommunicera till en annan användare, och nivåerna 1 och 2 ser till att den "fysiska" överföringen blir av mellan de inblandade noderna.

Utbyte av styr- och kontrollinformation nivåer emellan, sker genom att till den ursprungliga informationen (data) från applikationen addera till en "header" och på vissa nivåer även en "trailer". Dessa header och trailer kan endast förstås av respektive nivå. Det innebär att när applikationsnivån lägger till sin header och lämnar vidare till presentationsnivån så blir data för presentationsnivån den ursprungliga applikationsdatan plus headern från applikationsnivån. På den fysiska nivån är data för denna nivå lika med: den ursprungliga applikationsdatan plus headern från applikationsnivån plus headern från presentationsnivån plus headern från sessionsnivån plus headern från transportnivån plus headern och trailern från nätverksnivån plus headern och trailern från datalänk nivån. Dessa headers och trailers är det sätt som protokollen kommunicerar med motsvarande protokoll på den andra noden. Det finns endast en fysisk länk mellan noderna, men eftersom headers och trailers endast förstås av sina respektive nivåer, kan man säga att varje nivå har en virtuell kommunikation med sin egen nivå på den andra noden. Headers och trailers måste inte alltid finnas med, det beror på hur

protokollet är implementerat, modellen definierar bara att det kan finnas behov att ha både trailer och header.



AH, applikations nivåns header, som till exempel från vilken applikation till vilken applikation.

PH, presentations nivåns header, som till exempel typ av code set.

SH, session nivå header, som kommunikations information.

TH, transport nivå header, som till exempel checksumma.

NH, nätverk nivå header, som source och destination nätverksadress.

DH, data länk nivå header, som till exempel fysisk adress.

FH, fysisk nivå header, som till exempel preamble.

NT, nätverks nivå trailer.

DT, data länk nivå trailer, som till exempel checksumma.

FT, fysisk nivå trailer, som till exempel postamble.

Ett exempel, där vi har behov av att utnyttja en kommunikationstjänst, som att hämta ett dokument från filserver till ordbehandlingsprogrammet på arbetsstationen. Du sitter på arbetsstationen och använder ett ordbehandlingsprogram, till exempel Word, och du lagrar alla dina dokument på filservern, och nu skall du öppna ett befintligt brev. Word kommer då att begära få läsa brevet från filservern, därför ber Word om en kommunikationstjänst för att överföra denna begäran till filservern. Word begär kommunikationstjänsten och applikationsdata blir då namnet på den tillsammans med begäran om läsning av denna, detta "ramlar" ner genom OSI modellen och varje nivå adderar till sina header och trailer och som sedan överföres via den fysiska ledningen till filservern. Det paket som anländer till filservern består nu av filnamnet och begäran att få läsa den, plus alla header och trailer från respektive nivå. På filservern packas nu detta paket upp av respektive nivå som plockar bort sin header och trailer och lämnar resten som data upp till nästa nivå. När applikationsnivån lämnar det ifrån sig är det den ursprungliga begäran om att få läsa en speciell fil. Filservern utför denna begäran och anlitar nu kommunikationstjänsten för att överföra den begärda informationen som applikationsdata. På nytt "ramlar" detta ner genom OSI modellen och alla header och trailer läggs till, överföres till din arbetsstation och packas upp. Applikationsnivån i arbetsstationen lämnar nu den begärda informationen till Word och du kan utföra ditt arbete med brevet.

Gränssnitten mellan de olika nivåerna är inte för användarna tydliga och lätt avgränsade, utan användaren ser dem som produkter som skall laddas i datorn och engageras. Har rätt produkter laddats så kommer de att arbeta tillsammans, annars blir det ingen kommunikation.

Gränssnittet mellan nivå 1 och 2 är mycket otydligt för användaren, i till exempel LAN är de sammankopplade i en drivrutin som laddas till NIC (Network Interface Card). Mellan nivå 2 och 3 är det lite tydligare eftersom här är en annan logisk gräns som benämnes protokollstacken, som är nivå 3 och 4 och deras protokoll. En protokollstack kan operera mot flera typer av länkprotokoll och för användaren gäller det att välja de som hör ihop. Mellan nivå 3 och 4 har återigen ingen tydlighet eftersom protokollstacken hör samman och är ju dessa två nivåer. Mellan nivå 4 och 5 kommer igen ett lite tydligare gränssnitt eftersom här slutar protokollstacken och därmed skall andra produkter kopplas till stacken. Nivåerna 6 och 7 är oftast implementerade i applikationerna och är därför också väldigt otydliga för användaren. De som mest har anledning att tänka på de här gränssnitten är de programmerare som skall förverkliga de olika protokollen så att de blir effektiva som protokoll men också att de blir lätta att underhålla.

Exempel

Layer		Misc. examples	TCP/IP suite	SS7	AppleTalk suite	OSI suite	IPX suite	SNA	UMTS
#	Name								
7	Application	NNTP, HL7, Modbus, SIP, SSI	DHCP, FTP, Gopher, HTTP, NFS, NTP, RTP, SMPP, SMTP, SNMP, Telnet	ISUP, INAP, MAP, TUP, TCAP	AFP	FTAM, X.400, X.500, DAP		APPC	
6	Presentation	TDI, ASCII, EBCDIC, MIDI, MPEG	MIME, XDR, SSL, TLS (Not a separate layer)		AFP	ISO 8823, X.226			
5	Session	Named Pipes, NetBIOS, SAP, SDP	Sockets. Session establishment in TCP. SIP. (Not a separate layer with standardized API.), DNS		ASP, ADSP, ZIP, PAP	ISO 8327, X.225	NWL ink	DLC?	
4	Transport	NBF, nanoTCP, nanoUDP	TCP, UDP, SCTP	SCCP	ATP, NBP, AEP, RTMP	TP0, TP1, TP2, TP3,	SPX		

						TP4			
3	Network	NBF, Q.931	IP, ICMP, IPsec, ARP, RIP, OSPF	MT P-3	DDP	X.25 (PLP), CLNP	IPX		RRC (Radio Resource Control) PDCP (Packet Data Convergence Protocol) and Broadcast/Multicast Control (BMC)
2	Data Link	802.3 (Ethernet), 802.11a/b/g/n MAC/LLC, 802.1Q (VLAN), ATM, CDP, HDP, FDDI, Fibre Channel, Frame Relay, HDLC, ISL, PPP, Q.921, Token Ring	PPP, SLIP, PPTP, L2TP	MT P-2	LocalTalk, TokenTalk, EtherTalk, AppleTalk Remote Access, PPP	X.25 (LAPB), Token Bus	IEEE 802.3 framing, Ethernet II framing	SDLC	LLC (Logical Link Control), MAC (Media Access Control)

1	Physical	RS-232, V.35, V.34, I.430, I.431, T1, E1, 10BASE-T, 100BASE-TX, POTS, SONET, DSL, 802.11a/b/g/n PHY		MT P-1	RS-232, RS-422, STP, PhoneNet	X.25 (X.2 1bis, EIA/TIA-232, EIA/TIA-449, EIA-530, G.703)		Twinx	UMTS L1 (UMTS Physical Layer)
---	----------	---	--	--------	-------------------------------	---	--	-------	-------------------------------

Figuren visar några produkter som är vanliga i marknaden och hur dessa produkter i princip kan relateras till OSI modellen. Obs! det är ingen klar och tydlig rakt av jämförelse mellan dessa produkter och OSI eller mellan dessa inbördes, gränserna flyter lite i förhållande till OSI modellens nivåer och i förhållande mellan produkternas nivåer.

1.3.1.1. Olika typer av nätverk

Det finns olika typer av nätverk som klassificeras av hur kommunikationen sker. De delas in i paketkopplade nätverk, kretskopplade nätverk och meddelandekopplade nätverk.

I ett paketkopplat nätverk delas meddelandet som skall sändas upp i flera paket som sänds oberoende av varandra, och även kan varje paket utnyttja en annan väg genom nätverket än det föregående eller det kommande paketet. Sändare och mottagare behöver inte ha etablerat en förbindelse genom nätverket. Paketerna kan ha varierande storlek eller fast storlek. Exempel på paketförmedlande nätverk är Internet (TCP/IP), ATM (celler av 53 oktetter), frame relay, X.25.

I ett kretskopplat nätverk etableras först en förbindelse genom nätverket och som sedan är uppkopplad under hela kommunikationen, det uppträder som en fast förbindelse när det väl är uppkopplat. Exempel på kretskopplade nätverk är telefonnätet, datex.

1.3.2. Olika teckenkoder

1.3.2.1. Baudot

Baudot 5 bitars kod

5 bit binär kod	skift ner	skift upp
00 000	Blank	Blank
00 001	T	S
00 010	Vagn retur	Vagn retur

00 011	O	9
00 100	Tomt	Tomt
00 101	H	Pundtecknet
00 110	N	, (komma)
01 000	Radmatning	
01 001	L)
01 010	R	4
01 011	G	&
01 100	I	8
01 101	P	0
01 110	C	:
01 111	V	;
10 000	E	3
10 001	Z	"
10 010	D	\$
10 011	B	?
10 100	S	Bell

10 101	Y	6
10 110	F	!
10 111	X	/
11 000	A	-
11 001	W	2
11 010	J	'
11 011	skift ner	skift ner
11 100	U	7
11 101	Q	1
11 110	K	(
11 111	skift upp	skift upp

Datakommunikation kräver att den mänskliga informationen översätts på ett sätt som kan användas i den teknik som skall utnyttjas. Eftersom det är datorer inblandade och dessa arbetar med binära tecken så blir det en omvandling till binära tecken. Det som skall omvandlas är vårt alfabet, men även våra specialtecken, plus tecken som har tillkommit för att representera olika kommunikationsbetydelser.

Morse uppfann Morsekoden som är kombinationer av korta och långa signaler som representerar alfabetet och de numeriska tecknen, och denna lämpar sig inte för binär representation. Sedan kom Baudot med sin 5 bitars kod, där 5 bitar kan representera 32 olika tecken. Alfabetet som internationellt är a-z, vilket är 26 st. tecken, plus siffrorna som är 10 st. blir 36 st. Tecken, sedan finns det specialtecknen som till exempel, (komma) och . (punkt). Detta blir fler än de 32 tecken som Baudot koden kan representera, och därför använder man två speciella tecken för att fördubbla den möjliga teckenrepresentationen till 64, så att alla specialtecknen kommer med. Det går till på det här viset att två av koderna har betydelsen skift upp och skift ner. Man börjar med skift ner, och då används de kommande 5 bitarna till att representera alfabetets 26 st. tecken, sedan när man vill ha ett numeriskt eller specialtecken så använder man först skift upp tecknet, som då betyder att de kommande 5 bitarna nu representerar dessa tecken. Skall sedan ett alfabetstecken användas måste först ett skift ner tecken användas som återigen gör att de kommande 5 bitarna nu representerar alfabetet. Efter varje skifttecken kvarstår betydelsen som detta representerar tills nästa skift tecken ändrar på betydelsen.

1.3.2.2. ASCII

Baudotkoden var inte så bra när man började få behov av speciella kommunikationstecken till protokollen, som till exempel ACK för att indikera positiv kvittens. Detta kräver mer än 64 teckenkombinationer och 7 bitars ASCII koden föddes.

Styrtecken/kontrolltecken (Teckenkoderna 0-31)

De första 32 tecknen i ASCII tabellen är ej skrivbara utan används för att styra viss kringutrustning som t.ex skrivare där av namnet styrtecken.

D E C	OCT	H E X	BIN	Sym bol	HTML nummer	HTML namn	Beskrivning
0	000	00	00000000	NUL	�		Null char.
1	001	01	00000001	SOH			Start of Header
2	002	02	00000010	STX			Start of Text
3	003	03	00000011	ETX			End of Text
4	004	04	00000100	EOT			End of Transmission
5	005	05	00000101	ENQ			Enquiry
6	006	06	00000110	ACK			Acknowledgment
7	007	07	00000111	BEL			Bell
8	010	08	00001000	BS			Backspace
9	011	09	00001001	HT				Horizontal Tab
10	012	0A	00001010	LF	
		Line Feed
11	013	0B	00001011	VT			Vertical Tab
12	014	0C	00001100	FF			Form Feed
13	015	0D	00001101	CR			Carriage Return
14	016	0E	00001110	SO			Shift Out
15	017	0F	00001111	SI			Shift In

16	020	10	00010000	DLE			Data Link Escape
17	021	11	00010001	Dcl			XON Device Control 1
18	022	12	00010010	DC2			Device Control 2
19	023	13	00010011	DC3			XOFFDevice Control 3
20	024	14	00010100	DC4			Device Control 4
21	025	15	00010101	NAK			Negative Acknowledgement
22	026	16	00010110	SYN			Synchronous Idle
23	027	17	00010111	ETB			End of Trans. Block
24	030	18	00011000	CAN			Cancel
25	031	19	00011001	EM			End of Medium
26	032	1A	00011010	SUB			Substitute
27	033	1B	00011011	ESC			Escape
28	034	1C	00011100	FS			File Separator
29	035	1D	00011101	GS			Group Separator
30	036	1E	00011110	RS			Record Separator
31	037	1F	00011111	US			Unit Separator

ASCII-tecken (Teckenkoderna 32-127)

Tecknen 32-127 är gemensam för alla varianter av ASCII-tabeller, de kallas även för de skrivbara tecknen i ASCII. Nästan alla tecken förekommer på ditt tangentbord. Talet 127 representerar kommandot DEL.

DEC	OCT	HEX	BIN	Symbol	HTML nummer	HTML namn	Beskrivning
-----	-----	-----	-----	--------	-------------	-----------	-------------

32	040	20	00100000		 		Mellanslag
33	041	21	00100001	!	!		Utropstecken
34	042	22	00100010	"	"	"	Citationstecken (Citattecken)
35	043	23	00100011	#	#		Nummertecken (staket)
36	044	24	00100100	\$	$		Dollartecken
37	045	25	00100101	%	%		Procenttecken
38	046	26	00100110	&	&	&	Ochtecken
39	047	27	00100111	'	'		Apostrof
40	050	28	00101000	((Vänster parentes
41	051	29	00101001))		Höger parentes
42	052	2A	00101010	*	*		Asterisk
43	053	2B	00101011	+	+		Plus
44	054	2C	00101100	,	,		Komma
45	055	2D	00101101	-	-		Minus
46	056	2E	00101110	.	.		Punkt
47	057	2F	00101111	/	/		Divisionstecken (Slash)
48	060	30	00110000	0	0		Siffran 0
49	061	31	00110001	1	1		Siffran 1
50	062	32	00110010	2	2		Siffran 2
51	063	33	00110011	3	3		Siffran 3
52	064	34	00110100	4	4		Siffran 4
53	065	35	00110101	5	5		Siffran 5
54	066	36	00110110	6	6		Siffran 6

55	067	37	00110111	7	7		Siffran 7
56	070	38	00111000	8	8		Siffran 8
57	071	39	00111001	9	9		Siffran 9
58	072	3A	00111010	:	:		Kolon
59	073	3B	00111011	;	;		Semikolon
60	074	3C	00111100	< </td >	<	<	Mindre än
61	075	3D	00111101	=	=		Lika med
62	076	3E	00111110	>	>	>	Större än
63	077	3F	00111111	?	?		Frågetecken
64	100	40	01000000	@	@		Snabel-A (At)
65	101	41	01000001	A	A		Versalt A
66	102	42	01000010	B	B		Versalt B
67	103	43	01000011	C	C		Versalt C
68	104	44	01000100	D	D		Versalt D
69	105	45	01000101	E	E		Versalt E
70	106	46	01000110	F	F		Versalt F
71	107	47	01000111	G	G		Versalt G
72	110	48	01001000	H	H		Versalt H
73	111	49	01001001	I	I		Versalt I
74	112	4A	01001010	J	J		Versalt J

75	113	4B	01001011	K	K		Versalt K
76	114	4C	01001100	L	L		Versalt L
77	115	4D	01001101	M	M		Versalt M
78	116	4E	01001110	N	N		Versalt N
79	117	4F	01001111	O	O		Versalt O
80	120	50	01010000	P	P		Versalt P
81	121	51	01010001	Q	Q		Versalt Q
82	122	52	01010010	R	R		Versalt R
83	123	53	01010011	S	S		Versalt S
84	124	54	01010100	T	T		Versalt T
85	125	55	01010101	U	U		Versalt U
86	126	56	01010110	V	V		Versalt V
87	127	57	01010111	W	W		Versalt W
88	130	58	01011000	X	X		Versalt X
89	131	59	01011001	Y	Y		Versalt Y
90	132	5A	01011010	Z	Z		Versalt Z
91	133	5B	01011011	[[Vänster hakparentes
92	134	5C	01011100	\	\		Backslash
93	135	5D	01011101]]		Höger hakparentes
94	136	5E	01011110	^	^		Utelämningstecken (Caret)
95	137	5F	01011111	_	_		Horisontell linje
96	140	60	01100000	`	`		Grav (Acute accent)
97	141	61	01100001	a	a		Gement a

98	142	62	01100010	b	b		Gement b
99	143	63	01100011	c	c		Gement c
100	144	64	01100100	d	d		Gement d
101	145	65	01100101	e	e		Gement e
102	146	66	01100110	f	f		Gement f
103	147	67	01100111	g	g		Gement g
104	150	68	01101000	h	h		Gement h
105	151	69	01101001	i	i		Gement i
106	152	6A	01101010	j	j		Gement j
107	153	6B	01101011	k	k		Gement k
108	154	6C	01101100	l	l		Gement l
109	155	6D	01101101	m	m		Gement m
110	156	6E	01101110	n	n		Gement n
111	157	6F	01101111	o	o		Gement o
112	160	70	01110000	p	p		Gement p
113	161	71	01110001	q	q		Gement q
114	162	72	01110010	r	r		Gement r
115	163	73	01110011	s	s		Gement s
116	164	74	01110100	t	t		Gement t
117	165	75	01110101	u	u		Gement u
118	166	76	01110110	v	v		Gement v
119	167	77	01110111	w	w		Gement w

120	170	78	01111000	x	x		Gement x
121	171	79	01111001	y	y		Gement y
122	172	7A	01111010	z	z		Gement z
123	173	7B	01111011	{	{		Vänster krullparentes
124	174	7C	01111100		|		Vertikal linje
125	175	7D	01111101	}	}		Höger krullparentes
126	176	7E	01111110	~	~		Tilde
127	177	7F	01111111				Delete

Utökade ASCII-tecken (Teckenkoderna 128-255)

Finns flera varianter av 8-bitars ASCII men i denna tabell visas dem enligt ISO 8859-1, även kallad "Latin-1". Denna tabell innehåller de svenska tecknen å ä ö, både som versaler och gemener. Plats 129-159 består av Microsoft® Windows Latin-1 utökade tecken.

DE C	OC T	HE X	BIN	S y m b o l	HTML nummer	HTML namn	Beskrivning
128	200	80	10000000	€	€	€	Euro
129	201	81	10000001				
130	202	82	10000010	'	‚	‚	
131	203	83	10000011	f	ƒ	ƒ	
132	204	84	10000100	"	„	„	
133	205	85	10000101	...	…	…	Horisontal ellipsis
134	206	86	10000110	†	†	†	Dagger

135	207	87	10000111	‡	‡	‡	Double dagger
136	210	88	10001000	^	ˆ	ˆ	Circumflex
137	211	89	10001001	‰	‰	‰	Promille
138	212	8A	10001010	Š	Š	Š	
139	213	8B	10001011	‹	‹	‹	Enkelt vinklat citat vänster
140	214	8C	10001100	Œ	Œ	Œ	
141	215	8D	10001101				
142	216	8E	10001110	Ž	Ž		
143	217	8F	10001111				
144	220	90	10010000				
145	221	91	10010001	'	‘	‘	Citat enkelt vänster
146	222	92	10010010	'	’	’	Citat enkelt höger
147	223	93	10010011	"	“	“	Citat dubbel vänster
148	224	94	10010100	"	”	”	Citat dubbel höger
149	225	95	10010101	•	•	•	
150	226	96	10010110	-	•	–	Enkel dash
151	227	97	10010111	-	—	—	Dubbel dash
152	230	98	10011000	~	˜	˜	
153	231	99	10011001	™	™	™	Varumärke (Trademark)
154	232	9A	10011010	š	š	š	

155	233	9B	10011011	›	›	›	Enkelt vinklat citat höger
156	234	9C	10011100	œ	œ	œ	
157	235	9D	10011101				
158	236	9E	10011110	ž	ž		
159	237	9F	10011111	ÿ	Ÿ	Ÿ	
160	240	A0	10100000		 	 	Icke-brytande blanksteg
161	241	A1	10100001	¡	¡	¡	Upp-och-nedvänt utropstecken
162	242	A2	10100010	¢	¢	¢	Centtecken
163	243	A3	10100011	£	£	£	Pundtecken
164	244	A4	10100100	¤	¤	¤	Allmän valutasymbol
165	245	A5	10100101	¥	¥	¥	Yentecken
166	246	A6	10100110	¦	¦	¦	Pipe/Vertikalt brutet streck
167	247	A7	10100111	§	§	§	Paragrafsymbol
168	250	A8	10101000	¨	¨	¨	Trema (dieresis, omljud)
169	251	A9	10101001	©	©	©	Copyright-tecken
170	252	AA	10101010	ª	ª	ª	Feminine ordinal
171	253	AB	10101011	«	«	«	Dubbla vinkelcitationstecken (gåsögon), vänster
172	254	AC	10101100	¬	¬	¬	Logiskt icke-tecken
173	255	AD	10101101		­	­	Kort talstreck

174	256	AE	10101110	®	®	®	Registrerat varumärke
175	257	AF	10101111	—	¯	¯	Makron
176	260	B0	10110000	°	°	°	Gradtecken
177	261	B1	10110001	±	±	±	Plus-minus -tecken
178	262	B2	10110010	²	²	²	Upphöjt till 2
179	263	B3	10110011	³	³	³	Upphöjt till 3
180	264	B4	10110100	'	´	´	Accent
181	265	B5	10110101	μ	µ	µ	Mikrotecken
182	266	B6	10110110	¶	¶	¶	Paragraftecken
183	267	B7	10110111	·	·	·	Mittenpunkt (skalärprodukt)
184	270	B8	10111000	,	¸	¸	Cedilj
185	271	B9	10111001	¹	¹	¹	Mikro
186	272	BA	10111010	°	º	º	Masculine ordinal
187	273	BB	10111011	»	»	»	Dubbla vinkelcitationstecken (gåsögon), höger
188	274	BC	10111100	¹ / ₄	¼	¼	En fjärdedel
189	275	BD	10111101	¹ / ₂	½	½	En halv
190	276	BE	10111110	³ / ₄	¾	¾	Tre fjärdedelar
191	277	BF	10111111	¿	¿	¿	Upp-och-nedvänt frågetecken
192	300	C0	11000000	À	À	À	A med grav accent

193	301	cl	11000001	Á	Á	Á	A med akut accent
194	302	C2	11000010	Â	Â	Â	A med cirkumflex
195	303	C3	11000011	Ã	Ã	Ã	A med tilde
196	304	C4	11000100	Ä	Ä	Ä	A med trema
197	305	C5	11000101	Å	Å	Å	A med ring
198	306	C6	11000110	Æ	Æ	Æ	Ligatur A+E
199	307	C7	11000111	Ç	Ç	Ç	C med cedilj
200	310	C8	11001000	È	È	È	E med grav accent
201	311	C9	11001001	É	É	É	E med akut accent
202	312	CA	11001010	Ê	Ê	Ê	E med cirkumflex
203	313	CB	11001011	Ë	Ë	Ë	E med trema
204	314	CC	11001100	Ì	Ì	Ì	I med grav accent
205	315	CD	11001101	Í	Í	Í	I med akut accent
206	316	CE	11001110	Î	Î	Î	I med cirkumflex
207	317	CF	11001111	Ï	Ï	Ï	I med trema
208	320	D0	11010000	Ð	Ð	Ð	(ETH)
209	321	D1	11010001	Ñ	Ñ	Ñ	N med tilde

210	322	D2	11010010	Ò	Ò	Ò	O med grav accent
211	323	D3	11010011	Ó	Ó	Ó	O med akut accent
212	324	D4	11010100	Ô	Ô	Ô	O med cirkumflex
213	325	D5	11010101	Õ	Õ	Õ	O med tilde
214	326	D6	11010110	Ö	Ö	Ö	O med trema
215	327	D7	11010111	×	×	×	Multiplikationstecken
216	330	D8	11011000	Ø	Ø	Ø	Snedstruket O
217	331	D9	11011001	Ù	Ù	Ù	U med grav accent
218	332	DA	11011010	Ú	Ú	Ú	U med akut accent
219	333	DB	11011011	Û	Û	Û	U med cirkumflex
220	334	DC	11011100	Ü	Ü	Ü	U med trema/umlaut
221	335	DD	11011101	Ý	Ý	Ý	Tyskt dubbel-s
222	336	DE	11011110	Þ	Þ	Þ	Isländska THORN-tecknet
223	337	DF	11011111	ß	ß	ß	Tyskt dubbel-S
224	340	E0	11100000	à	à	à	a med grav accent
225	341	E1	11100001	á	á	á	a med akut accent

226	342	E2	11100010	â	â	â	a med cirkumflex
227	343	E3	11100011	ã	ã	ã	a med tilde
228	344	E4	11100100	ä	ä	ä	a med trema
229	345	E5	11100101	å	å	å	a med ring
230	346	E6	11100110	æ	æ	æ	aelig
231	347	E7	11100111	ç	ç	ç	c med cedilj
232	350	E8	11101000	è	è	è	e med grav accent
233	351	E9	11101001	é	é	é	e med akut accent
234	352	EA	11101010	ê	ê	ê	e med cirkumflex
235	353	EB	11101011	ë	ë	ë	e med trema
236	354	EC	11101100	ì	ì	ì	i med grav accent
237	355	ED	11101101	í	í	í	i med akut accent
238	356	EE	11101110	î	î	î	i med cirkumflex
239	357	EF	11101111	ï	ï	ï	i med trema
240	360	F0	11110000	ð	ð	ð	eth
241	361	F1	11110001	ñ	ñ	ñ	n med tilde
242	362	F2	11110010	ò	ò	ò	o med grav accent
243	363	F3	11110011	ó	ó	ó	o med akut accent
244	364	F4	11110100	ô	ô	ô	o med cirkumflex
245	365	F5	11110101	õ	õ	õ	o med tilde
246	366	F6	11110110	ö	ö	ö	o med trema
247	367	F7	11110111	÷	÷	÷	Divisionstecken

248	370	F8	11111000	ø	ø	ø	Snedstruket o
249	371	F9	11111001	ù	ù	ù	u med grav accent
250	372	FA	11111010	ú	ú	ú	u med akut accent
251	373	FB	11111011	û	û	û	u med cirkumflex
252	374	FC	11111100	ü	ü	ü	u med trema/umlaut
253	375	FD	11111101	ý	ý	ý	y med akut accent
254	376	FE	11111110	þ	þ	þ	Isländska thorn-tecknet
255	377	FF	11111111	ÿ	ÿ	ÿ	y med trema

NUL	Används som utfyllnadstecken, skall inte sammankopplas med blanktecknet.
SOH	Start Of Header, för att markera början av headern, som är adress och kontrolltecken innan
STX	Start Of Text, markerar var data börjar och slutet på headern.
ETX	End Of Text, markerar var data slutar, data är mellan STX och ETX.
EOT	End Of Transmission, anger slutet på en överföring
ENQ	ENQuiry, är en begäran om svar
ACK	ACKnowledge, är ett positivt svar på att data som mottagits är korrekt mottagen.
BEL	Används för att påkalla uppmärksamhet, en ljudsignal är oftast förknippad med detta tecken
BS	BackSpace, för att backa ett tecken.
HT	Horizontal Tab, en tabuleringsförflyttning, hur många positioner som flyttas beror på inställni
LF	Line Feed, radmatning, medför att nästa tecken kommer på raden under det föregående.
VT	Vertikal Tab, ett antal radmatningar baserade på inställning på utrustningen.
FF	Form Feed, sidmatning till nästa sida.
CR	Carriage Return, vagnretur, medför att nästa tecken skrivs i första positionen på raden.

SO	Shift Out, tecknen som kommer efter detta tecken skall tolkas efter en alternativ tabell än de
SI	Shift In, tecknen som kommer efter detta skall tolkas enligt standard tabellen.
DLE	Data Link Escape, för att ge styrkoder en annan betydelse, är protokollberoende.
DC1	Device Control 1, kan styra olika funktioner i en utrustning, beroende på denna utrustning be flödesstyrningstecken och betyder fortsatt sända.
DC2	Device Control 2, kan styra olika funktioner i en utrustning, beroende på denna utrustning be
DC3	Device Control 3, kan styra olika funktioner i en utrustning, beroende på denna utrustning be flödesstyrningstecken och betyder sluta sända.
DC4	Device Control 4, kan styra olika funktioner i en utrustning, beroende på denna utrustning be
NAK	Negativ AcKnowledge, är ett negativt svar på att data som mottagits är inte är korrekt mottag
SYN	SYNchronous idle, för att synkronisera utrustning vid synkron överföringsteknik.
ETB	End of Transmission Block, slut på ett logiskt block, mer förväntas komma.
CAN	CANcel, cancelera den data som mottagits
EM	End of Medium, slut på ett fysiskt block
SUB	SUBstitute, ersätter ett annat tecken som inte är giltigt med denna tecken tabell.
ESC	ESCape, ger efterföljande tecken en speciell betydelse, används i terminaler för att till exem
DEL	Delete, används för stansade pappersremсор för att markera ett stansfel.
FS	File Separator, avgränsare
GS	Group Separator, avgränsare
RS	Record Separator, avgränsare
US	Unit Separator, avgränsare

1.3.2.2. EBCDIC

De flesta av datorerna använder 8 bitar internt och använder en annan kod för att representera våra tecken, EBCDIC (Extended Binary-Coded Deciamal Interchange Code). Denna skapades också i USA och fick därför samma svaghet som ASCII , de landsspecifika

tecknen. Vissa datorleverantörer valde att även använda denna kod i sin datakommunikation, till exempel IBM.

Dec	Hex	Code	Dec	Hex	Code	Dec	Hex	Code	Dec	Hex	Code
0	00	NUL	32	20		64	40	space	96	60	-
1	01	SOH	33	21		65	41		97	61	/
2	02	STX	34	22		66	42		98	62	
3	03	ETX	35	23		67	43		99	63	
4	04		36	24		68	44		100	64	
5	05	HT	37	25	LF	69	45		101	65	
6	06		38	26	ETB	70	46		102	66	
7	07	DEL	39	27	ESC	71	47		103	67	
8	08		40	28		72	48		104	68	
9	09		41	29		73	49		105	69	
10	0A		42	2A		74	4A	[106	6A	
11	0B	VT	43	2B		75	4B	.	107	6B	,
12	0C	FF	44	2C		76	4C	<	108	6C	%
13	0D	CR	45	2D	ENQ	77	4D	(109	6D	_
14	0E	SO	46	2E	ACK	78	4E	+	110	6E	>
15	0F	SI	47	2F	BEL	79	4F	!	111	6F	?
16	10	DLE	48	30		80	50	&	112	70	
17	11		49	31		81	51		113	71	
18	12		50	32	SYN	82	52		114	72	
19	13		51	33		83	53		115	73	
20	14		52	34		84	54		116	74	

21	15		53	35		85	55		117	75	
22	16	BS	54	36		86	56		118	76	
23	17		55	37	EOT	87	57		119	77	
24	18	CAN	56	38		88	58		120	78	
25	19	EM	57	39		89	59		121	79	'
26	1A		58	3A		90	5A	!]]	122	7A	:
27	1B		59	3B		91	5B	\$	123	7B	#
28	1C	IFS	60	3C		92	5C	*	124	7C	@
29	1D	IGS	61	3D	NAK	93	5D)	125	7D	'
30	1E	IRS	62	3E		94	5E	;	126	7E	=
31	1F	IUS	63	3F	SUB	95	5F	^	127	7F	"

Dec	Hex	Code	Dec	Hex	Code	Dec	Hex	Code	Dec	Hex	Code
128	80		160	A0		192	C0	{	224	E0	\
129	81	a	161	A1	~	193	C1	A	225	E1	
130	82	b	162	A2	s	194	C2	B	226	E2	S
131	83	c	163	A3	t	195	C3	C	227	E3	T
132	84	d	164	A4	u	196	C4	D	228	E4	U
133	85	e	165	A5	v	197	C5	E	229	E5	V
134	86	f	166	A6	w	198	C6	F	230	E6	W
135	87	g	167	A7	x	199	C7	G	231	E7	X
136	88	h	168	A8	y	200	C8	H	232	E8	Y
137	89	i	169	A9	z	201	C9	I	233	E9	Z
138	8A		170	AA		202	CA		234	EA	
139	8B		171	AB		203	CB		235	EB	

140	8C		172	AC		204	CC		236	EC	
141	8D		173	AD		205	CD		237	ED	
142	8E		174	AE		206	CE		238	EE	
143	8F		175	AF		207	CF		239	EF	
144	90		176	B0		208	D0	}	240	F0	0
145	91	j	177	B1		209	D1	J	241	F1	1
146	92	k	178	B2		210	D2	K	242	F2	2
147	93	l	179	B3		211	D3	L	243	F3	3
148	94	m	180	B4		212	D4	M	244	F4	4
149	95	n	181	B5		213	D5	N	245	F5	5
150	96	o	182	B6		214	D6	O	246	F6	6
151	97	p	183	B7		215	D7	P	247	F7	7
152	98	q	184	B8		216	D8	Q	248	F8	8
153	99	r	185	B9		217	D9	R	249	F9	9
154	9A		186	BA		218	DA		250	FA	
155	9B		187	BB		219	DB		251	FB	
156	9C		188	BC		220	DC		252	FC	
157	9D		189	BD		221	DD		253	FD	
158	9E		190	BE		222	DE		254	FE	
159	9F		191	BF		223	DF		255	FF	

De svenska tecknen ersätter på dessa platser de ursprungliga i EBCDIC-tabellen. Effekten är att vi inte kan använda dessa ersatta tecken, men det har ansetts att dessa inte används i någon större utsträckning i Sverige.

	010	011	011	110	110
	1	0	1	0	1

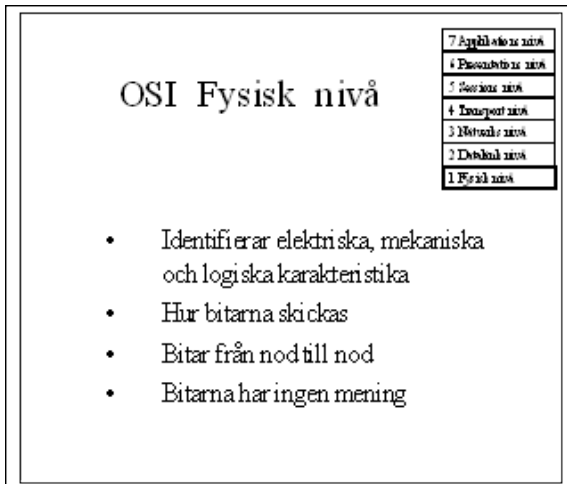
000 0				ä	å
101 0		ö			
101 1	Å		Ä		
110 0			Ö		

Textmassan ökar

- **Fader vår**
– **65 ord**
- **Amerikanska självständighetsförklaringen**
– **275 ord**
- **EU's regler för export av ankägg**
– **27962 ord**

Kapitel 2: Nivå 1, den fysiska nivån

2.1. Uppgift



Ansvarar för att överföra bitar från en kommunikationsenhet till en annan, som är "fysiskt" sammankopplade. Fysiskt inom situationstecken därför att det inte måste vara ett fysiskt media idag som sammanbinder kommunikationsenheterna, det finns radiovågor och infrarött ljus som även används för detta ändamål. Bitarna har ingen inneboende mening, som till exempel hur de skall grupperas för att vara data eller information.

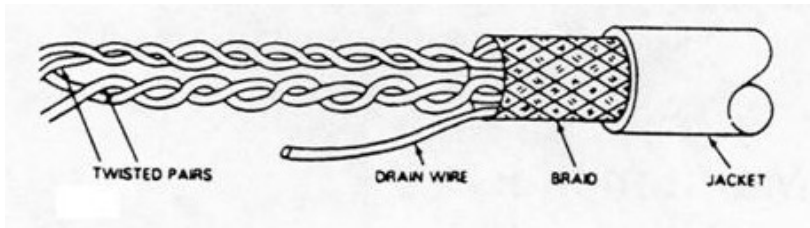
Den fysiska nivån beskriver allt det som hör ihop med att koppla samman hårdvaran, både dess elektriska, mekaniska och logiska karakteristiska. Definierar hur den fysiska kabeln skall kopplas till nätverkskortet via kontakter, alltså det fysiska gränssnittet. Till exempel hur många pinnar som finns i kontakten, och vad varje pinne har för funktion, hur måtten på kontakten skall vara så att hane och hona passar samman och inte ger kontaktproblem. Definierar vilken överföringsteknik som skall användas i överföringen. Definierar data encoding och bitsynkronisering, identifierar vad som är en 1 och vad som är 0 och elektrisk eller optisk karakteristik för dessa.

2.2. Medium

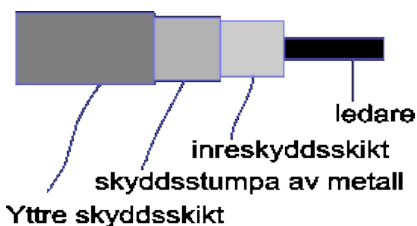
Kabel är det medium som mest är förknippat med datakommunikation, men överföringen kan också göras med radiovågor till exempel satelitlänkar, och ljus, som infrarött ljus. Kabel förknippas också oftast med kopparkabel, men idag finns det även fiberoptiska kablar som används för kommunikation. Överföringshastigheten beror delvis av kabelns fysiska karakteristik. En fiberoptisk kabel använder laserljus som medium och kan överföra stora mängder data, medan en kopparkabel har betydligt lägre kapacitet. För att kunna överföra en signal måste det finnas ett signalmedia och ett returmedia, normalt är detta utfört med två kablar.

Det finns olika typer av kopparkablar som har olika egenskaper både elektriskt och mekaniskt och hur de olika trådarna ligger inuti kabeln. Den mest vanliga kabeln idag för telefonnätet är en partvinnad kabel. Denna har blivit mest vanlig idag även för LAN, och den finns för olika överförings hastigheter, typ 3 är för max 10 Mbs och typ 5 är för 100 Mbs. Partvinnad kabel innebär att signaltråden och returtråden ligger tvinnade runt om varandra. Detta medför att störningar får mindre möjlighet att påverka signalen. Varje tråd har ett isoleringsskikt som

isolerar den från den andra tråden, och utanpå dessa finns ett yttre isolerings- och skyddsskikt. Det finns partvinnade kablar med en skyddsstrumpa mellan det yttre skyddsskiktet och de båda trådarna. Denna skyddsstrumpa är kopplad till jord, som ytterligare skyddar mot störningar. De partvinnade trådarna kan vara tillsammans med flera andra par i en och samma kabel också.



En annan kabeltyp som används från början till LAN-kablar är koaxkabel, och även denna finns i olika typer med olika kapaciteter.



En koax kabel har endast en ledare eller tråd som signalen använder, runt denna finns först ett isoleringsskikt och sedan en skyddsstrumpa och ytterst ett yttre isoleringsskikt. Det är skyddsstrumpan som är returledning för signalen. Koaxkablar för LAN finns i två sorter, en som kallas thinnet, chipnet eller thin black, har också beteckningen RG-58, och som har en begränsad användbar längd men är ganska smidig att hantera. En annan som kallas för fat yellow eller RG9, som har en betydligt längre användbar längd men är mycket styvare och besvärligare att hantera och är dyrare än thin black kabeln.

Fiberoptiska kablar har en kärna av glas, speciellt glas som är böjligt. Denna kärna omges av ett speciellt skyddsskikt av plast, och därutöver det yttre skyddsskiktet. Det krävs speciell utrustning för att skarva och ansluta fiberkablar, och de är enkelriktade så det krävs ett par för att kommunicera. Fiberkablar är betydligt dyrare än kopparkablar, men har mycket högre kapacitet, och är inte känslig för störningar som kopparkablarna är. LAN-kablar har benämningar som anger hastighet, signaltyp och maxlängd på segmentet.

2.2.1.1. 10Base5

Nätverk som benämns 10Base5 använder fat yellow koaxkabel, RG9, som medium och segmenten får vara max 500m långa, max 5 segment och max 100 noder per segment, men totalt max 200 noder. På den kabeln finns ringar var 2,5m för att markera minimiavståndet mellan två noder, och man rekommenderas till att ansluta till dessa markeringar för att inte bryta mot reglerna. Färgen gul rekommenderas i reglerna för att skilja den från starkströmskabel, men detta följs inte av alla tillverkare. Det yttre gula skiktet består av olika material beroende på om den skall vara inomhus eller utomhus, och vilken brandklass som används. Kabeln måste termineras i båda ändar med 50 Ω motstånd, och skärmen skall vara kopplad till jord på endast ett ställe per segment. (Ljushastigheten i RG9 är 0.77)

2.2.1.2. 10Base2

Nätverk som benämns 10Base2 använder thin black koaxkabel, RG58, som medium och segmenten får vara max 185m långa, max 5 segment och max 30 noder per segment, men

totalt max 100 noder. Det yttre skiktet består av olika material beroende på om den skall vara inomhus eller utomhus, och vilken brandklass som används. Kabeln måste termineras i båda ändrar med 50 Ω motstånd, och skärmen skall vara kopplad till jord på endast ett ställe per segment. (Ljushastigheten i RG58 är 0.65)

2.2.1.3. 10Base-T

Nätverk som benämns 10Base-T använder partvinnad kabel som medium och använder en hub som noderna kopplas till med ett max avstånd av 100m mellan nod och hub. Det finns inga begränsningar på antal segment eftersom detta typ av nätverk fysiskt blir ett stjärnnätverk men logiskt är ett bussnätverk. (Ljushastigheten i kat 5 är 0.59)

2.2.1.4. 10Base-F

Nätverk som benämns 10Base-F använder optisk fiberkabel som medium och är definierad i tre typer, FP, FB och FL. FB är en typ som är tänkt att koppla samman två nätverk eller segment som kan befinna sig på upp till 2 km från varandra, ingen arbetsstation kan kopplas till denna typ. FL och FP är de typer som används när arbetsstationer kopplas in, där skillnaden är att FP har en radie på 500 m och FL en radie på 2 km, båda bildar ett fysiskt stjärnnätverk.

2.2.1.5. 10Broad36

Nätverk som benämns 10Broad36 använder oftast koaxkabel, och en annan teknik, bredband. Längden på segmentet är 3600m.

2.2.2. Störningar

Störningar på kablar kan vara av olika sorter, dels finns det störningar som beror på fysiska/elektriska karakteristiska, dels som beror på miljön som kabeln finns i. Kablar av koppar utsätts lätt för störningar av olika slag, och hur kablarna drages i rummet har mycket stor betydelse för störningskänsligheten. Men även typ av kabel spelar mycket stor roll.

- Brus (termisk)
- Dämpning
- Dämpningsdistorsion
-
- Grupplöptidsdistorsion
 - Fasjitter
 - Fassprång
 - Överhörning/induktion
 - Eko/stående våg

Det finns brus som är av flera typer, är vanligt på telefonledningar och orsakar bitfel. Alla kablar dämpar signalen, och det är detta som bl.a. påverkar hur lång en kabel kan vara mellan två kommunikationsenheter. Dämpningen kan vara olika för olika frekvenser, och kallas då för dämpningsdistorsion. Men även de olika frekvenserna kan överföras olika fort, grupplöptidsdistorsion, kan motverkas med filter. Fasjitter och fassprång är störningar som påverkar signalens fas.

Kablarna måste termineras korrekt annars uppstår eko, vilket innebär att signalen studsar

tillbaka från ändpunkten och påverkar de signaler som kommer efter. En kabel kan bli utsatt för induktion, eller överhörning, från andra kablar som ligger parallellt, och därför rekommenderas att signal kablar dragas skilt från strömförande kablar, ca 1 meter från varandra.

Kablarna skall skyddas från mekaniskt slitage och man skall försöka ha så få anslutningspunkter som möjligt. Varje anslutningspunkt är en möjlig felkälla genom att skapa dålig kontakt som ger impedansförändringar i kabeln. Kablarnas yttre skyddslikt kan vara gjort av olika material och har olika egenskaper vid brand, det är därför mycket viktigt att man installerar rätt kabeltyp för de utrymmen som kabeln skall dragas genom.

En del störningar kan förebyggas genom en bra planering av nätverkets installation och funktion, som till exempel antalet anslutningspunkter, att förhindra parallellitet med strömförande kablar. Använd väl kända och beprövade utrustningar och dimensionera dessa för det befintliga kapacitetsbehovet, men även att en ökning av kapacitetsbehovet kan göras utan alltför stora förändringar eller avbrott.

2.2.3. Val av medium

- Kostnader
- Hastighet och kapacitet
- Tillgänglighet
- Expansion
- Felfrekvens
- Säkerhet
- Avstånd
- Underhåll
- Miljö

Vilken typ av media som passar bäst beror på olika faktorer som man måste klargöra. Det viktiga är att göra en noggrann kartläggning över alla faktorer som man kan se påverkar valet, och väga dessa mot varandra, eftersom en del faktorer står i visst motsatsförhållande till varandra. Till exempel kostnader och hastighet/kapacitet, det blir dyrare med högre hastighet, men det kan uppvägas av andra faktorer som säkerhet, expansion och underhåll.

Exempel på frågeställningar:

- Installationskostnader, kostnad för kontakter, vad kostar det att expandera ett befintligt nätverk när detta behov uppstår.
- Kan samverka med andra typer av kommunikation som telefoner eller kabel-TV vara önskvärd eller ett måste, finns det redan ett befintligt nätverk.
- Vilken kabel kräver den utrustning som finns, kommer denna utrustning att bytas ut inom en snar framtid. Vilken säkerhet krävs för information som överföres på nätverket, fiberkabel går inte att avlyssna.

Det finns anledning att försöka se så långt in i framtiden som möjligt, och inte enbart på dagens behov av nätverk. Det kan bli onödigt dyrt att expandera om man väljer fel. I detta dokument kommer att finnas ytterligare faktorer som kan påverka val av nätverk och det krävs en samlad bild av hur ett nätverk skall samverka till företagets bästa. Det är ett designjobb att göra ett bra fungerande nätverk, men grunden läggs i val av kabel.

	partvinnad	koax	fiberoptik	mikrovåg	radio	s a t e l i t
Tillgänglighet	bra	bra	bra	bra	möjlig stockning	godkä
Expansionsmöjligheter	godkänt	bra i lokala områden	bra	bra	bra	bra
Fel	godkänt	bra	bra	godkänt	godkänt	godkä
Säkerhet	godkänt	godkänt	bra	dålig	dålig	dålig
Avstånd	bra	godkänt	bra	bra	bra	bra
Miljö	godkänt	bra	bra	godkänt	godkänt	godkä

2.2.4. Hastigheter

Hastighet mäts i antal bitar som kan överföras per sekund, bps. Hastigheten påverkas av avståndet mellan kommunikationspunkterna och val av kabeltyp. Bandbredden som kabeln kan överföra påverkar också, och sambandet mellan en kanals bandbredd, brus och maximal teoretisk hastighet är: Max hastighet = $B \log_2 (1+S)$ bps. Där B = bandbredd, S = förhållandet signal- och brusstyrka.

Bandbredd är ett mått på hur många frekvenser, som kan överföras, eller annorlunda uttryckt, högsta frekvensen minus lägsta frekvensen. En telefon har en bandbredd på ca 3300 Hz, mellan 300 Hz och 3600 Hz.

Kabel för partvinnad tråd typ 5 har en maximal hastighet av 100 Mbs, fiberoptiska kablar har överföringshastigheter i Gbps området. Hastigheter i telefonnäten har utvecklats inom fasta intervaller som 75 bps, 150, 300, 600, 1200, 1800, 2400, 4800, 9600, 14400, 19200, 28800, 33600, 56k, 64k, 384k och 2 Mbps. För lokala nätverk finns det hastigheter från ? 200kbps till idag 100Mbps, och det forskas om Gbps. (när du läser detta kan Gbps vara ett faktum).

2.3. Signaleringsmetoder

För att överföra data på mediumet använder man sig av olika metoder. Val av metod kan ha stor betydelse för det nätverk som du skall bygga. Det finns olika sätt att representera den digitala informationen elektriskt på kabeln, det finns basbandsteknik och bredbandsteknik.

2.3.1. Basband

De digitala signalerna anpassas elektriskt men omvandlas inte utan överföringen blir pulser i form av fyrkantvågor, de digitala signalerna går direkt ut på kabeln. Denna teknik kan utnyttja

alla typer av kablar och använder kabelns hela bandbredd. Ettor och nollor kan kodas på olika sätt för att överföras på en basbandskabel. Man kan låta en etta vara spänning och en nolla frånvaro av spänning, spänningen kan vara positiv eller negativ. För att öka säkerheten i överföringen har olika tekniker tagits fram. Om frånvaro av spänning är en nolla så kan ju ett fel som gör att spänningen försvinner skapa felaktig data. NRZ (Non Return to Zero) är en metod som låter en etta vara en positiv spänning och en nolla en negativ spänning, detta innebär att det alltid skall finnas någon form av spänning. Nackdelen med NRZ är att om många bitar med samma tecken sändes efter varandra, så stannar signalen på en nivå och det ger synkroniseringsproblem. En metod som försöker komma till rätta med detta är Manchester encoding, där man låter spänningen skifta nivå mitt i ett tecken, varvid en etta är övergången till plus från minus och en nolla övergången till minus från plus. Då kan man synkronisera på denna transition (övergång). Det innebär att om två lika tecken följer efter varandra måste en transition göras mellan tecknen så att rätt övergång är i mitten av tecknet. Varför i mitten på tecknet? Obs! ett binärt tecken. Genom att känna av pulsen i mitten av tecknet så påverkar inte små förändringar i hastigheten tolkningen av tecknet. För att komma ifrån betydelsen av riktningen på transitionen har Differential Manchester Encoding framtagits, där transitioner i mitten av tecknet alltid görs för synkroniserings skull och en transition i början av tecknet betyder en nolla och frånvaron av en transition är en etta.

Basbandstekniken används i LAN-nätverk, och i korthållsmodem.

2.3.2. Bredband

I bredbandsteknik används en bärvåg med en bestämd frekvens, som datasignalen får modulera, och använder samma teknik som man gör vid radio och TV. Det innebär att en kanal finns kring bärvågen och flera kanaler kan finnas på samma kabel genom att det finns olika bärvågsfrekvenser. Kanalbredden är 6 Mhz på ett totalt 400 Mhz område enligt det amerikanska systemet CATV. När man utnyttjar denna teknik för LAN användes två kanaler, en för sändning och en för mottagning. Dessa kanaler kan vara på samma kabel eller på två olika kablar, men i båda fallen måste det finnas en Head-End enhet i änden av kabeln eller kablarna, som tar emot signalen på den inkommande kanalen från nodernas sändarsida och skickar ut signalen på den utgående kanalen till nodernas mottagarsida. Användes en kabel med olika kanaler så konverterar Head-End enheten till en annan kanal, och Head-End enheten kallas då för translator eller remodulator. Sändarkanalerna har lägre kanalfrekvens än mottagarkanalerna.

Bredbandsteknik används i FDDI (Fiber Distributed Data Interface) och på kabel-TV, och framtiden kommer att erbjuda LAN på samma kabel som TV använder.

2.4. Trafiksätt

2.4.1. Simplex - duplex

Vid kommunikation måste vissa regler följas för att det skall vara meningsfullt för båda parter. Vid ett samtal till exempel så är det inte mycket av kommunikationen som kommer fram om båda parter talar i munnen på varandra, utan man får prata en åt gången, en dialog har etablerats mellan parterna. Om det bara är den ena parten som pratar blir det en monolog. I datakommunikations sammanhang kallas överföring som sker endast i en riktning för simplex överföring, som till exempel rundradio och TV. Om båda kan överföra data till varandra kallas detta för duplex överföring. Duplex överföring innebär att båda kan sända och ta emot, och när båda parter kan sända och ta emot samtidigt kallas detta för full duplex. Det finns utrustning som både kan sända och ta emot men inte samtidigt och detta kallas för halv

duplex överföring. Exempel på full duplex är telefonen, där kan båda parter tala samtidigt och ta emot samtidigt, det kanske inte går att urskilja vad som sägs men det går. Halv duplex används i kommunikationsradio, där man har en speciell rutin för att bestämma vem som skall tala genom att när den som just talar är färdig, avslutar med "kom", och då kan den andre tala.

2.4.2. Accessmetoder

I datakommunikationssammanhang så är oftast flera kommunikationsenheter kopplade till samma medium och alla kan inte utnyttja detta på samma gång, och det finns olika sätt att få access till detta media. Man kan dela in de olika accessmetoderna i centraliserad styrning, slumpmässig styrning och distribuerad styrning.

2.4.2.1. Centraliserad

Centraliserad styrning är mest vanlig i terminalnätverk men kan förekomma i LAN. En central dator styr vilken terminal som får access till media för att sända eller ta emot data, och det finns några metoder som är användbara, till exempel avfrågning och tidsmultiplexering. Avfrågning eller poll kommer att beskrivas i OSI nivå 2 eftersom denna metod ligger på denna nivå, och multiplexering beskrivs senare i detta kapitel.

2.4.2.2. Slumpmässig

2.4.2.2.1. Csmacd

Csmacd är en förkortning för Carrier Sense Multipel Access with Collision Detect, alltså att många har access till kanalen, bärvågen avkänns och kollisioner upptäcks. Detta är ett trafiksätt som fungerar så att den som vill sända, först lyssnar på kanalen, om det finns signaler så betyder detta att kanalen är upptagen och väntar då tills kanalen blir ledig och startar då sändningen. På kanalen är många enheter anslutna och dessa befinner sig på olika avstånd från varandra, och eftersom det tar en viss tid för signalerna att färdas på kanalen så finns ju möjligheten att två (eller flera) vill sända samtidigt och finner då att det inte pågår någon trafik, och börjar sända. Det uppstår nu en kollision av dessa enheters sändning, detta upptäcks av någon på kanalen och denna sänder ett speciellt kollisionspaket. Detta upptäcks av alla, eftersom under pågående sändning avlyssnas samtidigt kanalen, och båda (alla) slutar att sända. Alla väntar en stund och sedan försöker de som vill sända på nytt. Eftersom denna väntan inte är lika lång för varje enhet så hinner någon att komma igång och de andra upptäcker detta och sänder inte. Att enheterna väntar olika länge är en algoritm som genererar en slumpmässig väntetid i varje enhet och eftersom den är slumpmässig så genererar inte två maskiner samma väntetid samtidigt, sannolikheten är mycket liten i alla fall.

Denna metod används i LAN av Ethernet typ. Ethernet är designat av Xerox, Digital Equipment och Intel för lokala nätverk. Denna typ av nätverk ger bra access till nätverket vid låga belastningar, men när det blir stor belastning blir kollisionerna fler och fler och det kommer till en mättnadsgräns. Varje enhet har ingen garanterad access till nätverket, så för vissa applikationer är detta en stor nackdel.

2.4.2.3. Distribuerad

Metoden utgår från någon form av turordning mellan anslutna enheter och kan göras fysiskt som i token ring, eller logiskt som i token bus och CSMA/CA.

2.4.2.3.1. Token

Med token menas att man använder en token som vandrar mellan enheterna på nätverket, och när en enhet skall sända måste den invänta att denna token skall komma och om den är ledig så kan data kopplas på denna token. En token är ett speciellt paket som är 24 bitar lång, 3 oktetter. Det finns två typer av token nätverk, token ring och token buss. Varje nod på nätverket läser in varje paket och sänder det sedan ut igen till nästa nod, vilket medför att alla noder måste fungera, annars bryts kedjan.

Ett token nätverk kan beskrivas som en järnväg med stationer och ett lokomotiv som åker runt och stannar vid alla stationer och hämtar eventuellt en vagn och lämnar tillbaks den vid nästa stopp på samma station. I token ring får max en vagn finnas kopplad till loket, och varje stations vagn måste hakas av när den passerar stationen. Det får endast finnas ett lok på spåret. FDDI är en variant där en nod får sända flera paket efter varandra, och "loket" kan ta flera vagnar.

När en nod har något att sända till en annan nod så inväntar den att ett tokenpaket skall anlända, och om token är ledig så kan data "hängas" på. Token plus data skickas nu som ett paket till nästa nod i kedjan, denna läser in detta paket och tittar på adressen (destination adress = egen adress) om det är avsett för denna nod. Om paketet inte är för denna nod skickas det ut igen till nästa nod, men om det är avsett för denna nod så kontrolleras paketet och kopieras in till noden, sedan skickas paketet ut igen och en flagga sätts i det utgående paketet om det är mottaget OK. Nu kommer detta paket att gå igenom alla noder i nätverket tills det kommer tillbaks till den nod som sände paketet. Denna nod läser också in paketet och upptäcker att det är dennes avsända paket (source adress = egen adress), kontrollerar hur det har mottagits och skickar ut ett paket till nästa nod med endast token och där flaggan är satt att den är ledig. Detta innebär att ett paket måste komma tillbaks till den nod som sände det och det är denna som "hakar" av data som den sänt så att nästa nod kan sända. I token ring får en nod endast skicka ett datapaket åt gången, blir det något fel får det felaktiga paketet skickas om igen när token kommer tillbaks och är ledig. I token nätverk så kan man garantera en viss access till nätverket genom att alla enheter får regelbundet chans att sända när de får en ledig token. En ledig token kommer med jämna mellanrum till varje nod, tiden mellan varje token beror på hur många noder som finns på nätverket.

Det krävs övervakning i ett token nätverk så att endast en token finns, men också att det finns en token. Till denna övervakning används en av noderna som inte gör annat än detta, tar bort extra token och sätter in en ny om den gamla är borta.

2.4.2.3.2. CSMA/CA

Detta är en förkortning för Carrier Sense Multiple Access/Collision Avoidance, att många har access till kanalen, bärvågen avkänns och kollisioner undviks. Denna metod är mycket lik CSMA/CD men undviker kollisioner genom att det finns en prioritetsordning upprättad mellan enheterna, och när en enhet vill sända, lyssnar på kanalen och väntar sedan en viss tid som är kopplad till prioriteten, och om kanalen fortfarande är ledig när väntetiden är slut så kan sändning påbörjas. Turordningen kan vara svår att vidmakthålla när ingen trafik pågår eftersom det är avslutningen av en sändning som triggar och synkroniserar prioriteringen. Denna typ av accessmetod används i LAN som använder radiokommunikation, s.k. trådlösa nätverk.

2.5. Gränssnitt

Gränssnitt är en beskrivning på hur två olika "världar" skall kunna utbyta data, till exempel hur fysiska enheter skall kopplas samman, eller hur utbyta information mellan två program. Till exempel en stickkontakt på kabeln till datorn måste passa i eluttaget i väggen. Eller hur

informationen presenteras på bildskärmen från ett applikationsprogram, användaren måste förstå vad som menas. Ta stickkontakten som exempel, om du reser utomlands och har tagit med dig ditt resestrykjärn, så kanske du inte kan sätta in el-kontakten i väggen på hotellet därför att de inte har samma passform, det finns speciella omvandlingskontakter för detta ändamål att köpa, gränssnittsomvandlare.

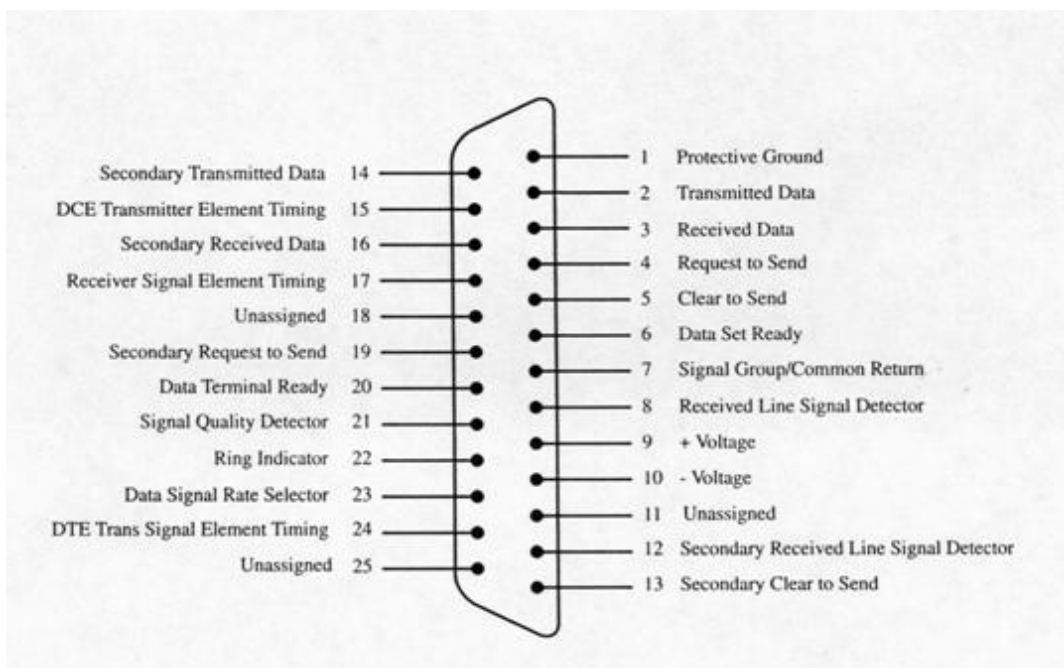
Inom datakommunikationsområdet finns det otaliga typer av maskiner som skall kopplas samman, och i början använde varje leverantör sina speciella kontakter, och det gick bra så länge som dessa kopplades ihop. Men när olika leverantörers maskiner skulle kopplas samman då måste man komma överens om hur gränssnittet skall vara. Det finns flera standardiseringsorganisationer som internationellt har angivit hur olika gränssnitt skall vara. Det finns ISO (Internationella Standardiserings Organisationen) en sammanslutning av nationella standardiseringsorganisationer, ITU-T (International Tele Unionen, före detta CCITT (Comité Consultatif International de Telegraphie et Telephonique)), IEEE (Institute of Electrical and Electronics Engineers), som har fastlagt olika normer och regler. CCITT's standarder är de som oftast refereras till och är beskrivna i dokument som har benämningen till exempel V.24, där V indikerar att det är ett dokument som beskriver analoga telefonnät. Om det står ett X framför istället för V så beskriver dokumentet digitala telefonnät.

2.5.1. Kontakter

Det mest tydliga av ett gränssnitt är de olika kontakter som finns mellan de kommunicerande enheterna. Tyvärr finns det inte en kontakt utan ett antal beroende på överföringshastighet, typ av kommunikation och flera andra saker som avgör vilken kontakt som skall användas.

2.5.1.1. ISO 2110

Detta är en fysisk och mekanisk beskrivning på en 25-polig kontakt som är den mest använda kontakten för låghastighetsöverföringar, vanlig upp till 56 000 bps.



2.5.1.2. RS232 (Revised Standard # 232, revision C)

Detta är en standard i USA utgiven av EIA (Electrical Industries Association), som beskriver både elektriska, mekaniska och logiska funktioner i en 25 polig kontakt. Det motsvarar för de

mekaniska beskrivningarna ISO 2110, för de elektriska V.28 och för de logiska V.24. I PC sammanhang används ofta denna beteckning på serieporten för att beskriva vad den står för, men eftersom det har visat sig att man inte använder alla 25 signalerna har även en 9-polig variant dykt upp som även följer RS232, bortsett från den fysiska kontakten.

2.5.1.3. Parallell

Egentligen hör inte denna kontakt hemma i datakommunikation eftersom datakommunikation är seriell. Men för att inte blanda ihop denna kontakt med RS232 så finns den med här. Den parallella porten är en 25-polig honkontakt på datorn och på en skrivare är det en speciell 36-polig kontakt som heter Centronics. (ett företag som har skapat en defactostandard). Kabeln som förbinder dessa två portar tappar 11 signaler, det är retursignalerna som sammankopplas så istället för 12 retur signaler finns det 8, och stift 15-18 plus 33-35 används inte, se tabellen.

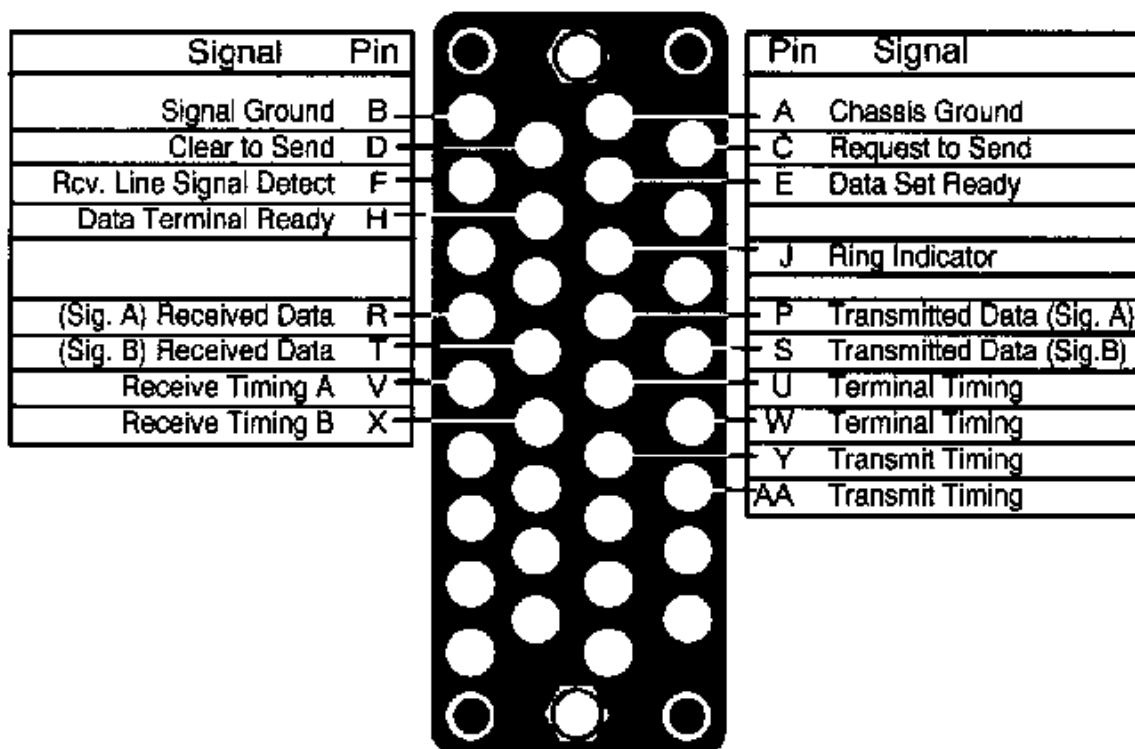
Centronics kontakt

Stift	Benämning	Stift	Benämning	25-pol	36-pol
1	Data strobe	19	Data strobe return	1-14	1-14
2	Data bit1	20	Data bit1 return	15	32
3	Data bit2	21	Data bit2 return	16	31
4	Data bit3	22	Data bit3 return	17	36
5	Data bit4	23	Data bit4 return	18-25	19-30,33
6	Data bit5	24	Data bit5 return		
7	Data bit6	25	Data bit6 return		
8	Data bit7	26	Data bit7 return		
9	Data bit8	27	Data bit8 return		
10	Acknowledge	28	Acknowledge return		
11	Busy	29	Busy return		
12	Paper end	30	Input prime return		
13	Select	31	Input prime		

14	Auto feed	32	Fault
15	Oscxt	33	Undefined
16	Logic ground	34	Undefined
17	Chassi ground	35	Undefined
18	5 volt	36	Select input

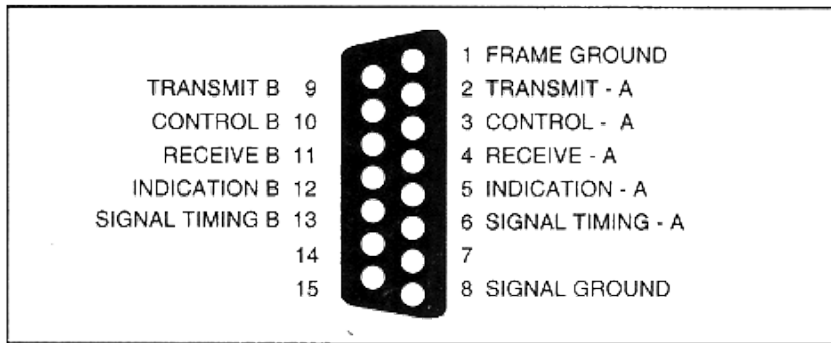
2.5.1.4. V.35

Detta är en 34-polig kontakt som används för överföringshastigheter på 64 kbps eller mer, och används i datel och digitel. De elektriska beskrivningarna anges i V.11.



2.5.1.5. X.21

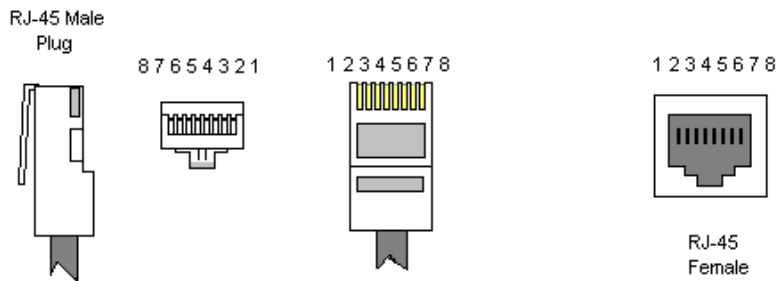
Denna 15-poliga kontakt används för anslutning till digitala nätverk som X.25 och DATEX. Den mekaniska beskrivningen är ISO 4903, den elektriska är X.26 och X.27, och för den logiska är X.24. Signalerna kombineras för att ge funktioner och status vid uppkoppling och under överföringen.



Functional characteristics of interchange circuits.

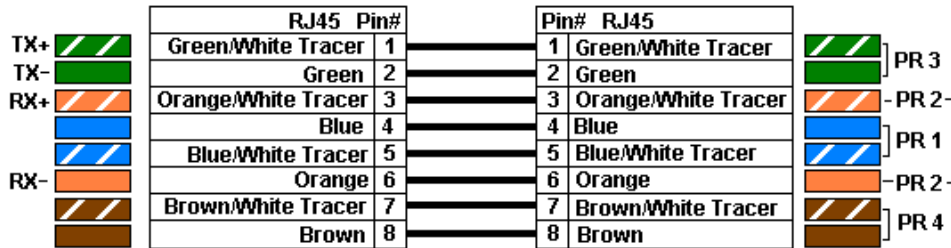
Interchange		Direction			Remarks
Circuits	DB15	Name	To DCE	From DCE	
G	1	Signal ground or common return.			See Note 1
Ga	8	DTE common return	X		
T	2 & 9	Transmit	X		
R	4 & 11	Receive		X	
C	3 & 10	Control	X		
I	5 & 12	Indication		X	
S	6 & 13	Signal element timing		X	See note 2
B		Byte timing		X	See note 3
X		DTE signal element timing	X		See note 4

2.5.1.6. RJ45



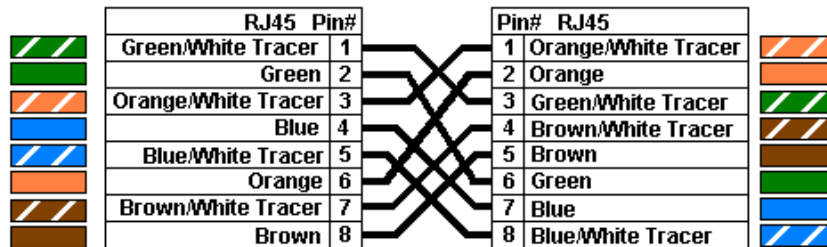
Color Standard
EIA/TIA T568A

Ethernet Patch Cable



Color Standard
EIA/TIA T568A

Ethernet Crossover Cable

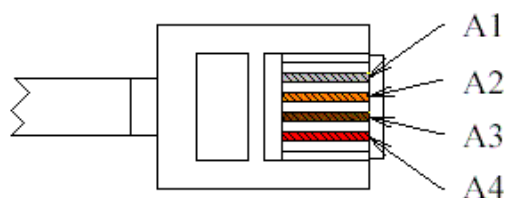


"A" is earlier

2006.06.28

Detta är en 8-polig kontakt som används till 10Base-T nätverk. Observera att det finns två par som överför signalerna, och det som är TD från den ena enheten blir RD på nästa, så om endast två enheter skall kopplas ihop kan man göra kabeln så att detta förhållande uppstår. Normalt är det huben som sköter denna koppling.

2.5.1.7. RJ11



PIN	Signal Name
A1	Ground
A2	Rx (Data Input)
A3	Tx (Data Output)
A4	Vcc (Power)

Detta är en 4-polig kontakt som används i telefoner, men kan även användas till datorer, är

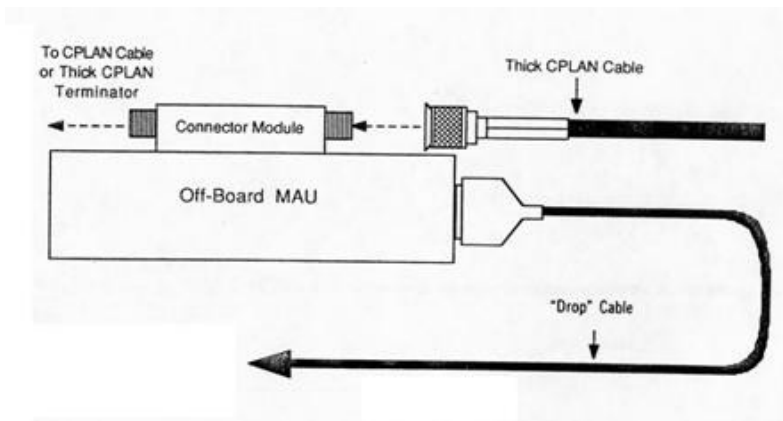
inte så vanlig där.

2.5.1.8. BNC

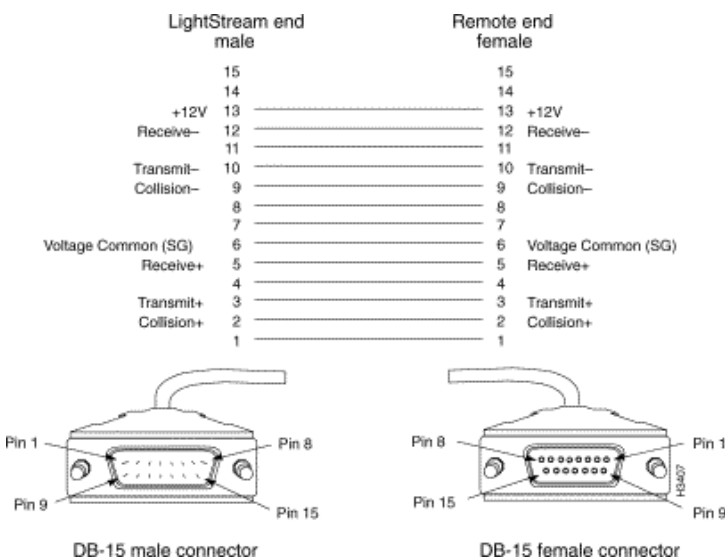


BNC kallas kontakten för koaxkabeln, och den är olika stor beroende på om man använder thin-black eller fat-yellow kabel till det lokala nätverket. När man använder fat yellow kabel kallas det N-typ. Det finns ett T-kors så att kabeln kan fortsätta från enheten samtidigt som signalen går in i enheten. Dessutom finns det en terminering som är ett motstånd som man satt i en BNC kontakt som man sätter på T-korset.

2.5.1.9. AUI



Denna kontakt används när det inte finns någon förstärkare på nätverkskortet utan den finns fristående. Det är en 15 pinnars kontakt som förmedlar även spänning till den externa förstärkaren. Vid 10Base5 nätverk används denna kontakt för att koppla in kabeln.



2.5.2. Elektriska nivåer

De elektriska nivåerna är beskrivna i olika standarddokument beroende på vilken typ av gränssnitt som används.

De elektriska nivåerna i RS232 följer två principer, en för kontrollsignaler och en för datasignaler, och är beskrivna i V.28. Kontrollsignalerna är de som reglerar proceduren mellan modemmet och den anslutande kommunikationsenheten, och datasignalerna är de som överför data. Kontrollsignalerna är aktiva eller passiva. Med en aktiv kontrollsignal menas att den funktion som signalen har är i funktion, till exempel Request To Send (105) aktiv innebär att datorn vill sända information över datasignalerna, är den passiv så vill datorn inte sända något. En datasignal är en etta eller nolla som överförs. En aktiv kontrollsignal är på nivån ? +6 volt, och ej aktiv på nivån ? -6 volt. En etta på datasignalen är ? -6 volt, kallas även att linjen är marking, och en nolla är ? +6 volt, kallas att linjen är spacing. Med andra ord är dessa två signaltyper inverterade. När ingen överföring sker så är normalt DTR och DSR aktiva, det vill säga ? +6 volt, alla de andra kontrollsignalerna är passiva, det vill säga ? -6 volt, och TD och RD är en etta, det vill säga ? -6 volt. Spänningar mellan + 3 volt och - 3 volt är odefinierade, det vill säga är varken 1 eller 0, passiv eller aktiv. 6 volt är den normala nivån på V.28, men standarden tillåter +/- 12 volt. Med dessa spänningsnivåer och den impedans som kablarna har så blir den effektiva kabellängden 80 - 100 m mellan kommunicerande enheter.

Det finns andra standarder när det gäller elektriska nivåer, som RS422 och RS423. RS422 är ett balanserat gränssnitt där det krävs två signaler, A och B, för att överföra data. Det är skillnaden i volt mellan dessa signaler som definierar 1 eller 0, 1 = A positiv i förhållande till B, 0 = A negativ i förhållande till B. Spänningsskillnaden är 2 till 6 volt mellan A och B. Fördelen med detta arrangemang är att det blir okänsligare för störningar. En störning påverkar båda signaler lika och på samma sätt, och eftersom det är skillnaden mellan dem som räknas blir effekten av störningen minimal. RS423 är ett obalanserat gränssnitt som kräver endast en signal för att överföra data. Spänningen är 0 till 6 volt, men den elektriska karakteristiken är förbättrad jämfört med RS232 så längre kablar kan användas och med högra överföringshastigheter.

2.5.3. Signaler för RS232

I gränssnittet beskrivs vilka signaler som finns på respektive stift och vad de har för funktion. Tabellen nedan visar signalerna för V.24 och RS232 i den 25-poliga kontakten.

Ursprung	Stift nr	V.24	Benämning	förkortning	RS232	
DTE/DCE	1	101	Circuit Ground	CG	AA	
DTE	2	103	Transmit Data	TD	BA	
DCE	3	104	Receive Data	RD	BB	
DTE	4	105	Request To Send	RTS	CA	
DCE	5	106	Clear To Send	CTS	CB	
DCE	6	107	Data Set Ready	DSR	CC	

DTE/DCE	7	102	Signal Ground	SG	AB
DCE	8	109	Data Carrier Detect	DCD	CF
	9		Not used		
	10		Not used		
DTE	11	126	Select Transmit Frequency	STF	CH
DCE	12	122	Secondary channel Data Carrier Detect	SDCD	SCF
DCE	13	121	Secondary channel Clear To Send	SCTS	SCB
DTE	14	118	Secondary channel Transmit Data	STD	SBA
DCE	15	114	Transmit Clock	TC	DB
DCE	16	119	Secondary channel Receive Data	SRD	SBB
DCE	17	115	Receive Clock	RC	DD
	18	141	Loop back		
DTE	19	120	Secondary channel Request To Send	SRTS	SCA
DTE	20	108/2	Data Terminal Ready	DTR	CD
DCE	21	110	Signal Quality Detector	SQD	CG
DCE	22	125	Ring Indicator	RI	CE
111=DTE 112=DCE	23	111/112	Data Signalling Rate Selector	DSRS	CH
DTE/Extern	24	113	Transmit Clock	TC	DA
	25	142	Loopback indicator		

TD Transmit Data är den signal som sändande dator skickar sin data som

skall till mottagaren. För att data skall kunna sändas krävs att följande signaler är aktiva: RTS, CTS, DTR och DSR.

- RD Receive Data är den signal som mottagande dator får data från den sändande datorn. För att data skall kunna mottagas krävs att följande signaler är aktiva: DTR, DSR och DCD.
- RTS Request To Send är den signal som sändande dator gör aktiv när en överföring av data skall ske, och data kan inte börja sändas förrän CTS signalen har mottagits. DTR och DSR måste vara aktiva.
- CTS Clear To Send är den signal som ger klartecken att börja sända data på TD signalen, den är ett svar på RTS. DTR och DSR måste vara aktiva.
- DSR Data Set Ready är signalen från modemmet att det är klart för kommunikation. DSR blir aktiv normalt när strömmen är påslagen och självtesten är klar. Den har inget att göra med om en förbindelse finns eller inte. När DSR är aktiv betyder detta också att de signaler som modemmet kontrollerar har den definierade funktionen, datorn kan använda dem för att styra kommunikationen mellan dator och modem.
- DCD Data Carrier Detect är den signal som när den aktiv betyder att bärivågen som mottagas är bra och att den data som är modellerad på denna kan tolkas som aktuell data. Om det är en dålig förbindelse, eller om förbindelsen bryts så blir denna signal passiv, och indikerar att det blivit ett fel på överföringen.
- TC Transmit Clock är den signal som genereras från modemmet till datorn och bestämmer bittakten som datorn skall lämna data till modemmet vid synkron överföring, används inte vid asynkron överföring.
- RC Reciev Clock är den signal som genereras från modemmet till datorn och bestämmer bittakten som datorn skall ta emot data från modemmet vid synkron överföring, används inte vid asynkron överföring.
- DTR Data Terminal Ready är den signal som datorn gör aktiv när den är klar för att kommunicera med modemmet. DTR blir aktiv normalt när strömmen är påslagen och operativsystemet har initierat datakommunikationsporten. När DTR är aktiv betyder detta också att de signaler som datorn kontrollerar har den definierade funktionen, modemmet kan använda dem för att styra kommunikationen mellan dator och modem.
- RI Ring Indicator är den signal som talar om att en förbindelse kopplas upp från någon utifrån.

De flesta av stiften i den 25-poliga kontakten används inte idag och därför har många leverantörer övergått från den 25-poliga till 9-poliga. Observera att det gäller endast för asynkron överföring. Observera också att stift 2 är RD och inte TD som i den 25-poliga!

Stift nr	Förkortning
1	DCD

2	RD
3	TD
4	DTR
5	SG
6	DSR
7	RTS
8	CTS
9	RI

2.5.4. Seriell

Seriell kommunikation innebär att det finns endast en signal som informationen skall överföras på, vilket betyder att bitarna måste komma efter varandra i en lång rad. Inuti datorn är informationen parallell, vilket innebär att det måste ske en omvandling från parallell till seriell någonstans på vägen. Det som kallas den seriella porten i en PC är en speciell krets, UART (Universal Asynchronous Receiver/Transmitter) som har till uppgift att dels omvandla parallellt till seriellt, och tvärtom, samt att styra de signaler som finns till gränssnittet till modemmet. Det finns olika typer av UART som har olika kapacitet av överföringshastighet till modemmet. Den första UART, 8250, har en kapacitet på max 19200 bps, och den har ett buffer för ett tecken. En utveckling av denna, 16450 har en kapacitet på max 115 200 bps, men fortfarande endast ett tecken i buffer. Dessa båda kräver mycket av processorn som måste tömma detta buffer ofta för att inte tappa någon information. 16550 UART har samma kapacitet som 16450 men har ett 16 tecken buffer, vilket avlastar processorn.

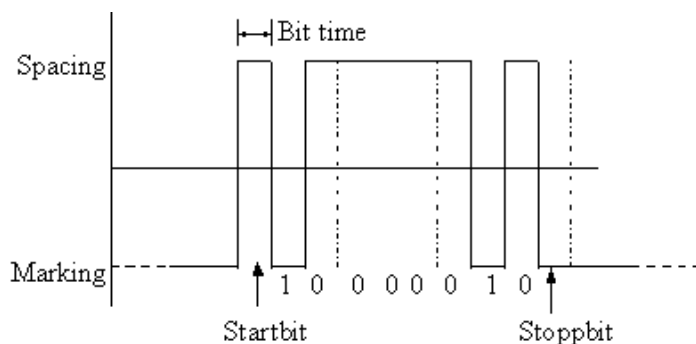
Standardmässigt finns det inte någon synkron port på en PC, men det finns att komplettera med, och då används en motsvarande krets USRT (Universal Synchronous Receiver/Transmitter), och som har samma funktioner som UART plus TC och RC signalerna. Kapaciteten på USRT är som 16550 och överföringshastigheten styrs av TC och RC.

När omvandlingen från parallell till seriell sker har det betydelse i vilken ordning som bitarna kommer, om det är den mest signifikanta biten som kommer först eller den minst signifikanta. Den mest signifikanta biten är den som har det högsta binära värdet, i teckentabellerna framgår detta. Bokstaven A har den binära koden 01000001, där den mest signifikanta biten står först., angivet i hex blir det 41. När detta tecken, A, skall omvandlas till seriellt så börjas först med den minst signifikanta biten, det vill säga 1. Med andra ord vi tar bitarna bakifrån, så det som kommer på den seriella signalen är först 1, sedan 0, 0, 0, 0, 0, 1 och sedan 0. Om paritetsbit skall användas kommer den sist, så att ordningen blir då, först 1 sedan 0, 0, 0, 0, 0, 1, 0 och sist paritetsbiten.

2.5.4.1. Asynkron överföring

Asynkron överföringsteknik innebär att varje tecken överförs utan relation till något annat än till sig själv. För att kunna identifiera start och slut av tecknet och för att kunna säkerställa identifieringen, används startbitar och stoppbitar som adderas till tecknets egna bitar. En startbit adderas till i början, och en eller två, eller en och en halv stoppbit adderas till i slutet. Detta är den första överföringsteknik som introducerades, och den tillåter att tiden mellan varje

tecken är variabel, det vill säga den kan vara allt mellan 0 och ?. Antalet stoppbitar bestäms av den utrustning som används, och är oftast en parameter som skall väljas vid initieringen av kommunikationen. Bittakten bestäms av den hastighet som skall användas vid överföringen och är också en parameter som skall väljas vid initieringen. Bittakten genereras av respektive sändare och mottagare själva. Nackdelen med denna teknik är att överföringen belastas med extra bitar som inte har med informationen att göra, det blir en overhead. Moderna utrustningar som använder asynkron teknik och åtta bitars teckenkod, har en startbit och en stoppbit, vilket ger en overhead på 20 %. Paritetsbitar kan ingå, och adderas då till teckenkoden. Man säger att asynkron teknik är teckenorienterad, där ett tecken normalt idag är 7 eller 8 bitar.



Asynkron teknik används idag när många skall kommunicera med en central, till exempel en BBS (Bulletin Board System), eftersom det är en billig och beprövad teknik, alla PC har asynkron port inbyggd.

2.5.4.2. Synkron överföring

Synkron överföringsteknik är att mottagare och sändare är i synkronisation med varandra. Det finns teckenorienterad och bitorienterad synkron överföringsteknik. I en teckenorienterad teknik sker synkroniseringen genom att först sända synkroniseringstecken på datasignalen. Dessa tecken är SYN (16, hex), i moderna utrustningar 2 stycken, och måste alltid komma först, sedan följer de tecken som skall representera data i en följd. I synkron överföringsteknik måste ett tecken följa på nästa utan något uppehåll, det blir en ström av tecken från första till sista, det blir ett block med tecken. Därför kallas synkron teknik för blockorienterad överföring. Om blocket är mycket långt måste SYN tecken läggas in inuti blocket för att bibehålla synkronisationen, vad som är långt beror på utrustningen och kvaliteten på förbindelsen. Synkron överföringsteknik har ingen overhead på teckennivå men har det i form av de extra synkroniseringstecknen. Denna overhead är försumbar jämförd med asynkron överföring. Säg att ett block på 1920 tecken (en full bildskärm, 80 tecken * 24 rader) skall sändas iväg och 2 SYN-tecken finns med i början, det blir en overhead på 0.1%. Denna beskrivning är för teckenbaserade protokoll, för bitorienterade protokoll är SYN-tecknet ersatt med en flagga, men det är fortfarande ett block som sänds. Observera att på den fysiska nivån har bitarna ingen betydelse och därför kan inget tecken tolkas på denna nivå, utan det är först på nästa nivå som detta kan ske.

2.5.5. Parallell

Parallell kommunikation innebär att flera bitar per tidsenhet kan överföras, och i datakommunikation sammanhang är det 8 bitar som samtidigt överföres. Jämfört med seriell överföring är den teoretiskt 8 gånger snabbare, men i praktiken är inte skillnaden så stor. På en PC finns som standard en parallell port som används till att sammankoppla en skrivare, bandstation, eller någon annan periferi som kan hantera ett parallellt gränssnitt. Det finns numera även modem som kopplas till parallellporten. Vid parallell överföring så finns det en datastrobe som styr när datasignalerna skall läsas av. och denna är en kort puls som anländer

i mitten på datasignalernas puls. Det finns några styrsignaler som reglerar statusen på de kommunicerande enheterna och dess handskakningsprocedur.

2.5.6. Modem

Modem är en förkortning och sammanslagning av två ord, MOdulator, som betyder att man lägger datasignalen till en bärvåg som moduleras av datasignalen, och DEModulator, som betyder att man tar bort bärvågen från datasignalen. Modem används när avstånden är stora och när man skall använda telefonnätet för överföringen. Telefonnätet är ett analogt nätverk, det överför signalerna med sinusvågor, till skillnad mot datorn som använder digitala signaler av ettor och nollor. Det krävs en omvandling av signalerna för att de skall kunna användas av respektive enhet. Denna omvandling står modemmet för, det tar emot digitala signaler från datorn och lämnar en analog signal till telefonnätet. På mottagarsidan måste också ett modem finnas som tar emot den analoga signalen, omvandlar den till en digital signal som lämnas till datorn.

2.5.6.1. Kopplingsprocedur

När dator och modem skall överföra information så måste en procedur följas så att saker och ting sker i rätt ordning. Denna procedur försäkrar att ordningen upprätthålls och klargör vem som skall göra vad vid bestämda tidpunkter.

En beskrivning av en half duplex, tvåtråds uppkoppling följer. När strömmen till modemmet slås på och när det är klart att ta emot uppdrag så blir signalen Data Set Ready (107) aktiv från modemmet till datorn. Denna signal aktiv innebär att de signaler som modemmet kontrollerar följer den definierade funktionen och nivåerna, när denna signal är inaktiv kan de signaler som modemmet kontrollerar ej garanteras att de befinner sig på rätt nivå, de är inte tillförlitliga, m.a.o. datorn skall inte "avlyssna" dessa när DSR är inaktiv. På samma sätt är det för signalen Data Terminal Ready (108/2), som kommer från datorn till modemmet. När datorn är i ett läge att den kan garantera nivå och funktion på de signaler som den kontrollerar så är DTR aktiv. Detta sker normalt när strömmen är på och kommunikationskanalen är initierad av operativsystemet, det vill säga UARTen är initierad som är kopplad till den seriella porten.

Nu vill datorn sända information via den seriella porten och gör därför signalen RTS (Request To Send, 105) aktiv. Modemet kommer nu att bygga upp sin bärvåg och när modemmet är klar för att ta emot datasignalen så returneras signalen CTS (Clear To Send, 106) till datorn, som då kan börja skicka information på TD (Transmit Data, 103). På mottagarsidans modem så upptäcker detta, genom att en bärvåg finns, och ger information om detta till datorn på mottagarsidan att bärvåg finns, DCD (Data Carrier Detect, 109). DCD aktiv innebär att data som finns på RD (Receive Data, 104) signalen är tillförlitlig. det vill säga den är kontrollerad av modemmet. Om DCD går inaktiv under mottagningen så betyder detta att signalerna på RD inte är giltig data. Den sändande datorn skickar nu sin information på TD signalen, som i sin tur mottagas av datorn på mottagarsidan på RD signalen, och detta håller på så länge sändande dator håller RTS signalen aktiv. När den sista data är överförd så gör sändande dator RTS passiv och signalerar till modemmet att den är klar med sändningen. Modemet tar ner bärvågen och returnerar CTS passiv till datorn. På mottagarsidan upptäcks att bärvågen försvinner och modemmet returnerar DCD passiv till sin dator, som då vet att överföringen är klar. Denna procedur kallas även en handskakningsprocedur.

Vad händer om det blir fel någonstans? De fel som kan uppstå här är att telefonlinjen utsätts för störningar så att bärvågen blir för svag eller försvinner helt, en längre eller kortare tid, eller att förbindelsen bryts. I det första fallet när bärvågen försvinner eller är för svag så upptäcker mottagardatorn detta genom att DCD blir passiv, och detta innebär att kommunikationen måste göras om. Sändande dator vet inget om detta förhållande utan den fortsätter tills den är färdig. Om förbindelsen bryts upptäcks detta av båda modemmen som informerar sina

respektive datorer detta. Sändande dator får information genom att CTS signalen går passiv, mottagare datorn får information via DCD. Lagg märke till att inget nämns om att datasignalen är korrekt eller inte. Denna kontroll sker på annat sätt än via gränssnittet, och som skall redogöras senare i avsnittet säkerhet på länknivån. Kom ihåg att bitarna har ingen mening i den fysiska nivån.

För att ta emot data efter att datorn har sänt iväg sina data så måste modemmet ställa om till mottagning, vilket innebär att bärvågen tages ner. Sedan när datorn skall sända igen så måste bärvågen byggas upp igen och detta tar en liten stund, ca 200 ms (millisekunder), och därför har många half duplex förbindelser fyrtråd så kan bärvågen kan vara på hela tiden på det par av trådar som används för sändning.

I full duplex så är RTS alltid aktiv och därmed CTS eftersom det finns en kanal för sändning och en för mottagning, alltså kan alltid sända och bärvågen är på hela tiden.

Denna beskrivning förutsätter att det finns en uppkopplad förbindelse mellan modemmet. Att koppla upp förbindelsen kan ske manuellt eller som oftast idag automatiskt av modemmet.

2.5.6.2. Hayes-kommandon

Modem skall hantera många olika förutsättningar, och klarar av att ringa upp själv det nummer som presenteras för modemmet. För att förenkla tillverkningen av modem så användes parametrar för att ställa in modemmet till önskad funktionalitet, och detta gjordes i början av byglar på modemets kretskort. Detta var en besvärlig procedur om man ville ändra på modemets funktion/konfiguration, så en man vid namn Dennis Hayes designade ett modem med register och skapade ett kommandospråk för att sätta upp registren, och dessa register ersätter alla byglar på kretskorten. Det blev nu möjligt att mjukvarumässigt ställa om modemmet, och han lyckades så bra med att sälja sina modem att det blev en defactostandard. Hayeskommandon sändes till modemmet som känner igen dessa, alla börjar med at och sedan följt av kommandot eller flera kommandon. De parametrar som man kan styra med är till exempel att man anger ett nummer som modemmet skall ringa upp, man anger hur många ringsignaler som skall mottagas innan modemmet svarar, om det skall vara tonval eller pulskod. En normal användare kommer inte i kontakt med Hayeskommandon utan drivrutiner eller kommunikationsprogrammen handhar detta. Endast genom en konfiguration där man anger en del parametrar brukar finnas, till exempel telefonnummer som skall ringas upp, tonval eller pulskod och så vidare Men som system- eller nätverksadministratör kan alla Hayeskommandon användas för att felsöka och korrigera eventuella fel.

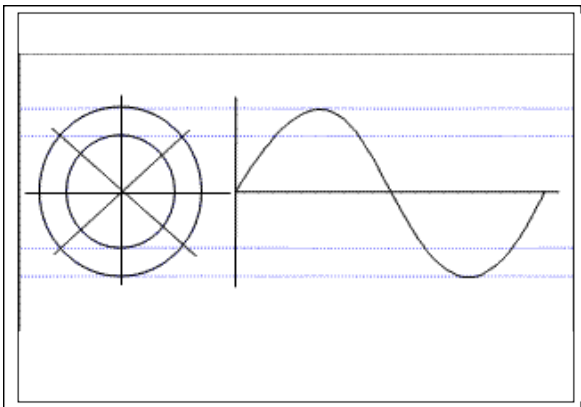
Exempel på Hayeskommandon är: ATDT0W020520521, betyder att ring upp nummer 020 520 521 via växeln där man slår 0 för att komma ut på linjen. Naturligtvis finns det kommandon som läser av registret och sänder detta tillbaks, vilket gör att du som operatör kan kontrollera att modemmet är rätt inställt för sin uppgift, ex. ATSp?, visa innehållet i register x. ATi4 visar innehållet i alla register, och är ett bra kommando vid felsökning av modemkommunikation.

AT	inleder alla kommando
D	Dial, slå nummer
T	Slå nummer med Tonval

W	Wait, vänta på kopplingston
S0-15	Systemregister

2.5.6.3. Överföringssätt

En signal (växelström) definieras med tre egenskaper, amplitud, frekvens och fas. Amplituden är hur högt den når från nollläget, på bilden är nollläget i origo i cirkeln respektive linjen i x-led, och amplituden är angiven av de streckade linjerna. Frekvensen anger hur många perioder den genomgår per sekund, på bilden är en period ritad. Fasen anges i grader och anger läget vid varje tidsenhet, på bilden är 0, 45, 90, 135, 180, 225, 270 och 315 angivna. Bärsvågen är en signal som är bestämd i amplitud, frekvens och fas, det är en sinusvåg som på bilden.



Moduleringen av bärsvågen kan ske på olika sätt, dels via amplituden, dels via frekvensen och dels via fasan. Hur många gånger per sekund som man modulerar bärsvågen kallas för modulationshastigheten, och den anges i Baud. Modulationshastigheten blandas ofta ihop med överföringshastigheten genom att likställa dessa med varandra, vilket som vi skall se, är felaktigt. Grunden till detta är att i starten av modemtekniken, så var dessa lika, men efter hand som nya tekniker för modulering kommit, är det idag så att moduleringshastigheten är betydligt lägre än överföringshastigheten.

När man amplitudmodulerar bärsvågen adderas datasignalen till bärsvågen och därmed kommer amplituden på bärsvågen att variera med datasignalen mellan två definierade amplitudnivåer, som då representerar 0 respektive 1. Vid frekvensmodulering adderas datasignalen till bärsvågen som därmed kommer att variera i frekvens med datasignalen mellan två definierade frekvenser, som representerar 0 respektive 1. Fasmodulering görs med fassprång, där man definierar att en etta har en viss fas till exempel 90° och noll 270°. Det går att kombinera dessa olika moduleringsätt, och i QAM (Quadrature Amplitude Modulation), är bärsvågen förändrad med datasignalen genom både fassprång och amplitudändring för att representera en bitgrupp vid varje modulering. Till exempel en grupp om fyra bitar klaras med 16 olika tillstånd på linjen, med två amplitudnivåer och 8 fassprång. Om överföringshastigheten är 28 800 bps så blir modulationshastigheten med QAM 1800 Baud. Till exempel 0000 = fas = 0° + låg amplitud, 0001 = fas 45° + låg amplitud, 1000 = fas 0° + hög amplitud, 1110 = fas 315° + hög amplitud.

En ny standard har kommit som är benämnd V.90 och innebär att överföringshastigheten är upp till 56 kbps. Denna standard använder digital teknik om telefonanslutning är gjord sådan, annars analog teknik som i vanliga modem. De flesta abonnenter är anslutna till sin telefonoperatör med analog teknik, men till exempel Internetoperatörer som erbjuder

uppkoppling till Internet har digital anslutning. Hastigheter upp till 56 kbps kan uppnås, och det är när abonnenten är ansluten digitalt till telefonnätet. Det normala för en person som skall ansluta sig till Internet via telefonnätet, är att den har en analog anslutning och använder ett V.90 modem. Internetoperatören har en digital anslutning och detta medför att V.90 modem hos personen använder 33.6 kbps för att överföra till operatören, medan denna kan sända med 56 kbps. I Internetsammanhang där man i regel mottager mycket mer data än man sänder är detta en tillfredsställande situation.

2.5.6.4. Korthållsmodem

Korthållsmodem används för att förlänga avståndet lokalt inom ett privat område, och dessa använder basband och någon form av linjekod. Linjekod är för att minska likströmskomponenter och att öka synkroniseringsinformationen, exempel på metoder är ADI (Alternate Digit Inversion) ? varannan bit inverteras, AMI (Alternate Mark Inversion) ? varannan etta polvändes, Manchesterkodning ? polariteten kastas om vid halva bittiden.

2.5.6.5. Felkorrigering och komprimering

Modemen har utvecklats i konkurrens mellan modemtillverkarna, och idag finns det modem som själva kontrollerar att det som de skickar mellan varandra är korrekt. Modemen har ett buffer och använder paritetskontroll på hela meddelandet, och blir det fel omsändes det. Paritetskontrollen sker med en checksiffra enligt en speciell algoritm som kan upptäcka fel och i vissa fall även kan korrigera fel. Felkorrigering finns med en defactostandard MNP-4 (lanserat av modemtillverkaren Microcom), och den officiella heter V.42. För att kontrollera att all data överföres korrekt så blir denna felkorrigering belastad med lite overhead, paketnumrering, checksiffror, vilket medfört att datakompression också tillkommit mellan modemen.

Datakomprimering har också en defactostandard MNP-5, som även inkluderar MNP-4, och den officiella standarden heter V.42bis. Defactostandarden och den officiella är inte 100 % samma och därför är det viktigt att veta vilken standard som de kommunicerande modemen använder, det måste ju vara lika standard i båda ändarna av kommunikationen. Det finns modem som inkluderar alla standarderna och där dessa modem först "förhandlar" med varandra för att klargöra vad som skall gälla för den uppkopplade förbindelsen.

Datakomprimering kan göras p.g.a. att det finns mycket "luft" i den data som överföres. Ett ordbehandlingsdokument har mycket blanktecken till exempel och finns det låt säga 20 st. blanktecken efter varandra, så kan dessa ersättas med en speciell kod som säger att just 20 st. blanktecken skall ersättas av denna kod. Koden är på kanske 5 tecken, så 15 tecken sparas. Det finns speciella matematiska algoritmer som används för att komprimera data, och hur mycket data kan komprimeras beror på datan själv. En exekverbar fil som skall överföras kan oftast inte komprimeras lika mycket som ett ordbehandlingsdokument. Bilder som redan är komprimerade med speciella algoritmer, till exempel JPEG för bildbehandling kan oftast inte komprimeras mera än de redan är.

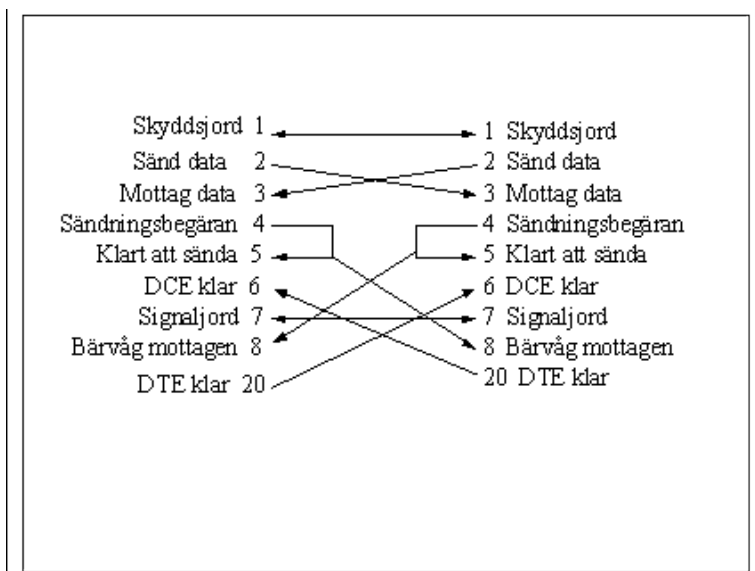
2.5.6.6. Nollmodem

Vid vissa tillfällen är det inte lämpligt att använda ett modem när två enheter skall kommunicera, men man vill använda samma procedur som när ett modem finns till hands. Man vill simulera att det finns ett modem som respektive enhet är ansluten till. Då använder man en speciell kabel som man kallar nollmodem, och denna kabel är konstruerad så att signalerna är "korskopplade". Ena änden kallar vi A och den andra B, då är TD-signalen i A kopplad till RD-signalen i B, och TD i B till RD i A. Dessutom måste en del andra signaler korskopplas eller kopplas speciellt så att båda ändar tror att det finns ett modem där. Ett

exempel på en nollmodemkabel är en seriell skrivarkabel. Skrivaren är kopplad till RS232 porten och denna är ju konstruerad för att kommunicera med ett modem, liksom skrivarens port. I princip skulle man kunna ställa skrivaren långt från PC'n och koppla dem samman med ett modem, men det är inte så praktiskt och därför mycket ovanligt.

Vissa program som använder en noll-modemkabel följer inte den procedur som beskrivits här, utan har definierat sina egna procedurer. Till exempel man endast korskopplar datasignalerna och använder mjukvarustyrd flödeskontroll. Eller man använder RTS och CTS korskopplade som flödesstyrning, RTS aktiv innebär att denna enhet är klar för att ta emot data, passiv medför att den ej kan ta emot. Därför kan den nedan beskrivna noll-modemkabeln inte fungera i alla situationer och dokumentationen för programmet får konsulteras för hur den skall se ut för just detta program.

Nollmodem kan man använda när man skall felsöka två enheter som har problem med kommunikationen, de måste förstås då finnas i samma rum och man kan då se direkt på respektive enhet vad som händer. Alternativet är att två personer måste engageras, en på varje ställe, samtidigt som man då också har ytterligare två felkällor, modemerna och televerkets linjer.

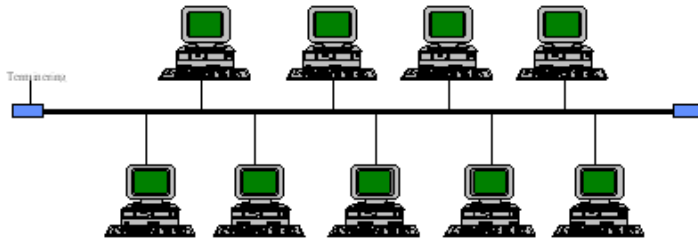


En seriell skrivarkabel är en noll-modemkabel, där det oftast är mjukvarustyrd flödeskontroll och endast pinne 2 och 3 har betydelse, på datorsidan är oftast pinne 20 sammankopplad med 5 & 6, så endast pinne 2,3, 5, 6 och 20 på datorsidan används och endast 2 och 3 går över till skrivaren. **Obs! riktningen på signalerna, det går inte att koppla ihop två utgående eller två ingående signaler.**

2.6. Topologi

Topologi är en beskrivning på nätverkets fysiska utbredning i rummet. Men ibland refereras även till den logiska utbredningen.

2.6.1. Bus

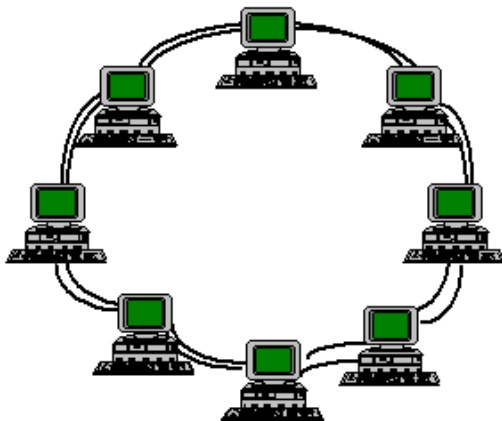


Bussnätverk är ett nät där alla noder är kopplade till nätverket, som en motorväg, det vill säga att nätverket är en lång kabel och på denna finns tillfarter som noderna kan ansluta sig till. I ett bussnätverk "ser" alla noderna all kommunikation på kabeln, men reagerar endast på det paket som har den egna adressen påsatt. Fysiskt kan bussen gå från enhet till enhet men kan också vara en enda lång kabel som enheterna ansluter sig till via ett vampyruttag. När man ansluter enheterna genom att kabeln först går till den ena sedan till den andra och så vidare blir det mycket störkänsligt. Om någon enhet kopplar ur sig så bryts bussen och ingen kan kommunicera. Termineringen i båda ändar är mycket viktig.

Bussen kan fysiskt också vara ett stjärnnätverk som när partvinnad Ethernet används. Med partvinnad Ethernet används en hub som är den egentliga bussen, och denna är konstruerad så att man kan koppla in och ur enheter utan att bussen störs.

Bussnätverk kan användas med en token så att man får en logisk ordning på enheterna och därmed vilken ordning som enheterna kan sända. Varje enhet måste då känna till vilka enheter som logiskt är före respektive efter sig själv och går till så att en enhet börjar med att sända ut en token till nästa enhet. Denna tar emot token och om enheten vill sända något så markeras token upptagen och meddelandet adderas till, skickar sedan detta paket eller token vidare till nästa enhet. Om ett paket läggs till, och när detta kommer fram till den mottagande enheten så tar denne hand om paketet, kontrollerar att det är korrekt, sätter en flagga på token och låter det gå vidare. När så paketet återkommer till den sändande enheten avläses token hur det mottogs, och om det är OK så plockas paketet bort från token och denna flaggas som ledig, och skickas till nästa enhet annars skickas paketet ut en gång till. Den ordning som token går har inte med den fysiska placeringen av enheterna att göra, utan är en helt igenom logisk funktion.

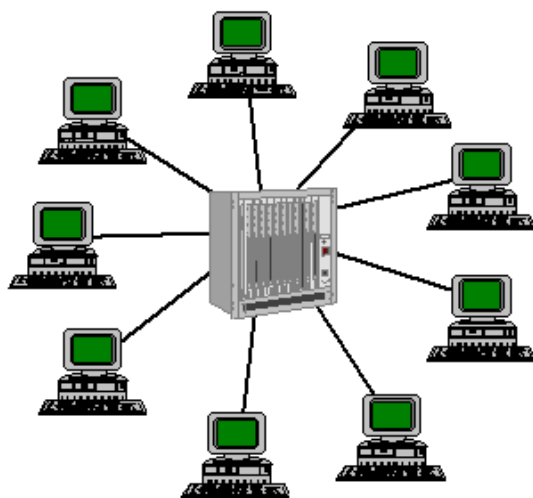
2.6.2. Ring



Ringnätverk är de nätverk där en signal återkommer till den sändande enheten efter att gått ett varv. Det mest vanliga ringnätverket är token ring, som lanserats av IBM och det är nästan endast i IBM-nätverk som token ring används. Det finns en token som vandrar i nätverket och som när en enhet mottager denna token kan sända om den är ledig. Genom att denna token snurrar runt i nätverket och återkommer till "ursprunget" så är man garanterad en viss access på nätverket. Det fungerar som ett token bussnätverk logiskt men fysiskt är det en sändarkanal och en mottagarkanal och det är fysiskt ett stjärnnätverk av praktiska skäl. Ett ringnätverk är känsligt för störningar, kopplas en enhet bort eller ej fungerar, så bryts ringen och ingen kan kommunicera. Genom att koppla det som ett fysiskt stjärnnätverk med en enhet i mitten, en MAU (Multistation Access Unit) som kan koppla förbi enheterna så kan kommunikationen bibehållas. I ett Token ringnätverk kan max 96 noder finnas med max 12 MAU, och där avstånd mellan första och sista MAU är max 400 fot. Max avstånd MAU och noden är 150 fot.

FDDI är ett annat exempel på ringnätverk som är ett ringnätverk både fysiskt och logiskt.

2.6.3. Stjärna



Ett stjärnnätverk är ett nätverk där det finns en enhet i mitten som styr. Terminalnätverk är ofta stjärnnätverk., men partvinnad Ethernet och token ring är också en typ av stjärnnätverk. I partvinnad Ethernet är enheten i mitten bussen, och kallas hub, och kablarna är anslutningarna till bussen. I token ring är enheten i mitten en MAU som underlättar den fysiska

kopplingen och möjliggör förbikoppling.

2.6.4. Kretskoppling

Kretskopplade nätverk är nätverk som man etablerar en väg genom nätverket och har denna uppkopplad under överföringen. Kretskopplade nätverk är normalt det allmänna telefonnätet, men det finns även rena datanätverk som är kretskopplade. I ett kretskopplat nätverk så är man inte ansluten till en speciell motpart, utan man kan koppla upp sig till olika motparter genom att ringa olika telefonnummer. Kretskopplade nätverk kan vara circuit switch, message switch eller packet switch.

2.6.4.1. Datel

Datel är den tjänst som Telia erbjuder när det gäller det allmänna telefonnätet och där man ansluter sig via modem. Den finns i två varianter, dels datel uppringd dels datel fast. Med datel uppringd etableras kontakt med motparten vid varje kommunikationstillfälle och däremellan finns ingen kontakt mellan parterna, medan med datel fast är en hyrd linje mellan parterna och denna finns alltid uppkopplad. Datel fast finns i två varianter, datel anatel och datel digitel, och skillnaden finns i det som namnen antyder och i kapacitet. Anatel finns i upp till 19200 bps och digitel finns i tre hastigheter, 64 kbps, 2 Mbps och 34 Mbps. Anatel kan även ha droppunkter, upp till 8 stycken.

Det finns tilläggstjänster som för anatel är felkorrigering, komprimering, automatuppringning, och motuppringning, och för digitel automatisk uppringd reservväg, digital linjedelare, multidrop, multiplexor.

Observera att dessa hastigheter inte är komprimerade, så om man använder modem som kan komprimera mellan modemen blir överföringshastigheten högre gentemot terminalen, men på Telias nätverk är de angivna bps taket för linjen.

2.6.4.2. Datex

Detta är ett speciellt digitalt datanätverk som de nordiska länderna har byggt upp, och till skillnad från datel är man endast uppkopplad under överföring av data, alltså inte uppkopplad under hela kommunikationen. Detta medför att motparten måste kopplas upp varje gång data skall överföras, men uppkopplingstiderna är mycket snabba ? 0,5 sekunder, normalt 0,2 s. I datex används inte modem utan en enhet som kallas DCE-X, och som kräver att terminalen hanterar upp/nerkoppling enligt X.21. Om terminalen inte kan hantera X.21-upp/nerkoppling så finns det DCE-V för synkron V.24, DCE-Vpc för asynkron V.24 och DCE-Vp/st. för Hayeskompatibel V.24.

2.7. Flödeskontroll

De kommunicerande enheterna måste kunna styra överföringen så att ingen information tappas. Det går ju inte att den sändande parten skickar data i en ström som den mottagande parten ej hinner bearbeta. På nivå 1 sker detta på flera sätt beroende på vilket trafiksätt som används, simplex, halv eller full duplex. I WAN om hårdvarustyrd flödeskontroll används så sker denna på nivå 1 genom att använda signalerna i gränssnittet som DSR & CTS. DSR betyder ju att data set är klar för operation, om DSR blir passiv så innebär detta att den andra parten ej kan sända något enligt den beskrivning som tidigare gjorts. På samma sätt, men mindre drastiskt eftersom det följer ett logiskt mönster, så om CTS göres passiv innebär detta att den sändande enheten inte har klartecken för att sända mera data. Om DSR blir passiv medför detta att kommunikationen i verkligheten har avbrutits, medan CTS är en del av just

flödesstyrningen. I LAN sker flödeskontrollen genom att definiera olika paketstorlekar och genom den accessmetod som valts sker det en automatisk flödeskontroll.

2.8. NIC

NIC (Network Interface Card) är kortet i noden som hanterar nivå ett för LAN. Detta kort måste vara avsett för den typ av nätverk som skall anslutas. Detta kort hanterar även delar av nivå två i OSI modellen.

2.9. Multiplexor

En multiplexor (MUX) är en enhet som koncentrerar flera linjer till en trunklinje och som kommunicerar via denna med en multiplexor i andra änden. Trunklinjen är uppdelad på ett antal kanaler, och detta kan ske med olika tekniker som frekvensdelning, vid bredbandsöverföring, och som tidsmultiplexering där tiden delas upp i tidsluckor som kan utnyttjas av terminalerna, och detta kan ske statistiskt eller statistiskt. Statisk tidsmultiplexering innebär att varje enhet har en bestämd tidslucka till sitt förfogande för att sända sin data. Finns det inget att sända så är denna tidslucka outnyttjad, därav har en statistisk teknik utvecklats som låter de tomma luckorna utnyttjas av någon som har behov. Trunklinjens kapacitet beror på vilken typ av multiplexorteknik som används, och måste i princip vara tillräckligt stor för att klara alla anslutna enheters kapacitet. Om sex enheter är kopplade till MUXén med 4800 bps var, måste trunklinjen klara av $6 \cdot 4800 = 28\ 800$ bps om man inte använder statistisk multiplexeringsteknik. Statistisk teknik bygger på att sannolikheten för att alla enheter skall kommunicera samtidigt är ganska låg, och därför har man lägre kapacitet på trunklinjen än alla enheternas sammanlagda kapacitet eftersom alla tidsluckor kan utnyttjas. Med statistisk teknik måste också MUXen ha buffer för att klara topparna som kan uppstå då alla terminaler vill kommunicera.

MUX används mest i terminalnätverk, men kan användas varhelst man vill minska antalet linjer mellan flera enheter som skall kommunicera. Om MUX använder WAN så är oftast modem inbyggda i enheten.

2.10. Hub/Switch

Detta är en benämning för de enheter som sitter i centrum av ett nätverk, eftersom hub betyder nav, som allt kretsar kring. Noderna är kopplade till hubben via portar som är konstruerade så att man kan koppla loss en kabel från en port utan att störa någon annan port. Den som finns i 10baseT och är en buss logiskt men en stjärntopologi fysiskt. Bussen finns inuti hubben i dess bakplan, och normalt har hubbarna en AUI eller BNC gränssnitt så att man kan kombinera dem så att det blir en trädstruktur. Normalt har hubbarna också en kaskadkoppling så att deras bakplan kopplas samman och man kan kombinera flera hubbar i en stack. Hubben möjliggör att enheter kan kommunicera med varandra genom att omvandla respektive enhets sändsignal till den andres mottagningsignal och vice versa. I en hub sera alla portarna all trafik som finns i hubben, vilket gör att varje port påverkas av trafiken på de andra portarna.

En variant av hub är en switch, som fungerar i stort som en hub men skillanden ligger i att varje port är "isolerad" från de andra portarna så de ser inte de andra portarnas trafik. Se närmare kapitle 3.7.

I token ring nätverk är det ett logiskt ringnätverk men också där ett fysiskt stjärnnätverk och där kallas hubben för MAU (Multistation Acces Unit). En ring måste vara intakt för att någon kommunikation skall kunna ske och den är därför mycket sårbar för avbrott på ringen, fysiskt eller logiskt. MAU har inbyggd förbikoppling av enheterna så att ringen kan hållas intakt.

2.11. Repeater

Repeater används i LAN och är i princip en förstärkare mellan flera nätverkssegment så att man förlänger den tillgängliga längden på nätverket. En repeater består av 2 st. NIC som står i förbindelse med respektive segment. Varje paket tages emot på den ena av NIC och sändes ut på den andra. Vid mottagande NIC kan signalen vara distorderad i olika hög grad, men när den sänds ut igen så är den återställd till normalt. En repeater återgenererar signalen, en förstärkare förstärker signalen men alla störningar förstärks också, därav att det inte är en riktig förstärkare. Det går inte att ha hur många repeaters som helst i ett nätverk. I ett Ethernet-nätverk så tillåts nätverket att vara delat i max fem segment och därmed fyra repeaters. Denna begränsning beror på att det finns en fördröjning i en repeater, och två kommunicerande enheter får ha max fyra repeaters mellan sig. Denna begränsning kan påverkas av nya typer av multiport repeaters, det vill säga en repeater har flera utgångar så segmenten blir parallella istället för seriella och därmed öka antalet möjliga segment utan att bryta mot reglerna.

2.12. Felupptäckt

Felupptäckt på nivå 1 är att upptäcka om förbindelsen bryts under överföringen. Eftersom denna nivå inte kan förstå något sammanhang med bitarna kan det inte ske någon kontroll av bitarna. Undantaget är när modem används som har V.42 eller V.42bis, där modemmet hanterar fel mellan de kommunicerande modemerna, men detta vet inte nivå 1 någonting om, det hanteras helt av de inblandade modemerna.

2.13. Sammanfattning

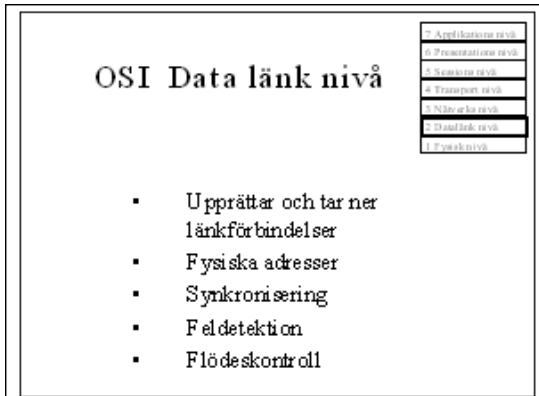
Nivå 1 i OSI modellen beskriver hur kommunicerande enheter kopplas samman fysiskt. Denna nivå är av nödvändighet mycket hårdvara beroende och implementeringen av modellen blir därför med hårdvara som kablar, modem och så vidare. Den hårdvara som väljs för att lösa funktionerna i nivå 1 kommer att påverka hela kommunikationskedjan, eftersom det är ju här som själva kommunikationen sker, och därför är det av yttersta vikt att bra och väl fungerande material väljs. Att spara pengar eller att välja för enkla lösningar på denna hårdvara kan bli mycket kostsamt, detta är ryggraden och den viktigaste länken i kedjan.

Det som skall sändas på kommunikationslänken är för nivå 1 den data som levereras från nivå 2, som sedan adderas till med headers och trailers för att det skall kunna användas på den typ av länk och teknik som används för kommunikationen, till exempel start och stopp bitar för asynkron teknik. Detta omvandlas till elektriska signaler som sedan används på det media som valts.

Vid mottagning omvandlas först de elektriska signalerna till ettor och nollor och sedan så kommer nivå 1 att plocka bort "sina" headers och trailers och lämnar sedan resten som data till nivå 2 för att där vidarebehandlas enligt de regler som gäller där.

Kapitel 3: Nivå 2, data länk nivå

3.1. Uppgift



Data länk nivån har som uppgift att upprätta och kontrollera två kommunikationsenheter som är fysiskt sammankopplade, som benämns en länk. Formaterar bitar från den fysiska nivån till grupper eller ramar (frames) som blir data och dessa ramar (frames) kan vara tecken, block eller paket.

Ansvarar för att dessa datagrupper/frames överförs felfritt på länken, genom att upptäcka fel och att hantera omsändningar.

Arbetar med fysiska adresser som identifierar kommunikationsenheterna som en nod, till exempel ethernetadress på ett lokalt nätverk. Datalänk nivån är den första nivå som kontrollerar att överföringen sker felfritt, denna kontroll är endast på länken, noder som är fysiskt kopplade tillsammans. Om det finns flera noder mellan slutanvändarna så sker kontrollen mellan noderna på nivå 2 och kontrollen mellan slutanvändarnoderna sker på nivå 4.

Nivån är ytterligare uppdelad i två undernivåer, LLC (Logical Link Control), och MAC (Media Access Control) för att hantera i främsta hand LAN.

3.2. LLC - MAC

Data länk nivån är uppdelad i ytterligare två subnivåer för LAN och det är LLC (Logical Link Control) överst mot nivå 3 och MAC (Media Access Control) mot nivå 1. LLC hanterar det som är protokollorienterat i nivå 2, och MAC hanterar det som är orienterat mot det media som används för länken. LLC har den officiella beteckningen 8802.2 och MAC's beteckning beror på vilken typ av nätverk, 8802.3 för CSMA/CD, 8802.4 för token bus, 8802.5 för token ring, 8802.6 för metropolitan area networks (MAN), 8802.7 för slotted ring och 9314 för FDDI. Detta är beteckningar på de dokument som beskriver standarden, och för MAC gäller denna beskrivning även nivå 1.

3.2.1. LLC

Tre tjänster är beskrivna som kan utföras av LLC: Typ 1 är en förbindelsefri tjänst utan bekräftelser eller kontroller, som kallas datagram, typ 2 en förbindelseinriktad tjänst med

kvittens och felkontroll och typ 3 som ett mellanting mellan 1 och 2. I typ 1 kan datagrammen skickas till individuella adresser eller till gruppadresser, medan typ 2 endast kan skickas till individuella adresser. OSI nivå 3 och 4 kallas för protokollstacken, ett samlingsnamn för de protokoll som används på dessa nivåer. LLC's uppgift är att hålla reda på till vilken stack som paketet hör hemma. LLC tillåter flera protokollstackar mot samma MAC.

3.2.2. MAC

Media Access Control handhar de specifika mediarelaterade bitarna i 8802.2 standarden, vilket inkluderar 8802.3 som är buss (Ethernet) CSMA/CD, 8802.4 som är token bus, 8802.5 som är token ring. MAC isolerar de övre skikten från mediarelaterade funktioner och gör att en brygga kan ha en token ring på ena sidan och ett Ethernet nätverk på den andra och klarar av att omvandla mellan dessa. OBS! 8802.x beskrivningen av respektive MAC inkluderar nivå 1 för denna, medan MAC är en beskrivning av nivå 2. Alltså 8802.3 är en beskrivning för nivå 1 och för MAC på nivå 2

3.2.2.1. 8802.3

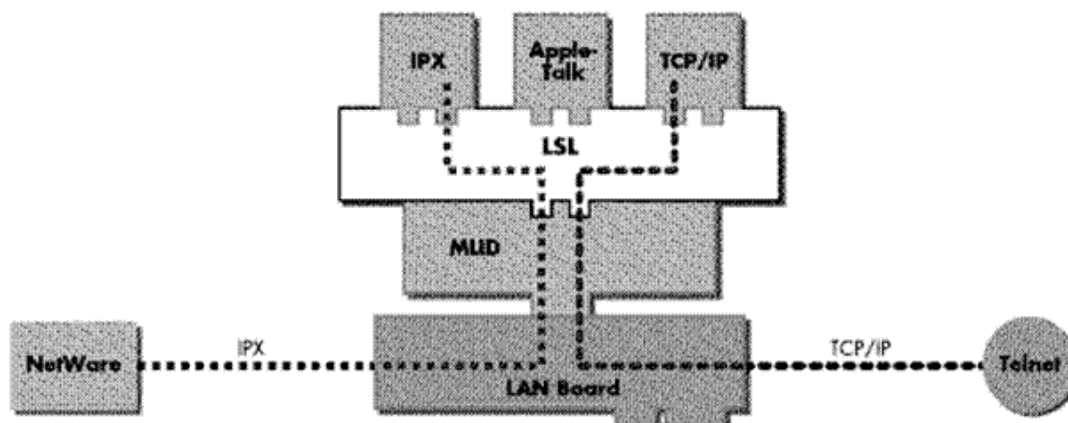
Denna beskriver ett bussnätverk med trafiksättet CSMA/CD. Inom denna finns ett antal typer av ramar som beskrivs som 802.2 som Novell 4.1 nätverk använder som standard, 802.3 även kallad raw, som Novell 3 använder som standard, 802.SNAP som används i Apple nätverk, 802.II (DIX) som används i TCP/IP nätverk. Obs! Ramtyperna heter delvis samma som standardbeteckningen för några nätverkstyper, men är inte samma, se avsnitt 4.4.2.2.

3.2.2.1. 8802.5

Denna beskriver ett token ring nätverk, och även här finns olika typer av ramar som beskrivs som 802.2, 802.SNAP, PC network II, PC network II SNAP, RX-net short, RX-net long och RX-net exception. Obs! Ramtyperna heter delvis samma som standardbeteckningen för några nätverkstyper, men är inte samma, se avsnitt 4.4.2.3.

3.2.3. ODI

ODI (Open Data-Link Interface) är Novells implementering av LLC-MAC och detta har gjort för att kunna ha mer än ett protokoll i nivåerna 3-7 mot samma NIC, till exempel TCP/IP och SPX/IPX kan använda samma NIC och samma nätverk. ODI består av två nivåer, LSL (Link Support Layer) och MLID (Multiple Link Interface Driver).



LSL hanterar till vilken protokollstack som informationen skall till eller kommer ifrån, och antalet protokollstackar som kan var igång samtidigt beror på minnesstorleken i noden. MLID

är NIC drivrutiner, så om man har ett 3C5x9 så måste den drivrutin som laddas till denna vara ODI kompatibel. LSL tillåter flera NIC, vilket gör den till en sorts trafikpolis för paketen som anländer från MAC eller protokollstacken/arna

3.2.4. NDIS

NDIS (Network device Identification Specification) är Microsofts sätt att lösa LLC-MAC i Windows-baserade nätverk. NDIS kan också ha flera protokollstackar i nivåerna 3 - 7 att arbeta mot ett eller upp till fyra NIC. NDIS finns i versioner för Windows NT och för Windows 95, men dessa är inte samma.

3.3. Adressering

Nivå 2 har som uppgift att hantera de fysiska adresserna. För att kunna ha mer än en kommunikationsenhet på länken krävs att varje enhet kan identifieras, och detta i sin tur har lett till att protokollen utvecklats för att hantera detta. En nod är beteckningen på en kommunikationsenhet som har en egen identitet, en adress. Den adress som används i denna nivå är en adress som oftast är hårdvaru-baserad och kan i vissa fall inte ändras, som i Ethernet där tillverkaren av NIC sätter adressen så att alla NIC i hela världen har unika adresser. I terminaler eller terminalemulatorer så kan adressen sättas via den programstyrda konfigurationen. Adresserna har olika längd beroende på vilket länkprotokoll som används, men det grundläggande är att noderna är unika.

Många protokoll har både sänd adress och mottagare adress, och detta används för att veta till vem som man skall kvittera paketen. Den här typen av protokoll används i nätverk där många kommunicerar till många och inte i master-slav situationer.

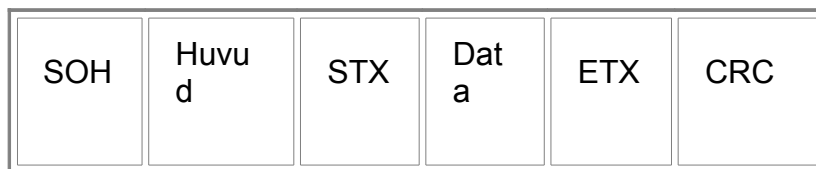
Dessa adresser skall inte sammankopplas med adresser på högre nivåer som är logiska adresser och som identifierar noden logiskt. Dessa logiska adresser måste översättas till fysiska och det sker i nivå 3.

3.4. Länkprotokoll

Protokoll är ju regler som gör att de kommunicerande enheterna kan överföra data, om de använder samma protokoll. Det finns många protokoll och det låter sig inte göras att redovisa alla som finns, utan här redovisas de mest vanliga och de som representerar en viss typ av protokoll. Data länk nivån förstår innebörden av bitarna, eftersom man arbetar på ramar, det vill säga grupper av bitar. Man kan indela protokollen i teckenbaserade och bitorienterade, där det finns likheter men även stora olikheter mellan dem. Protokoll kan använda olika tekniker i överföringen på nivå 1 som asynkron eller synkron överföringsteknik.

3.4.1. Teckenbaserat

Teckenbaserade protokoll betyder att bitarna sätts ihop till 7 eller 8 bitar för att representera tecken enligt ASCII eller EBCDIC tabellerna. Som tidigare redovisats finns det speciella tecken som har speciella funktioner och betydelser, och i teckenbaserade protokoll används de i dessa funktioner. I ett teckenbaserat protokoll är det tecknen och inte positionen som avgör vad som överföres. Nedan följer ett exempel på ett teckenbaserat protokoll, som består av ett huvud och en kropp, där huvudet innehåller adressen och eventuell styrinformation och kroppen innehåller den data som skall överföras.



CRC är Cyclic Redundancy Check och betyder att det är ett speciellt tecken som har skapats genom en algoritm av alla de tecken från STX t.o.m. ETX, och som används för att kontrollera att överföringen av data är korrekt. I huvudet finns adressen till mottagaren och vad det är för typ av meddelande.

Nackdelen med denna typ av protokoll är att i datadelen kan inte vilka teckenkombinationer som helst sändas. Till exempel om en exekverbar fil, en binär fil, skulle överföras blir sannolikheten stor att något av kontrolltecknen finns representerade i datafältet och om det är ETX som finns som bitkombination så tolkas detta som att det är slut på datafältet vilket det inte är och följdriktigt kan inte detta mottagas korrekt. Detta har lett till att nya protokoll utvecklats som är transparenta för data som överföres, det vill säga att datafältet kan innehålla vilken bitkombination som helst.

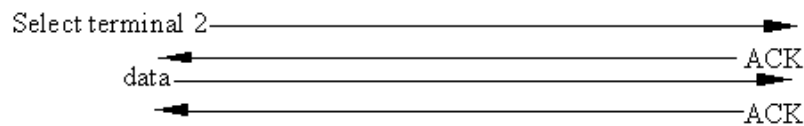
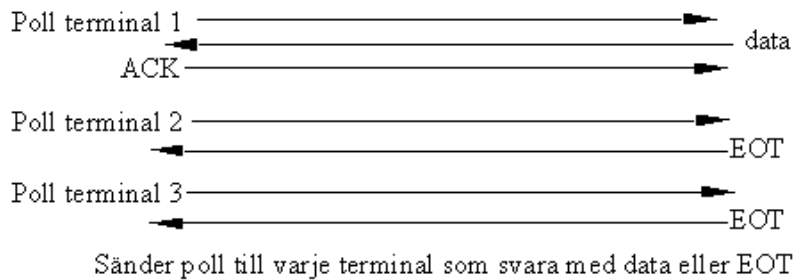
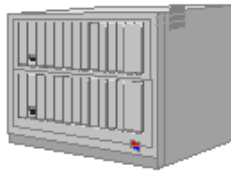
Men det finns ett alternativ i teckenbaserade protokoll för att lösa detta problem med transparent data, och det är med hjälp av ESC tecknet och kallas därför att man använder en escape-sekvens. ESC tecknet plus ett annat tecken definierar att det tecken som kommer efter inte skall behandlas som något kontrolltecken. En escape-sekvens används även för att sända styrinformation till en bildskärm, som till exempel att markören skall flyttas till en viss rad och position på denna, så att data som följer escape-sekvensen hamnar just där. Dessa escape-sekvenser är leverantörsberoende eftersom det inte finns någon standard framtagen. En sådan leverantörs escape-sekvens har blivit defactostandard, och det är Digital Equipments VT100 terminaler.

Observera att det ovan beskrivna protokollet har formen av ett block men måste inte därför vara ett protokoll som använder synkron teknik, det går alldeles utmärkt att använda asynkron teknik eftersom det är ett teckenbaserat protokoll.

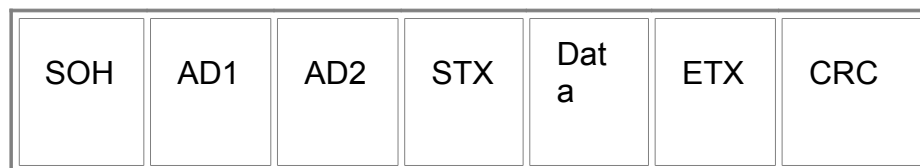
I ett protokoll bestäms även hur fel och omsändningar skall hanteras, och även proceduren för att få sända på nätverket. Med centraliserad styrning och ett teckenbaserat avfrågningsprotokoll finns det ett flöde som bestämmer hur kommunikationen sker och beskrivs nedan.

3.4.1.1. Avfrågning (poll)

I datakommunikationens barndom så var det terminaler som var kopplade till stordatorer i stora terminalnätverk och stordatorn var den som kontrollerade all trafik mellan terminalerna och sig själv. Det innebär att stordatorn är en master, styr vilken och när en terminal kan överföra information. Terminalnätverken var oftast kopplade i ett stjärnnätverk och där endast en terminal per linje fanns. Men en utveckling av terminaler där flera fanns på samma linje kom genom att behovet av att ha en linje med flera dropppunkter. Dessa dropppunkter kunde även vara på långa avstånd från stordatorn. Ett sätt att hantera dessa droppnätverk är via poll/select, som betyder att mastern frågar, poll, om en terminal har något att sända. Detta sätt att avfråga varje terminal kallas också roll-call. Terminalen måste svara på detta med nej eller med att sända data. Mastern frågar i tur och ordning alla terminalerna på linjen om de har något att sända genom att sända en speciell fråga, ett pollpaket. Genom att bestämma i vilken ordning som terminalerna frågas så blir detta även ett sätt att prioritera mellan terminalerna. Om terminalen sänder data måste detta kvitteras, positivt eller negativt, och om det kvitteras negativt måste terminalen sända samma data en gång till. Om mastern vill sända data till en terminal så frågar den först om terminalen är klar för att ta emot data, select, och terminalen svara positivt eller negativt på denna fråga. Om ett positivt svar så sänder mastern sin data till terminalen som måste kvittera detta. Denna procedur beskrivs så här i ett flödesschema.



Terminalen måste alltid vara beredd på att sända meddelandet igen om det mottagas felaktigt i mastern, och får besked om detta genom att mastern sänder ett NAK istället för ACK, dessa är ASCII-tecken. Men även när mastern sänder data till terminalen måste denna kontrollera mottagen data och be om omsändning om den är felaktig. Data som sänds är i formatet



och frågan har följande format, där AD1 och AD2 är ASCII-tecken för att ge terminalen en adress med 2 tecken, till exempel QT.



Genom att förändra i listan av terminaler som skall avfrågas kan en prioritering skapas mellan terminalerna, antingen statiskt eller dynamiskt. Till exempel kan en terminal som inte har något att sända under en viss tid tagas ur listan för en bestämd tid, för att sedan komma med igen och se om den har blivit aktiv. En terminal som ej svarar kan tagas bort från listan en längre tid eftersom den troligen är avstängd, och om den inte svarar måste en timeout situation uppstå, det vill säga varje terminal har en viss tid på sig att svara sedan blir det timeout, och denna brukar vara i storleksordningen sekunder. Om flera terminaler som är avstängda skulle vara med i listan tar det väldigt lång tid mellan förfrågningarna till de andra terminalerna. Man kan också ha de mest aktiva terminalerna först på listan, och efter varje svar med data från någon terminal så börjar avfrågningen från början av listan. Detta ger den sista terminalen mycket liten möjlighet att sända, så avfrågningslistan måste anpassas till den aktuella situationen.

Tiden för en avfrågning kan beräknas enligt följande: Pollmeddelandet är 5 tecken långt, terminalmodemet skall vänta för att kunna svara, och svaret är 1 tecken, indikerande ingen data att sända, mastermodemet skall vänta för att kunna sända nästa fråga. 7 bitars ASCII med paritet och asynkron teknik ger 5*10 bitar för pollmeddelandet + modemvändning 150 ms + 10 bitar svar + 150 ms modemvändning. Överföringshastighet är 9600 bps ger en bittid på 0.1 ms, för exemplet blir det då 5 ms + 150 ms + 1ms + 150 ms = 306 ms. Om då timeout är 1 s så motsvarar detta ca tre förfrågningar, är då flera terminaler avstängda så ökar svarstiderna

med en faktor 2,5 till 3 jämfört med när alla terminaler svarar med att de inte har någon data. Med en skärmbilddata (1920 tecken) som svar blir tiden enligt följande: fråga 5ms + 150 ms modemvändning + $1920 \cdot 1 = 1920$ ms data + 150 ms modemvändning + 1 ms ACK = 2226 ms, 2,2 s. Med 8 terminaler på samma linje så tar det i värsta fall $7 \cdot 2,2$ s = 15,4 s innan den sista har blivit tillfrågad, I bästa fall $7 \cdot 0,3$ s = 2,1 s.

Vändningstiderna i modemmet tar väldigt mycket av svarstiden och genom att införa en annan typ av avfrågning som kallas hubpoll så kan svarstiderna halveras genom att spara en modemvändning per terminal. Hubpoll går till så att mastern frågar först den första terminalen, som sedan skickar frågan vidare till nästa terminal om den inte har något att sända eller efter att den sänt sin data, och så vidare. Det är endast den sista terminalen i kedjan som svara mastern.

3.4.2. Bitbaserat

Bitbaserade protokoll är exempel på transparenta protokoll. Bitbaserat protokoll betyder att positionen av bitarna är den som räknas, det vill säga att ett visst antal bitar in i meddelandet skall det komma ett speciellt fält som har sin egen betydelse och att data som överföres behandlas som bitar. Flera bitprotokoll finns och nedan visas hur HDLC protokollet ser ut. Det är ett protokoll som ISO och CCITT har standardiserat, HDLC för ISO och LAPB för CCITT, men det finns ett nästan exakt lika protokoll som IBM har tagit fram, SDLC. Datafältet måste vara fulla oktetter (8 bitar), och om bitarna i den data som skall överföras inte blir fulla oktetter så sker det en bitfyllnad till en full oktett.

SDLC har obegränsad storlek på datafältet medan LAPB skiljer sig, eftersom detta har fasta storlekar, standard är 128 oktetter men följande kan också användas, 16, 32, 64, 256, 512, 1024. HDLC kan ha längre addressfält, en multipel av en oktett, även kontrollfältet kan vara större, Scx och Rcx kan vara 7 bitar var.

För att kunna skicka vilken data som helst i datafältet så krävs det ju att inga bitkombinationer kan ha någon betydelse för styrningen av kommunikationen. Detta är ju ett stort problem i teckenbaserade protokoll, men här i bitorienterade protokoll finns endast ett bitmönster som kan feltolkas och det är FLAG. Om flaggans bitmönster skulle finnas någonstans i datafältet skulle detta tolkas som att ramen var slut, vilket medför att överföringen misslyckas. För att lösa detta problem används en teknik med nollinsättning, som fungerar så här. Vid varje bitkombination av fem på varandra följande ettor inom datafältet, så sättes en nolla in efter den femte ettan vid sändningen. Detta gör att mottagaren inte kommer att tolka flaggan felaktigt, men denna nolla måste naturligtvis tagas bort på mottagarsidan, och det sker genom att ta bort en nolla som kommer efter fem ettor. Vad händer då om det skulle vara en nolla efter dessa fem ettor? Jo, det blir två nollor vid sändningen, och vid mottagningen så tagas den ena bort och det är som det var från början.

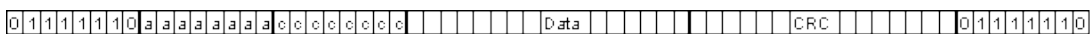
3.4.2.1. HDLC

I ett HDLC nätverk finns det tre typer av noder.

- En primär, som ansvarar för flödeskontroll och felhantering av länken till andra noder. Paket som sänds ut av en primär nod kallas kommando.
-
- En sekundär, som arbetar under kontroll av en primär nod och har inget ansvar för flöde och felhantering. Paket som sändes ut av en sekundär nod kallas svar.
-
- En kombinerad, som är både primär och sekundär nod i samma nod.
-

Noder i ett HDLC nätverk konfigureras som balanserat eller obalanserat, där balanserat innebär en punkt till punkt förbindelse med två kombinerade nodtyper, och där obalanserat menas med en primär och en eller flera sekundära noder i en flerpunkts förbindelse eller punkt till punkt förbindelse. Båda konfigurationerna kan hantera både full och halv duplex.

I en obalanserad konfiguration kan en sekundär nod endast sända om den blivit frågad av primärnoden, primärnoden är som en master i det tidigare redovisade poll/sel protokollet. Primär noden sänder ett paket med pollbiten satt till 1, och det får bara finnas ett paket med en pollbit = 1 vid något tillfälle. Kvittens av paket sker genom att sekvensfälten används för att indikera vilket paket som mottagits och därigenom kvitterar alla fram till detta sekvensnummer minus ett, se nedan bilden på HDLC paketets struktur. HDLC tillåter att flera paket finns utskickade utan att vara kvitterade, med ett fönster avgörs hur många. Storleken på detta fönster beror på om kontrollfältet är 8 bitar eller 16 bitar. Är kontrollfältet 8 bitar har fönstret storleken 7 paket, är kontrollfältet 16 bitar är fönstret 127 paket.



01111110 = FLAG för att synkronisera sändare och mottagare, och att indikera start på blocket

a = Adress field: 2⁷ adressmöjligheter = 128 adresser, expanderbart i 8 bitars intervaller vilket ökar adresseringsmöjligheten. Sista biten i adressen avgör om den är expanderad eller inte, är den noll så finns ytterliggare en oktett, är den 1 är detta sista adressoktetten.

c = Control field: 1100Pccc Unnumbered frame P = poll/final
 expanderbart med 8 bitar c = code

I HDLC finns oftast en individuell adress och en gruppadress på varje nod. Gruppadressen används när paketet skall mottagas av mer än en nod. Eftersom det finns valmöjligheter vad gäller antal adressbitar och kontrollfältets storlek måste båda kommunicerande parter följa samma val, annars kommer paketen att tolkas felaktigt. Storleken på data fältet är inte definierat men det finns en praktisk övre gräns på hur stort detta fält kan vara.

Hur upptäcks CRC , det finns ju inget fält som anger längden på datafältet? Jo eftersom blocket skall avslutas med en flagga, så är det bara att räkna från flaggan framåt i blocket det antal bitar som används för CRC, 16 eller 32. Den avslutande flaggan tillåts vara en startflagga för nästa block. Mer information kan fås på denna länk

<http://en.wikipedia.org/wiki/HDLC>

3.4.2.2. PPP

Point to Point Protocol, är ett protokoll för att koppla upp två noder via ett modem och sedan använda TCP/IP protokollstacken på nivåerna ovanför. PPP kontrollerar först linjen och därefter kan IP paket sändas. PPP använder HDLC som länkprotokoll med adressfältet = 11111111, och kontrollfältet = 00000011.

Detta protokoll används när klienter skall kopplas upp mot Internet via en så kallad "Internet provider", alltså ett företag som erbjuder en tjänst att kunna koppla up sig mot Internet via en modempool.

3.4.2.3. SLIP

Serial Line Internet Protocol, är en äldre variant av PPP, men som inte hade någon kontroll av

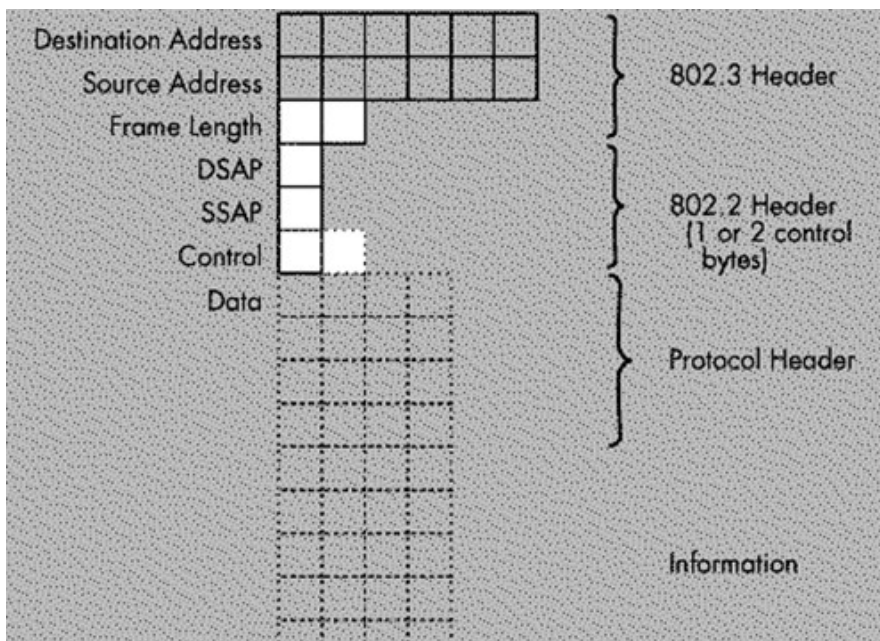
linjen. Har numera liten betydelse vid modemuppkopplingar.

3.4.2.4. CSMA/CD

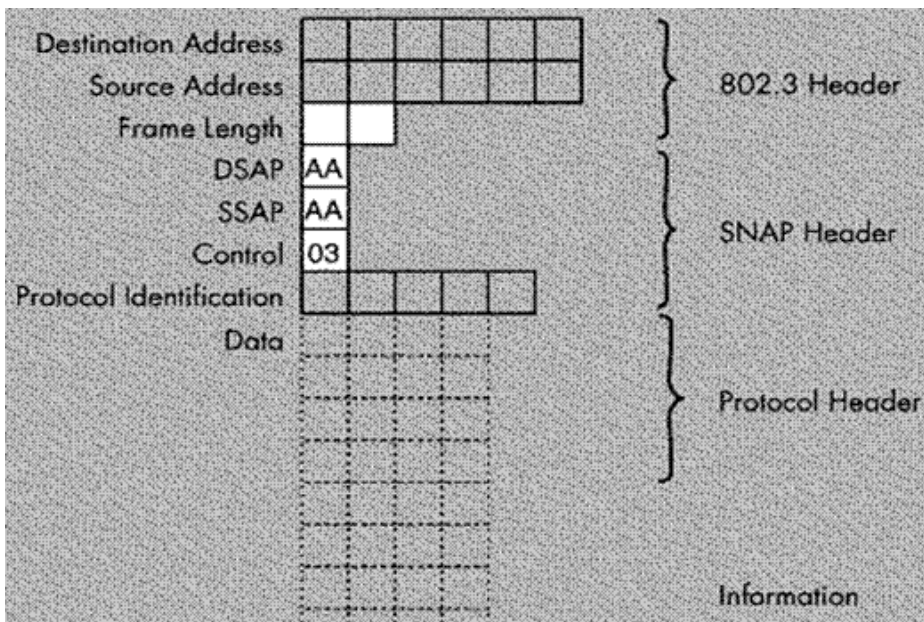
Paketen som skickas på ett Ethernet bussnätverk har en basstruktur av en preamble på 7 bytes vars mönster är alternerande 1 och 0, följt av 1 byte Start of Frame Delimiter (SFD) med mönstret 10101011 som betyder slut på preamble och start på själva datapaketet. Därefter kommer länkprotokollets header, därefter den data som länkprotokollet skall överföra, och sist kommer en kontrollfunktion på paketet. Preamble används för att mottagarens elektronik skall synkronisera sig med den inkommande signalen och den sista biten i SFD signalerar start på datafälten. Preamble hanteras helt av nivå 1. Datadelen som följer efter SFD ser ut på olika sätt för olika typer av nätverk, TCP/IP, Apple, eller Novell. Följande illustrationer visar på hur dessa paket skiljer sig åt. Alla paket avslutas med Frame Check Sequence (FCS) på 4 byte och är en CRC32 felkontroll (se vidare i avsnittet felupptäckt i detta kapitel).

Minimum längd på ett paket är 64 oktetter, och maximum är 1518 oktetter inklusive allt utom preamble och SFD. Minimumlängden är till för att kollisioner skall kunna upptäckas av den sändande noden så att den kan försöka skicka om paketet.

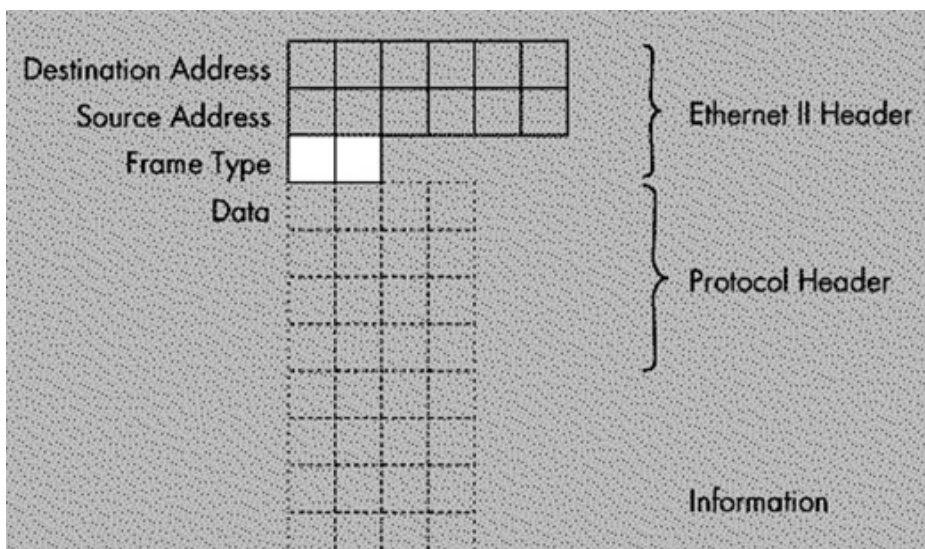
802.2 ramformat: Frame length är mindre än eller lika med 1500 decimal



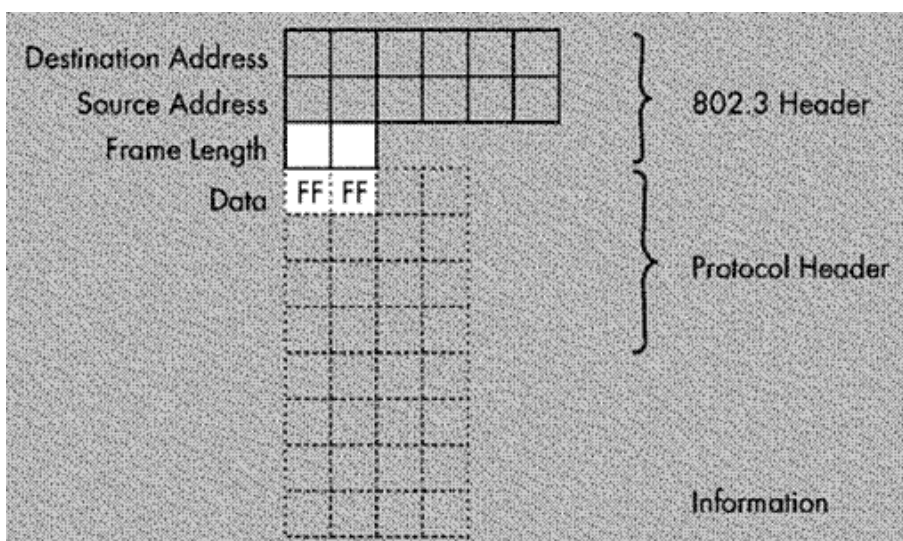
802.SNAP ramformat: Frame length är mindre än eller lika med 1500 decimal



802.11 ramformat: Frame length är mindre än eller lika med 1500 decimal

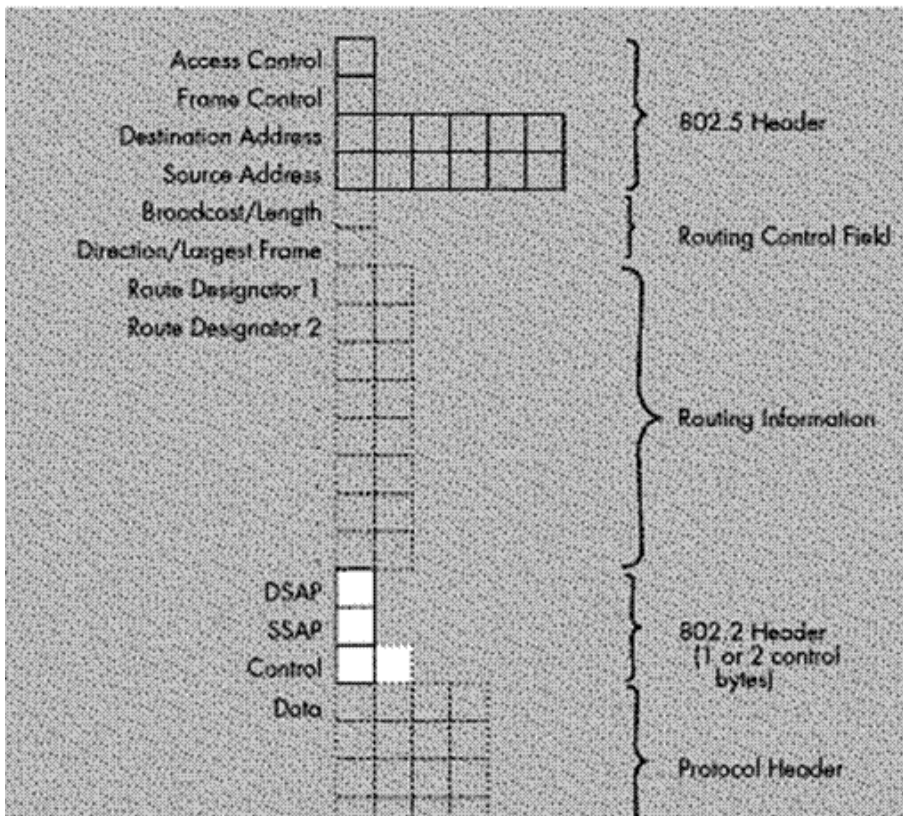


802.3 ramformat: Frame length är mindre än eller lika med 1500 decimal



3.4.2.5. Token ring

Token ring 802.2 ramformat



3.4.2.6. Frame relay

Frame relay är ett paketväxlat nätverk mycket likt X.25 som ITU-T har definierat. X.25 inkluderar nivåerna 1-3 i OSI-modellen, där funktionerna på nivå 3 är övervakning och felhantering. Frame relay har inte denna övervakning och felhantering och blir därigenom mycket snabbare när denna "overhead" ej används. Orsaken till att man beslutat sig för att utelämna "overheaden" är att moderna nätverk har en så låg felfrekvens att det inte motiverar denna extra belastning.

3.4.2.7. ATM

ATM är en teknik som förmedlar fasta paket, celler kallas de i detta sammanhang, med en storlek på 53 oktetter. De första 5 oktetterna är adresseringen och resterande 48 är data. Denna teknik har framtagits för att förbättra möjligheten att transportera data genom de publika nätverken, men har även utvecklats till LAN protokoll. Genom att cellerna har fast längd och är så korta blir kopplingen i växlar snabb.

I LAN används ATM som "backbone", alltså som stamnätverk för att binda ihop flera segment eller subnätverk.

3.4.2.8. ADSL

3.5. Terminaler

Terminaler var de första kommunikationsenheterna, och hade ingen egen beräkningskapacitet utan var helt konstruerade för att hantera ett speciellt länkprotokoll. Idag används PC med program som emulerar terminaler såvida inte applikationen har utvecklats till ett klient-server system. Terminalemuleringsprogrammen uppträder som en "riktig" terminal och måste därför kunna konfigureras på samma sätt som dess ursprungliga produkt. IBM och Digital Equipment har varit framgångsrika med sina terminaler, så framgångsrika att de bildat defacto-standarder

för terminalemuleringsprogram. Emuleringsprogram för terminaler kan vara bra att använda för att testa grundläggande kommunikation mellan två enheter, inklusive modem. Hayes kommandon kan skickas från terminalprogrammet till modemmet och med rätt kommandon kan modemmet konfigureras eller kontrolleras hur det är konfigurerat, se avsnittet om modem.

3.5.1. VT100

Detta är Digital Equipments mest kända enhet för asynkron överföring, och ändvänder ofta som terminal eller i emuleringsprogram. Det finns många parametrar som behövs sättas upp för en terminal så att den fungerar enligt önskemål. Parametrar som bithastighet, hur många stopp bitar det skall vara om det skall vara någon paritet och i så fall vilken typ, udda eller jämn. Emuleringsprogram som kan hantera flera typer av terminaler har speciella konfigurations menyer så att de olika parametrarna kan ställas in på rätt sätt.

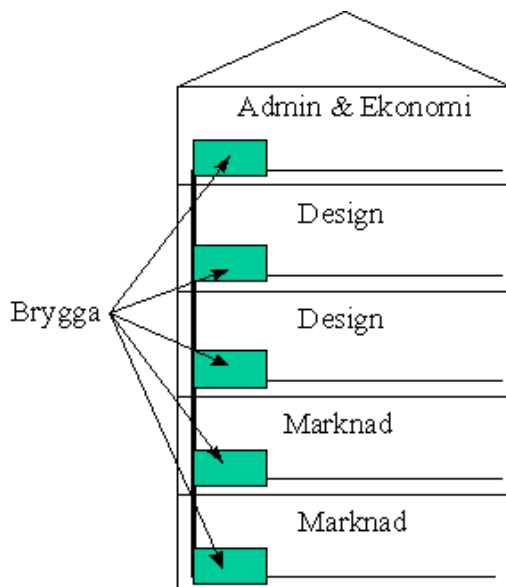
3.6. Brygga

En brygga är en funktion som möjliggör filtrering av paket i ett lokalt nätverk och på så sätt kan användas till att fördela nätverkstrafiken mellan olika delar av ett nätverk. Bryggan har två NIC, och de noder som finns på respektive NIC's nätverk säges höra till samma nätsegment, alltså det finns noder på varje sida om bryggan på var sitt nätsegment.

I ett stort LAN med flera kabelsegment som är sammankopplade med repeaters, är detta ett nätsegment, där alla noder ser alla paket, men varje nod tar bara hand om de paket som är adresserade till sig själva. Detta medför att när en nod sänder måste alla vänta tills den är klar innan någon annan kan sända eftersom alla skall ju använda samma medium för sändning.

Varje paket har ju en sänd adress och en mottagare adress och dessa använder bryggan för sin filtrering. Filtrering går till så att bryggan läser in hela paketet och vet från vilket segment paketet kommer ifrån och på mottagare adressen får reda på vilket segment det skall till. Om sänd och mottagare adress är på olika segment, så sänder bryggan iväg det mottagna paketet igen på den andra NIC'en så att mottagaren kan få det. Om sänd och mottagare adress är på samma nätsegment, det vill säga finns på samma NIC så "slängs" paketet, det skickas inte vidare, det är ju på rätt segment redan. Effekten av detta är att den trafik som skulle finnas på hela nätverket och skulle ses av alla noderna om inte bryggan fanns, kommer nu att fördelas så att bara den trafik som behöver se paketen ser dem.

Detta används för att dela in ett stort nätverk i trafikpooler så att bara den trafik som behövs i ett nätsegment finns där. Ta till exempel ett kontor med flera våningar, där varje våning är en egen avdelning med egna servers och skrivare som just den avdelningen behöver, och mycket sällan har behov att nå data på någon annans server. Då är det lämpligt att separera avdelningarna med en brygga istället för en repeater, så att alla fortfarande kan nå varandra om de vill. Men eftersom varje avdelnings trafik nästan uteslutande kommer att vara mellan sina arbetsstationer och sin server och skrivarna så stannar denna trafik på sin sida av bryggan och belastar inte de andra avdelningarna.



Hur vet bryggan vilket segment en adress hör till? Adresserna i sig ger ingen ledning, eftersom adresserna är allmängiltiga på ett NIC. I bryggan krävs det lite logik som kan räkna ut vilka adresser som finns var, och det är genom att titta på alla paket som finns på respektive NIC och titta på sänd adresserna. En tabell byggs upp för att hålla reda på vilket segment respektive adress finns på, bryggan lär sig sedan efter hand. Från början är tabellen tom, så alla paket skickas vidare, men väldigt snabbt byggs tabellen upp och filtreringen blir effektiv. Varje gång som bryggan startas om av någon anledning så startas även inläringen om.

Bryggan hanterar funktioner i nivå 1 och nivå 2, därmed så har en brygga även funktionen som repeater och har två funktioner i samma "enhet".

3.7. Växel (Switch)

I LAN har en växel (switch) introducerats för att öka kapaciteten i nätverket. Den ser ut som en hub och har nästan samma funktion. I hubben kan alla kommunicera med alla och all trafik passerar alla anslutna noder. I växeln kan alla kommunicera med alla men trafiken finns endast mellan de två kommunicerande noderna. Det är i grund samma teknik som det allmänna telefonnätet och dess växlar. En abonnent ringer ett nummer och via växeln kopplas till den andra abonnenten, och det är endast dessa två sammankopplade abonnenter som kan kommunicera. När en nod vill kommunicera med en annan nod så kopplas dessa samman via växeln, och det är endast dessa två som kan se trafiken.

En växel kopplar samman noderna på två olika sätt. Antingen som "cut-through" eller "store-and-forward". Cut-through innebär att när ett paket mottages på en port så buffras det tills destinationsadressen är mottagen. Då kan växeln avgöra till vilken port som denna nod är ansluten och börjar skicka paketet från buffer till denna port. På detta sätt är dessa två noder anslutna till varandra under kommunikationen. Fördelen med denna teknik är att fördröjningen i växeln är liten, endast tiden tills destinationsadressen är mottagen. Nackdelen är att om det är ett felaktigt paket så skickas detta vidare automatiskt. Store-and-forward innebär att hela paketet tages emot och kontrolleras för korrekthet innan destinationsadressen kontrolleras. Sedan skickas paketet om det var korrekt, till den port där destinationsnoden finns. Fördelen med denna teknik är att det blir en filtrering av dåliga paket. Nackdelen är att det är en stor fördröjning genom att hela paketet buffras innan det sänds ut igen.

Det finns växlar som kan ha en nod per port, kallas single-MAC, och de som kan ha flera noder per port, kallat multi-MAC. I en multi-MAC kan det vara en hub som är kopplad till porten. En växel kan ha både 10 Mbps och 100Mbps i samma enhet, till exempel en port med 100 Mbps som servern är ansluten till och de andra portarna 10 Mbps som klienterna är

anslutna till. Switchen fungerar som en brygga genom att endast de paket som skall mellan två noder kommer fram. Med rätt konfiguration blir ett växlat nätverk effektivt.

3.8. Flödeskontroll

På denna nivå är flödeskontrollen genomförd av protokollet som bestämmer genom de regler som används för att kvittera paket, hantera fel, och hur noderna skall få tillåtelse att sända. Vid poll/sel bestämmer mastern hur ofta som avfrågningen sker, men även de som frågas kan avböja att svara eller svara med passivitet. Genom att kontrollera ordningen på avfrågningen styrs hur ofta en nod får sända. Noden som blir tillfrågad men inte har tid eller kan ta emot eller skicka något kan ju bara skicka ett passivt svar och därigenom påverkar själv flödet.

Hur fel i kommunikationen hanteras påverkar också hur flödet går. Mycket fel i överföringen medför många omsändningar och därigenom hur ofta som noden kan skicka ny data. Data som inte blivit accepterad ligger kvar i ett buffer och detta kan inte frigöras förrän en positiv kvittens kommer eller att sändningen avbrytes p.g.a. för mycket fel.

Hur ofta som paketen skall kvitteras bestämmer även flödet. Protokollet som beskrivs i exemplet med poll/sel, kallas Stop/Wait, eftersom inget nytt kan skickas förrän det förra är positivt kvitterat. Det blir mottagaren som styr flödet på nätverket. Ett annat sätt att kvittera paketen är att ett visst antal paket tillåts sändas innan kvittens måste ske. Antalet paket som kan skickas innan kvittens kan vara fast eller dynamiskt. Fast antal innebär att efter t.ex. 8 paket skickas en positiv kvittens eller omedelbart när ett fel uppstår. Dynamiskt innebär att paketen kvitteras med ett positivt svar på det sist korrekt mottagna paketet inom ett fönster av mottagna paket. Kvittens kan också ske när mottagaren skall skicka något annat paket själv och då automatiskt kvitterar tidigare mottagna paket. Men detta innebär att det inte får gå för lång tid innan att mottagaren skall sända något, i annat fall måste ett speciellt kvittens paket skickas.

När flera paket finns som inte är kvitterade, måste varje paket kunna identifieras, så att det går att hålla reda på vilka som är OK och vilka som måste sändas om. Detta sker genom att man använder sekvensnummer på varje paket, och detta läggs i headern för nivå 2. Det ankommer på protokoll på högre nivå att kontrollera att alla delar av ett meddelande har mottagits, vilket medför en sekvenskontroll på högre nivå och detta skall inte blandas samman med denna typ av sekvensnummer.

3.9. Felupptäckt

Nivå 2 är den första nivån (detta beror naturligtvis från vilket håll man ser på modellen) som kontrollerar hur och om data kommit fram på länken. Denna kontroll beror på vilket protokoll som används och därmed vilken teknik som dessa utnyttjar. Kontrollen har två faser, den första är vid sändningen, då en metod används för att skapa någon form av kontrollmekanismen som sedan även sändes tillsammans med den ursprungliga data. Fas två är vid mottagningen, då motsvarande metod används för att kontrollera den mottagna data och jämföra med den mottagna kontrollmekanismen. I fas två bestäms vad som skall ske om det har mottagits felaktig data, det normala är att begära omsändning av den aktuella data.

Den enklaste formen av kontroll sker i teckenbaserade protokoll som använder sig av VRC (Vertical Redundancy Check), som är paritetskontroll på varje tecken, och pariteten kan vara udda eller jämn. Udda paritet betyder att summan av alla bitar, inklusive paritetsbiten i ett tecken skall vara just udda, om de inte är det så blir paritetsbiten ett för att uppfylla kravet. På samma sätt är jämn paritet uppbyggd, med skillnaden att summan skall vara jämn. Lagg märke till att start och stopp bitar inte räknas in i ett tecken i detta sammanhang, de är ju bortkopplade på nivå 1.

Nästa sätt som används i teckenbaserade protokoll är LRC, Longitudinal Redundancy Check, som är paritetskontroll av alla tecken i ett block. LRC kan precis som VRC vara udda eller jämn. Ett block består av många tecken, uppdelat i dels data dels i styrinformation, och LRC är kontroll på datadelen i blocket. LRC är ett tecken som läggs till sist i blocket, hur detta tecken skall se ut beror på data i blocket, det blir ett paritetstecken. På samma bit i varje tecken i hela datadelen av blocket, säg teckenbit nummer 1, summeras alla ettor och en etta eller nolla sättes i paritetstecknets motsvarande teckenbit för att uppnå udda eller jämn paritet på denna bit, som blir paritetsbit för alla datatecknets teckenbit nummer 1. På samma sätt göres med de andra bitarna i varje tecken i blocket, utom för den vertikala paritetsbiten för dessa tecken (det går att kombinera VRC och LRC), och ett nytt tecken har skapats med dessa paritetsbitar. Detta tecken kan bli vilket tecken som helst i ASCII tabellen, och om VRC används så sätts en VRC paritets bit på för detta tecken, som tidigare beskrivits och därmed blir det paritetsbit på paritetstecknet.

VRC och LRC kan kombineras som sagt, men de förblir ganska enkla medel på kontroll. De kan upptäcka en bit fel, men om flera bitar är fel så upptäcks det inte, därför har mera sofistikerade metoder tagits fram för att upptäcka flera bitar som är fel. Dessa metoder använder speciella algoritmer på den data som skall kontrolleras, och genererar sedan ett bitmönster som sändes tillsammans med data. Antalet bitar i detta mönster beror på val av metod, men det vanligaste är 16 bitar eller 32, där 32 bitar upptäcker fler bitfel. Dessa metoder är mest vanliga i bitorienterade protokoll. Varje oktett i datadelen går igenom algoritmen som producerar ett bitmönster som sedan sändes sist i blocket, detta bitmönster beror då på den data som skall skickas. Mottagaren gör på samma sätt med varje oktett i datadelen, sedan jämförs det mottagna bitmönstret med det som genererats vid mottagningen. Om mottagningen har skett felfri blir denna jämförelse lika, om inte föreligger ett eller flera bitfel i mottagningen. Den vanligaste metoden kallas CRC (Cyclic Redundancy Check), och finns i CRC16, som mest används i WAN, och CRC32 som mest används i LAN. Det finns också algoritmer som kan korrigera de felaktiga bitarna, visserligen inte alla typer av bitfel, därigenom sparas en eller flera omsändningar. Denna typ av felkorrigering kallas för Hamming kod. Modem i nivå1 som använder MNP-4 eller V42 har denna typ av felkorrigering

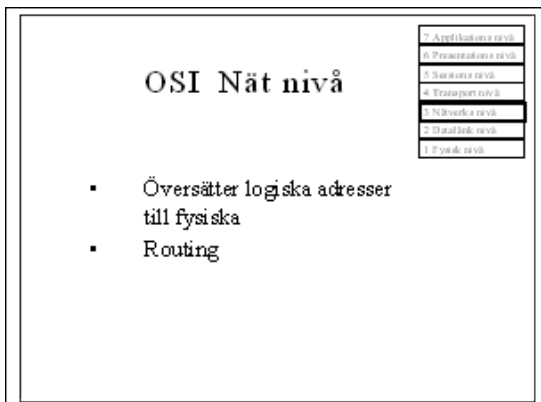
3.10. Sammanfattning

I länknivån tillsammans med nivå 1 (fysiska) är där det sker, alltså kommunikationen mellan sammankopplade enheter. Det är den första kontrollnivån vid mottagning att kommunikationen har fungerat felfritt. Denna kontrollfunktion garanterar att den data som lämnas till nästa nivå, nivå 3, är mottagen korrekt. Vid sändning placeras kontrollfunktioner i paketet så att mottagaren kan utföra sin kontrollfunktion. Kontrollfunktionerna är flera typer beroende på vilket protokoll som används, som VRC, LRC och CRC.

På denna nivå finns ett antal enheter som hjälp till att strukturera länken, som brygga i LAN och MUX i WAN. Dessa enheter har inte egna adresser, fastän de finns på nivå 2. De är alltså inte noder i ett nätverk eller nät.

Kapitel 4: Nivå 3, nätverk.

4.1. Uppgift



Nätverks nivån skall hålla reda på nätverket så att kommunikationen kan nå fram även via flera mellannoder om det behövs, alltså, att flytta paket mellan ändenheterna. Nätverksnivån är ansvarig för att omvandla logiska adresser (nätverksadresser) till länkadresser, som sedan gives till länknivån så att denna kan adressera rätt nod.

Ett nätverk består av sammankopplade enheter som tillsammans kan kommunicera med varandra direkt eller via en brygga eller router. Flera nätverk av denna sort kan sedan kopplas samman till större nät, till exempel Internet, och de olika nätverken blir som noder i detta större nät. Nätverk är sammankopplade noder där alla noder kan kommunicera direkt med varandra. Nät är sammankopplade nätverk som kan kommunicera med varandra. Observera att ett nät inte innebär att nätverken skall vara direkt kopplade till varandra, det är just denna bit som gör att nivå 3 finns till så att paket kan gå via mellanliggande nätverk. Routing är en viktig funktion om det finns mer än ett nätverk, som gör att noder i de mindre nätverken kan kommunicera med varandra. Routing är den mekanismen som håller reda på de olika nätverken, hur dessa är sammankopplade i större nät och vilka vägar det finns mellan de olika nätverken. Routing är att välja väg i nätet.

Problemet är att vid kommunikation mellan noder som ej finns i samma nätverk, måste vägen mellan dessa noder räknas ut på något sätt. Det kan finnas flera vägar att nå destinationen, det kan finnas olika snabba länkar på dessa vägar, det kan finnas olika mycket trafik på dessa.

Man kan ställa upp kriterier som skall uppfyllas vid vägvalet, till exempel

- Kortaste avstånd i tid eller "hops"(antal routrar som skall passeras)
- Minsta länkelastning.
- Fel på mellanliggande noder.
- Sekretess.

4.2. Nätverk - nät

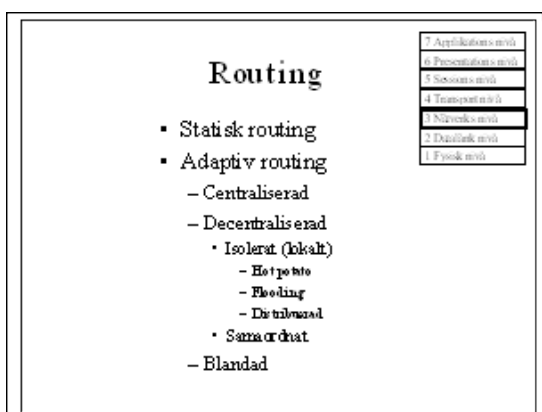
För att klara ut begreppen i detta avsnitt, så menas med ett nätverk att det är sammankopplade noder som alla kan kommunicera direkt med varandra, förutom via eventuella bryggor/repeatrar, alltså ett LAN. Ett nät är att flera nätverk är sammankopplade på något sätt och att alla kan kommunicera med alla, men inte så att alla kan kommunicera med

alla direkt. Varje nätverk måste ha en egen identitet som nätverk och som gör det unikt i det stora sammankopplade nätet, nätverket har en adress, en nätverksadress. Denna adress är helt frikopplad från nodadressen som ju endast identifierar noden i ett nätverk. Nätverksadresserna är logiska till skillnad från länkadresserna som är fysiska.

I nätet med de sammankopplade nätverken kan man se varje nätverk som en nätverksnod på detta nät och som då har en egen adress i nätet. Paket som skall sändas mellan två nätverk har då en nätverksadress att tillgå som gör att paketet kan komma fram. Men eftersom alla nätverk inte är direkt sammankopplade med varandra kan ett paket behöva passera flera mellanliggande nätverk för att komma fram till sitt destinationsnätverk. Varje nätverk måste då kunna hantera dessa paket, som ju inte har någon adress inuti det egna nätverket, utan dessa paket skall i princip bara skickas vidare.

Nätverk som ingår i sammankopplade nät på detta beskrivna sätt är det allmänt kända Internet ett bra exempel. För att alla nätverk skall kunna kommunicera med alla andra nätverk, fastän de inte alla är direkt sammankopplade, innebär att flera av de ingående nätverken måste ha mer än en ingång/utgång från sitt eget nätverk. Denna ingång/utgång är normalt en enhet som kallas router och som innehåller funktioner för nivå 1, 2 och 3 enligt OSI modellen. Där nivå 3 funktioner är till för att kunna hantera passerande paket och att kunna skicka paket mot rätt adressat från det egna nätet.

4.3. Routing



Routing är att göra vägval för paket. En nod i ett nätverk som skall kommunicera med en nod i ett annat nätverk måste få detta paket att gå via routern, som antingen är sammankopplad med det nätverk där destinationsnoden finns eller med ett nätverk som via andra mellanliggande nätverk når det nätverk där destinationsnoden finns. Nätverksadressen blir den faktor som bestämmer om ett paket skall lämna det egna nätverket eller stanna inom det. Varje paket har i headern för nivå 3 en nätverksadress för sändaren och en nätverksadress för mottagaren. Noden som skall sända detta paket ser om mottagaren har samma nätverksadress som sig själv, som då innebär att paketet kan skickas direkt till mottagaren genom att använda länkadressen. Har mottagaren en annan nätverksadress så måste paketet lämna det egna nätet och då måste det först gå till routern som ju är utgången från det egna nätet.

Ett paket som skall lämna det egna nätverket måste först skickas till routern, och denna är ju en nod på det egna nätverket, så genom att ange routerns länk adress för detta paket så kommer det att sändas dit. Routern tar emot paketet eftersom det på länknivå är adresserat till den, men eftersom destinationens nätverksadress finns kvar i paketet, så när routern tagit emot paketet och tittar i headern för nivå 3 så upptäcks att detta paket inte var avsett för routern själv utan skall skickas vidare. Om routern har två kommunikationskanaler så är det lätt att räkna ut att paketet måste sändas vidare på den andra kanalen i förhållande

till den kanal som paketet anlände på. Den andra kommunikationskanalen kan vara det nätverk som mottagarnoden finns på, i så fall kommer routern att sätta denna nods länkadress på paketet och uppdraget är utfört, mottagaren har fått paketet. Det innebär att för att två noder skall kommunicera och där endast en router finns mellan dem så krävs det att paketet först går till routern och sedan till mottagaren.

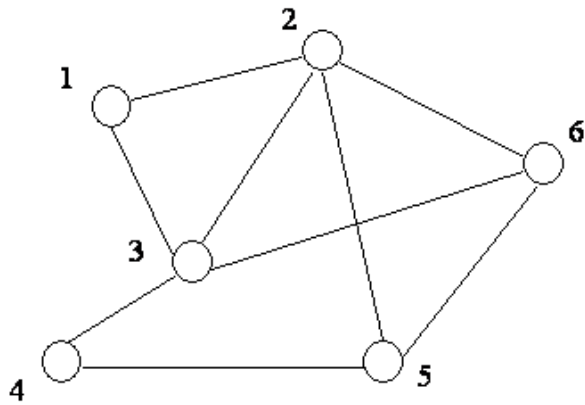
Har routern flera kommunikationskanaler så uppstår en svårare valsituation, vilken kanal skall det skickas vidare på? Om mottagaren finns på någon av de tillgängliga kommunikationskanalerna så skickas naturligtvis paketet dit. Men om mottagaren finns lite längre bort i nätet, vilken väg skall då väljas? Det finns olika sätt som detta kan lösas på men grunden är att routern måste ha information som underlag för beslutet. Denna information är till exempel om trafikbelastningen på respektive kanal, om kanalerna är operativa, om paketet måste gå på en viss kanal därför att den har bättre säkerhet och så vidare. Denna information skaffar sig routern genom att kommunicera med andra routrar och utbyta information om vilka nätverk som de är kopplade direkt till och så vidare. Denna kommunikation sker med speciella protokoll som endast finns på denna nivå och det är nivån själv som kan initiera denna kommunikation mellan routrar. Informationen lagras i tabeller som anlitas för att göra vägvalet, tabellerna kan vara statiska eller dynamiska. Informationen som hämtas är för de närmaste routrarna, inte för hela nätet med alla dess olika nätverk och vägar mellan dem. Sådana tabeller skulle bli oerhört stora och det skulle ta lång tid att leta i dem för att finna det bästa vägvalet.

Två metoder kan urskiljas hur detta skall ske, det finns statisk och adaptiv (dynamisk) routing. I statisk routing lagras förutbestämda vägval i en tabell som sedan används för att slå upp hur man skall skicka meddelandet vidare ut på nätet. Tabellerna skapas vid ett tillfälle av någon administratör som har kunskap om nätet och hur det ser ut i den närmaste omgivningen. Om det sker någon förändring i kommunikationskanalerna så måste tabellen göras om, och detta är omständligt och tar lång tid eftersom det göres manuellt.

Adaptiv routing kan i sin tur delas upp i centraliserad styrning, decentraliserad styrning och en blandning av dessa två. I centraliserad styrning meddelar routrarna då och då sin belastning och eventuella avbrott till en nätcentral, eller en speciell router som tar på sig denna uppgift, som då räknar om bästa vägval och skickar sedan detta tillbaka till tabellerna i routrarna. Nackdelar är att det ger extra belastning på nätet eftersom en hel del paket kommer att finnas ute på nätet som inte förmedlar någon "nyttolast" men tar kapacitet av nätet. Nätcentralen måste vara dimensionerad för att hantera dessa beräkningar och de paket som är inblandade. Nya och gamla tabeller kommer att finnas ute i nätet samtidigt och skapar mer-sändningar genom att fel vägval göres och omsändningar får ske.

Decentraliserad styrning kan delas upp i isolerad adaptiv styrning (lokal information), och samordnad adaptiv styrning (lokal information + grannarnas information). Isolerad adaptiv styrning är att det endast är den egna routerns situation som får betydelse på vägvalet och kan göras på olika sätt, till exempel "hot potato" som skickar ut inkommande paket på den kanal som har den kortaste kön oberoende av andra kriterier. Eller "flooding", som skickar paketet till alla kanalerna i routern, detta innebär att flera paket som innehåller samma information kan anlända till mottagaren som då får hantera detta. Flooding är ett sätt som militären vill använda sig av för att om paketet skickas ut på alla tillgängliga kanaler så kommer det fram så småningom om det finns någon väg öppen mellan sändare och mottagare. Nackdelen med flooding är uppenbar, det belastar nätet med oerhört många extra paket som egentligen inte borde vara där, men i krigstider när kommunikationslänkar kan slås ut och sedan kopplas upp igen är det ett acceptabelt sätt.

Vid samordnad adaptiv styrning så skaffar sig routern information om och från de routrar som är direkt kopplade till sig, och denna information tages med vid vägvalet.



Figuren ovan är ett exempel på nätverk som är sammankopplade till ett större nät, där siffrorna representerar de ingående nätverken som är ett LAN och med en router som utgång från respektive nät.. En tabell med möjliga vägar för alla nätverkens kommunikation med varandra följer här.

Från 1 till	Hopp	Via	Från 2 till	Hopp	Via	Från 3 till	Hopp	Via
2	1		1	1		1	1	
2	2	3	1	2	3	1	2	2
2	3	3,6	1	3	6,3	1	3	6,2
2	4	3,4,5	1	4	5,4,3	1	4	4,5,2
2	4	3,6,5	1	4	5,6,3	1	4	6,5,2
2	5	3,4,5,6	1	5	6,5,4,3	1	5	4,5,6,2
3	1		3	1		2	1	
3	2	2	3	2	1	2	2	1
3	3	2,6	3	2	6	2	2	6

3	4	2,5,4	3	3	5,4	2	3	4,5
3	5	2,6,5, 4	3	4	6,5, 4	2	4	6,5
4	2	3	4	2	3	2	4	4,5,6
4	3	2,3	4	2	5	4	1	
4	3	2,5	4	3	1,3	4	3	2,5
4	4	2,6,3	4	3	6,3	4	3	6,5
4	4	2,6,5	4	3	6,5	4	4	1,2,5
4	4	3,2,5	4	4	3,6, 5	4	4	2,6,5
4	4	3,6,5	4	5	1,3, 6,5	4	5	1,2,6,5
4	5	3,2,6, 5	5	1		5	2	2
4	5	3,6,2, 5	5	2	6	5	2	4
5	2	2	5	3	3,4	5	2	6
5	3	2,6	5	3	3,6	5	3	1,2
5	3	3,2	5	4	1,3, 4	5	3	2,6
5	3	3,4	5	4	1,3, 6	5	3	6,2

5	3	3,6	5	4	6,3, 4	5	4	1,2,6
5	4	2,3,4	6	1		6	1	
5	4	2,3,6	6	2	3	6	2	2
5	4	3,2,6	6	2	5	6	3	1,2
5	5	2,6,3, 4	6	3	1,3	6	3	2,5
6	2	2	6	4	3,4, 5	6	3	4,5
6	2	3	6	4	5,4, 3	6	4	1,2,5
6	3	2,3	6	5	1,3, 4,5	6	4	4,5,2
6	3	2,5						
6	4	3,2,5						
6	4	3,4,5						
6	5	2,3,4, 5						
6	5	2,5,4, 3						
6	5	3,4,5, 2						

Från
4 till

Hopp

Via

Från 5
till

Hopp

Via

Från 6
till

Hopp

Via

1	2	3	1	2	2	1	2	2
1	3	3,2	1	3	2,3	1	2	3
1	3	5,2	1	3	4,3	1	3	2,3
1	4	3,6,2	1	3	6,2	1	3	3,2
1	4	5,6,2	1	3	6,3	1	3	5,2
1	4	5,6,3	1	4	2,6, 3	1	4	5,4,3
1	5	3,6,5, 2	1	4	4,3, 2	1	5	2,5,4,3
1	5	5,6,3, 2	1	5	4,3, 6,2	1	5	3,4,5,2
2	2	3	2	1		1	5	5,4,3,2
2	2	5	2	2	6	2	1	
2	3	3,1	2	3	4,3	2	2	3
2	3	3,6	2	3	6,3	2	2	5
2	3	5,6	2	4	4,3, 1	2	3	3,1
2	4	3,6,5	2	4	4,3, 6	2	4	3,4,5
2	4	5,6,3	2	4	6,3, 1	2	4	5,4,3
2	5	5,6,3,	3	2	4	2	5	5,4,3,1

		1						
3	1		3	2	6	3	1	
3	3	5,2	3	3	6,2	3	2	2
3	3	5,6	3	4	6,2, 1	3	3	2,1
3	4	5,6,2	4	1		3	3	5,2
3	4	5,2,1	4	3	2,3	3	3	5,4
3	5	5,6,2, 1	4	3	6,3	3	4	5,2,1
5	1		4	4	2,1, 3	4	2	3
5	3	3,2	4	4	2,6, 3	4	2	5
5	3	3,6	4	4	6,2, 3	4	3	2,3
5	4	3,1,2	4	5	6,2, 1,3	4	3	2,5
5	4	3,2,6	6	1		4	4	2,1,3
6	2	3	6	2	2	4	4	5,2,3
6	2	5	6	3	2,3	4	5	5,2,1,3
6	3	3,2	6	3	4,3	5	1	
6	3	5,2	6	4	2,1,	5	2	2

					3			
6	4	3,1,2	6	4	4,3, 2	5	3	3,2
6	4	3,2,5	6	5	4,3, 1,2	5	3	3,4
6	4	5,2,3				5	4	2,3,4
6	5	3,1,2, 5				5	4	3,1,2
6	5	5,2,1, 3				5	5	2,1,3,4

Som framgår av dessa tabeller så kan ett paket gå flera vägar för att komma fram till destinationen. Om då minst hopp är det viktigaste kriteriet så kommer den väg att väljas som uppfyller det kravet och alla de andra vägarna används inte. Om någon kommunikationslänk bryts så kan hopp kolumnen sättas till ett värde, till exempel 99 och då betyder det att den vägen är blockerad. En routingtabell för nätverk 3's router skulle kunna se ut så här, lägg märke till att det är endast vägen till närmaste router och antal hopp som behövs finnas med. När paketet kommer till den anvisade routern så har den sin egen tabell som den gör sina beslut från.

Till	Hopp	Via		Till	Hopp	Via
1	1	1		1	2	2
1	3	6		1	4	4
2	1	2		2	2	1
2	2	6		2	3	4
4	1	4		5	2	2
5	2	4		5	2	6

6	1	6		6	2	2
6	3	1		6	3	2
6	3	4				

Genom att addera en kolumn till som får ett värde beroende på vilken typ av länk som finns mellan nätverken, kan ytterliggare avväganden ske vid vägvalet. Om länken mellan 3 och 2 är en WAN länk med hastigheten 28800, och länken mellan 3 och 6 och likaså mellan 6 och 2, är en 2 Mbps länk, är det snabbare att skicka paketen från 3 via 6 till 2 än att sända dem direkt till 2 via den långsamma länken. Kolumnen i tabellen har ett mycket lägre värde på den snabba länken än på den långsammare, och vägvalet göres av den länk som har lägsta värdet. Men om nu länken mellan 6 och 2 blir hårt belastad med trafik från flera av de andra nätverken, så kan dels själva länken bli en flaskhals, dels kan routern i nätverk 6 bli en flaskhals, den måste ju hantera trafiken på länkarna. Vid en sådan situation kan routern i 6 informera de andra om situationen så att färre paket skickas den vägen, och det göres genom att ändra på värdet i tabellen i varje router för hastigheten på denna länk, så att det blir högre än tidigare, och router 6 skickar meddelande om att den har mycket att göra och varje gång ökas värdena i tabellen. När värdet blivit så högt att andra vägar väljs minskar trafiken vid den belastade länken och när den sjunkit till normal nivå så meddelar router 6 detta till de andra och värdet i tabellen kan minskas igen. Hur mycket värdena skall vara och hur mycket de skall minskas med varierar från protokoll till protokoll, även den tid som går innan andra meddelas om förändringar i belastning varierar.

När ett paket skall från 5 till 1 till exempel så läggs alla värdena för länkarna ihop på de olika möjliga vägarna, och den som får lägsta värdet blir det vägval som göres. Dessa vägval göres i varje router, och därför kan ett vägval som var bra från en router bli ändrat på vägen till mottagaren.

Routing är bara möjligt med kommunikationskanaler som inte kräver att mottagare och sändare skall ha en förbindelse mellan sig, det kallas att kanalen är förbindelsefri, annars kallas den förbindelse orienterad. Om flera paket som hör till samma meddelande skickas, kan varje paket ta en egen väg genom nätet, och det kan bli så att de anländer i olika ordning som de sändes. Normalt så är de protokoll som finns på nivå 3 av den typ som just arbetar med förbindelsefri överföring och det är protokoll högre upp som får ta hand om att paketen sätts ihop i rätt ordning till ett meddelande.

Om nu det beskrivna nätet, med nätverken 1 - 6, endast är en del av ett större nät och där nätverk 4 har en eller flera vägar till detta större nät, så uppstår naturligtvis frågan hur skall tabellerna se ut då? Tabellerna kompletteras i varje router med en rad som säger att alla andra nätverksadresser än 1 - 6 skall skickas till nätverk 4's router.

Till	Hopp	Via		Till	Hopp	Via
1	1	1		1	2	2
1	3	6		1	4	4

2	1	2		2	2	1
2	2	6		2	3	4
4	1	4		5	2	2
5	2	4		5	2	6
6	1	6		6	2	2
6	3	1		6	3	2
6	3	4		any	1	4

Så med andra ord, om inte nätverksadressen är känd, så skicka paketet till en standard router som då förhoppningsvis känner adressen. Eller så skickar denna i sin tur paketet till sin standard router och om nätverket finns så kommer paketet så småningom fram till en router som känner till adressen. Naturligtvis förutsätter detta att de inblandade routrarna har byggt upp sina tabeller på rätt sätt och att de har kommunicerat med varandra.

4.3.1. Routing protokoll

Routing protocol	
▪	EGP External Gateway Protocol
▪	RIP Routing Information Protocol
▪	IGRP Internal Gateway Routing Protocol

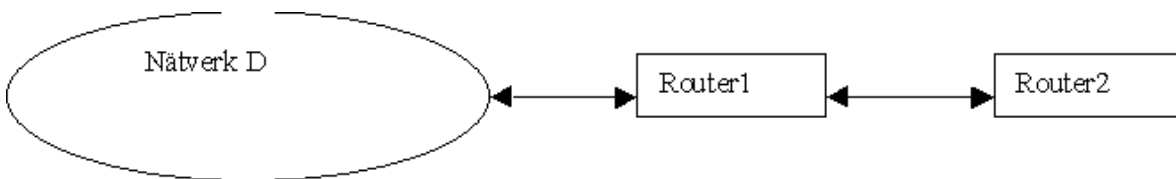
EGP används av ArpaNet

RIP används av IP och av IPX men är då inte samma protokoll, bara samma namn.

IGRP används av CISCO (en leverantör av routrar)

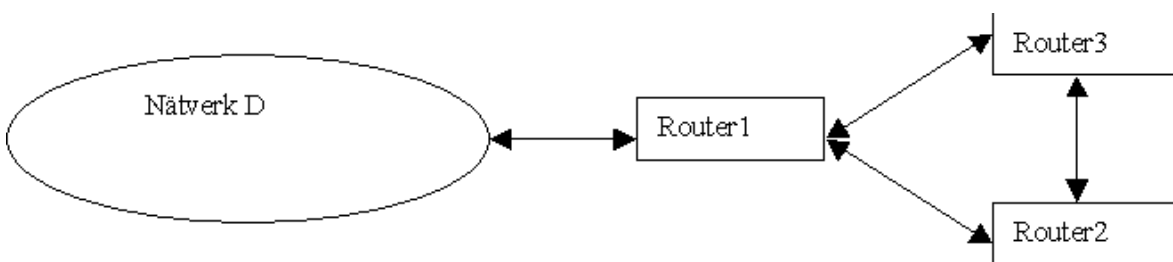
Det finns speciella protokoll som är anpassade för att utbyta information mellan routers, en del som är standardiserade en del som blivit defacto-standarder. Leverantörer av routrar har av naturliga skäl varit aktiva på att finna effektiva routingprotokoll och därför finns det sådana som endast fungerar i just den leverantörens utrustning.

Routingprotokoll är till för att uppdatera routingtabellerna. RIP i TCP/IP världen är ett distansvektor protokoll som varje router periodiskt uppdaterar sina tabeller genom broadcast meddelanden var 30:e sekund till de andra deltagande routrarna, med ett speciellt paket. I detta paket finns alla denna routers kända nätverk och vilken route som finns till dessa. De mottagande routrarna uppdaterar sina egna tabeller, och finns det mer än en väg till ett nätverk så sparas endast den som har den kortaste vägen dit. Distansvektor är antalet hopp eller antalet routrar som måste passeras för att komma till destinationsnätverket, och RIP använder alltid den som har det minsta antalet hopp. Max antal hopp som tillåts i ett RIP routing nätverk är 16, och varje gång som paketet passerar en router så räknas ett fält upp i headern. Har inte paketet anlänt till destinationsnätverket när denna räknare är på 16 så slängs paketet. Om en väg till ett nätverk ej finns med i sex följande broadcastmeddelande så tages denna väg bort från tabellen, efter 180 sekunder. Genom dessa broadcastmeddelanden så får varje router reda på vilken väg som skall väljas för att nå ett annat nätverk. Om en väg till ett nätverk slutar att fungera av någon orsak så kan det uppstå routing loopar, eller om man konfigurerar sitt nät felaktigt. En loop kan uppstå om det finns en situation som på bilden



och länken från router 1 till nätverk D fungerar inte. Då tar router 1 bort vägen till nätverk D från sin tabell. När sedan router 2 informerar router 1 genom broadcastmeddelande sin tabell där vägen till nätverk D fortfarande är kvar eftersom den inte tages bort förrän efter 180 sekunder, och är 2 hopp, så kommer router 1 upptäcka att det finns en väg från router 2 till nätverk D som är 2 hopp avlägsen. Då lägger router 1 till denna väg i sin tabell med tillägget att den är 3 hopp avlägsen. Effekten av detta är att paket som kommer via router 2 och skall till nätverk D skickas till router 1, som i sin tur skickar det till router 2, och så har vi en loop. Nu blir ju inte paketet långlivat utan efter 16 gånger fram och tillbaka så slängs ju paketet, men det blir extra belastning på länken. Det finns sätt att förhindra routingloopar genom "split horizon", som innebär att en route inte skickas tillbaka till den router som man har lärt sig av att routen finns.

Om det är tre routrar kopplade så här



kan det fortfarande uppstå en loop med "split horizon". Om länken till nätverk D går ner så kommer inte router 2 och router 3 att få någon information från router 1 att det finns en väg till nätverk D och efter 180 sekunder så försvinner detta från tabellerna. Om nu detta sker i router 2 före det sker i router 3, och om router 3 hinner skicka ett broadcastmeddelande till router 2 att det finns en väg till nätverk D som är 2 hopp avlägsen, uppdateras tabellen i router 2 att det finns en väg till nätverk D som är 3 hopp avlägsen. Router 3 skickar inte detta till router 1 eftersom den har lärt sig av denna att det finns en väg till nätverk D, "split horizon". Nu kommer router 2 att skicka ett broadcastmeddelande till router 1, den har ju inte lärt sig av denna den "nya" vägen med 3 hopp, att det finns en väg till nätverk D som är 3 hopp avlägsen, och router 1 uppdaterar sin tabell. Nu har vi en loop mellan 1, 2 och 3 för paket som skall till nätverk D. Detta kan förhindras med "poison reverse" som innebär att om man har lärt

sig en väg till en destination från en router så om det uppstår en ny väg till samma destination så meddelar man tillbaks att den har en kostnad av 16 hopp, och därigenom förhindras att paketen hamnar i en loop. I detta fall så när router 2 skall skicka ett meddelande till router 1 att det finns en väg till nätverk D så anges hoppet till 16 eftersom router 2 har lärt sig av router 1 från början att det fanns en väg till nätverk D.

Det finns flera typer av routingprotokoll som har andra algoritmer än distansvektor, som t.ex. link state för OSPF (Open Shortest Path First). Det finns routrar som fungerar inom ett autonomt system, det vill säga ett företagsnät, och de protokoll som används av dessa kallas Interior Gateway Protocol. Routrar som finns mellan olika autonoma system, där kallas protokollen för Exterior Gateway Protocol.

4.3.2. Nätverksadresser

4.3.2.1. IP

Beroende på vilket typ av protokoll som används på nivå 3 så ser nätverksadresserna olika ut. Nätverksadressen på TCP/IP protokollsviten kallas IP-adress eftersom den används i IP protokollet, och har en speciell utformning. Adressdelen i headern är på 32 bitar och dessa bitar delas vanligen in i fyra delar med 8 bitar i varje. Sedan visas dessa 8 bitars decimala värde och mellan de fyra delarna sättes punkter. 8 bitar blir värden från 0 till 255, men 0 och 255 har speciell betydelse i dessa adresser. Exempel på en IP adress är 194.16.201.137. Den första delen, i exemplet 194, är uppdelad så att den avgör vilken typ av adress det är, A, B, C eller D. Typ A är adresser från 1 till 126, typ B är från 128 till 191, typ C är från 192 till 223 och typ D från 224 till 254. Mer exakt så definieras typen av 32-bitars adressens msb's värde enligt följande :

MSB		
0	Type A	1 - 126
10	Type B	128 - 191
110	Type C	192 - 223
1110	Type D	224 - 254

IP adressen är uppdelad i en nätverksadress och en host adress (eller kanske mer riktigt nodadress, men i TCP/IP vokabulären heter det host) och typen avgör vilken del av IP adressen som är vad. För typ A är nätverksadressen den första delen, och de andra tre delarna är hostadresser. Detta medför att ett typ A nätverk kan ha ~ 16 miljoner hostar i sitt nätverk, nätverksdelen är 8 bitar och hostdelen är 24 bitar. En typ B nätverksadress består av de två första delarna av IP adressen och de andra två är hostadresser, vilket ger ~ 65 tusen hostar i nätverket, nätverksdelen är 16 bitar och hostdelen är 16 bitar. Typ C betyder att nätverksadressen är de tre första delarna och den sista delen är hostadress, det ger 254 hostar i detta nätverk, nätverksdelen är 24 bitar och hostdelen är 8 bitar. Exempeladressen är en typ C och nätverksadressen är då 194.16.201 och hostadressen är 137. Värdet 0 och 255 används för att ange alla hostar på nätverket, och värdet 127 i första delen med 0 i de resterande delarna är en adress till hosten själv, och används för att testa hostens

kommunikation.

4.3.2.1.1. Subnät och subnätmask

Typ A och B nätverk har ju väldigt många noder och för att dela upp detta i lämpliga områden så kan subnätverk användas. Normalt finns det en subnätmask som anger vad som är vad i IP-adressen, vad som är nätverksadress och vad som är hostadress. För typ A är subnätverksmasken 255.0.0.0 , för typ B är den 255.255.0.0 och för typ C är den 255.255.255.0. Det innebär att en etta markerar nätverksadress och en nolla markerar hostadress. Om man vill dela upp ett nät i subnätverk så markerar man i subnätverksmasken med ettor vad som används för detta. Exempel, en typ B adress som 139.27.0.0 skall delas upp i flera subnätverk och man väljer att använda 6 bitar av hostadressen för detta. Då väljs de 6 mest signifikanta bitarna av hostadressen för att identifiera subnätverken, det blir 62 subnätverk, subnätverk 0 och med alla ettor är inte tillåtna. Varje subnätverk får 1022 hostar, och subnätverksmasken sätts till 255.255.252.0

Första subnätverket blir då 139.27.4.1 till 139.27.7.254, andra blir 139.27.8.1 till 139.27.11.254, tredje 139.27.12.1 till 15.254, och så vidare det sista blir 139.27.248.1 till 139.27.251.254. Mellan dessa subnätverk måste finnas en eller flera routrar som sammankopplar dessa subnätverk till ett nätverk, 139.27.0.0 Om detta nätverk skall kopplas upp mot andra nätverk så används inte subnätverken mellan dessa utan det är endast inom respektive nätverk som subnätverken finns. Det innebär att det finns en "gränsrouter" som är ingång till och utgång från det stora nätverket och det är endast den som känner till de andra routrarna som kopplar samman subnätverken. Det kan finnas mer än en gränsrouter, varje måste då känna till det "inhemska" nätverkets routrar. Det är bara gränsroutern som känner till nätverken utanför det egna.

139.27.4.1	10001011	00011011	00000100	00000001
Subnetmask	11111111	11111111	11111100	00000000
Nätverksadress	nnnnnnnn	nnnnnnnn		
Subnät adress			111111	
Hostadress			hh	hhhhhhh

4.3.2.2. Novell

I Novells nätverk byggs nätverksadressen upp av flera delar, dels en del som är numret på nätverket som det definieras för Novell själv, dels nodadressen (länkadressen) och så en socket adress. Socket adressen är att den här kommunikationen använder en process som identifieras med denna adress. Varje nivå erbjuder ju tjänster till den övre nivån, och flera kommunikationer kan pågå "samtidigt" och för att identifiera dessa används socketadress så att rätt paket kommer till rätt adress. Nodadressen i ett Ethernet nätverk är NIC'ens länkadress.

Adressen är upp byggd enligt följande :

- Fyra byte nätverks-nummer
- Sex byte nod-nummer (länkadress)
- Två byte socket-nummer

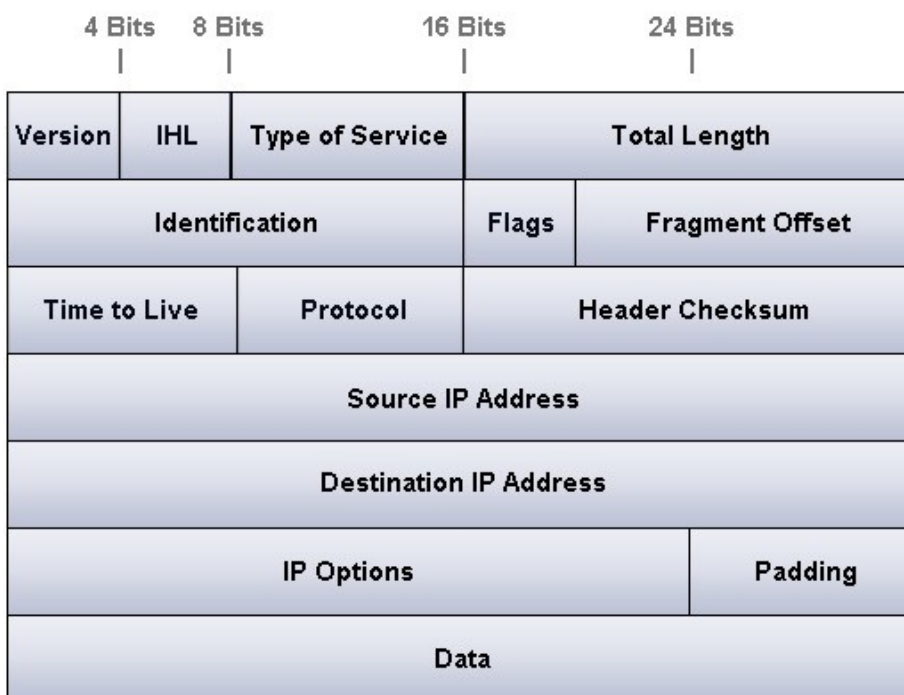
Exempel på en fullständig nätverksadress : FEDCBA98 1A2B3C5D7E9F 0453

4.4. Exempel på protokoll

4.4.1. IP

Internet Protocol, är ett standardiserat protokoll på Internet och är ett förbindelsefritt paketorienterat protokoll som är en del av TCP/IP protokollsviten. Paketerna kallas datagram enligt den terminologi som finns inom Internet. Det är tillåtet att fragmentera ett datagram på denna nivå. Vilket betyder att mottagaren måste veta detta och kunna sätta ihop fragmenten till ett komplett datagram. IP-headern är den del av datagramet som protokollet lagrar sina olika värden för att lösa sina uppgifter.

4.4.1.1. IP header



Version: 4 bitar
Version fältet indikerar vilket format som används, två format finns 4 och 6. Detta beskriver format 4.

Version	Description
0	reserved.
1 2 3	
4	IP, Internet Protocol.

5	ST, ST Datagram Mode.
6	SIP, Simple Internet Protocol. SIPP, Simple Internet Protocol Plus. IPv6, Internet Protocol.
7	TP/IX , The Next Internet.
8	PIP, The P Internet Protocol.
9	TUBA
10 - 14	
15	reserved.

IHL: 4 bitar IP header längd (Internet Header Length) är längden på antalet rader i header, där en rad är 32 bitar. Notera att minimum värde för en korrekt header är 5.

Type of Service: 8 bitar Typ av tjänst (The Type of Service) anger vilken typ av service som önskas för detta datagram. Nätverk kan erbjuda tjänsten precedence, som innebär att detta datagram ses som viktigare än andra datagram. Typ av tjänst används för att precisera hur datagrammet skall hanteras vid överföringen genom nätverket.

Bits 0-2: Precedence

Bit 3 0 = Normal Delay, 1 = Low Delay.

Bit 4 0 = Normal Throughput, 1 = High Throughput.

Bit 5 0 = Normal Reliability, 1 = High Reliability

Bit 6 0 = Normal monetary cost 1 = Minimize monetary cost

Bit 7 Reserved for Future Use = 0.

Precedence.3 bits.

Value	Description
0	Routine.
1	Priority.
2	Immediate.
3	Flash.
4	Flash override.
5	CRITIC/ECP.
6	The Internetwork control.
7	The Network control.

Användningen av Delay, Throughput, och Reliability indikatorerna kan öka

kostnaderna för tjänsten. I vissa nätverk kan bättre prestanda uppnås för en av dessa parametrar på bekostnad av sämre prestanda på en annan. Med undantag för mycket speciella situationer bör högst två av dessa tre parametrar användas. The Network Control precedence valet är avsett för nätverk och inte nät. The Internetwork Control valet är endast avsett för gateway (router) control.

Total length. 16 bits. Längden på hela IP paketet

Identification. 16 bits. Används för att identifiera fragmenterade datagram så att mottagaren kan sätta ihop dem i rätt ordning.

Flags. 3 bits. Används för att identifiera fragmenterade datagram så att mottagaren kan sätta ihop dem i rätt ordning.

00	01	02
R	DF	MF

R = reserverad, måste vara noll
DF=0 = Tillåt fragmentering, **DF=1** = Ingen fragmentering.
MF=0 = Sista fragmentet, **MF=1** = Fler fragment finns.

Fragment Offset. 13 bits. Pekare som indikerar var i datagramet som detta fragment skall vara. Mäts i hur många 8 oktetter (64 bitar) som detta fragment finns från början av det ursprungliga datagramet. För det första fragmentet är detta fält 0.

TTL, Time to Live. 8 bits. Detta fält anger hur länge ett datagram får befinna sig i trafik på nätverket/nätet. Varje nod som tar in detta datagram och processar det i nivå 3, skall minska detta fält med 1. När fältet är 0 och datagramet inte har nått sin destination slängs datagramet. Fältet är till för att hantera datagram som ej kan levereras till destinationen, de har förmodligen hamnat i en loop mellan olika routrar.

Protocol. 8 bits. Anger vilket protokoll som används på nivå 4 för detta datagram

Value	Protocol	References
0	HOPOPT, IPv6 Hop-by-Hop Option.	RFC 1883
1	ICMP, Internet Control Message Protocol.	RFC 792
2	IGAP, IGMP for user Authentication Protocol. IGMP, Internet Group Management Protocol. RGMP, Router-port Group Management Protocol.	
3	GGP, Gateway to Gateway Protocol.	
4	IP in IP encapsulation.	
5	ST, Internet Stream Protocol.	
6	TCP, Transmission Control Protocol.	

7	UCL, CBT .	
8	EGP , Exterior Gateway Protocol.	
9	IGRP , Interior Gateway Routing Protocol.	
10	BBN RCC Monitoring.	
11	NVP , Network Voice Protocol.	
12	PUP.	
13	ARGUS.	
14	EMCON, Emission Control Protocol.	
15	XNET , Cross Net Debugger .	
16	Chaos.	
17	UDP , User Datagram Protocol.	
18	TMux , Transport Multiplexing Protocol.	
19	DCN Measurement Subsystems.	
20	HMP , Host Monitoring Protocol.	
21	Packet Radio Measurement.	
22	XEROX NS IDP.	
23	Trunk-1.	
24	Trunk-2.	
25	Leaf-1.	
26	Leaf-2.	
27	RDP , Reliable Data Protocol.	
28	IRTP , Internet Reliable Transaction Protocol.	
29	ISO Transport Protocol Class 4.	
30	NETBLT , Network Block Transfer.	

31	MFE Network Services Protocol.	
32	MERIT Internodal Protocol.	
33	DCCP , Datagram Congestion Control Protocol.	
34	Third Party Connect Protocol.	
35	IDPR , Inter-Domain Policy Routing Protocol.	
36	XTP , Xpress Transfer Protocol.	
37	Datagram Delivery Protocol.	
38	IDPR , Control Message Transport Protocol.	
39	TP++ Transport Protocol.	
40	IL Transport Protocol.	
41	IPv6 over IPv4.	
42	SDRP , Source Demand Routing Protocol.	
43	IPv6 Routing header.	
44	IPv6 Fragment header.	
45	IDRP, Inter-Domain Routing Protocol.	
46	RSVP , Reservation Protocol.	
47	GRE , General Routing Encapsulation.	
48	DSR , Dynamic Source Routing Protocol.	
49	BNA.	
50	ESP , Encapsulating Security Payload.	
51	AH , Authentication Header.	
52	I-NLSP, Integrated Net Layer Security TUBA.	
53	SWIPE, IP with Encryption.	

54	NARP , NBMA Address Resolution Protocol.	
55	Minimal Encapsulation Protocol .	
56	TLSP, Transport Layer Security Protocol using Kryptonnet key management.	
57	SKIP.	
58	ICMPv6 , Internet Control Message Protocol for IPv6. MLD , Multicast Listener Discovery.	
59	IPv6 No Next Header.	
60	IPv6 Destination Options.	
61	Any host internal protocol.	
62	CFTP.	
63	Any local network.	
64	SATNET and Backroom EXPAK.	
65	Kryptolan.	
66	MIT Remote Virtual Disk Protocol.	
67	Internet Pluribus Packet Core.	
68	Any distributed file system.	
69	SATNET Monitoring.	
70	VISA Protocol.	
71	Internet Packet Core Utility.	
72	Computer Protocol Network Executive.	
73	Computer Protocol Heart Beat.	
74	Wang Span Network.	
75	Packet Video Protocol.	
76	Backroom SATNET Monitoring.	

77	SUN ND PROTOCOL-Temporary.	
78	WIDEBAND Monitoring.	
79	WIDEBAND EXPAK.	
80	ISO-IP.	
81	VMTP, Versatile Message Transaction Protocol.	
82	SECURE-VMTP	
83	VINES.	
84	TTP.	
85	NSFNET-IGP.	
86	Dissimilar Gateway Protocol.	
87	TCF.	
88	EIGRP.	
89	OSPF, Open Shortest Path First Routing Protocol. MOSPF, Multicast Open Shortest Path First.	
90	Sprite RPC Protocol.	
91	Locus Address Resolution Protocol.	
92	MTP, Multicast Transport Protocol.	
93	AX.25.	
94	IP-within-IP Encapsulation Protocol.	
95	Mobile Internetworking Control Protocol.	
96	Semaphore Communications Sec. Pro.	
97	EtherIP.	
98	Encapsulation Header.	

99	Any private encryption scheme.	
100	GMTP.	
101	IFMP , Ipsilon Flow Management Protocol.	
102	PNNI over IP.	
103	PIM , Protocol Independent Multicast.	
104	ARIS.	
105	SCPS.	
106	QNX.	
107	Active Networks.	
108	IPPCP , IP Payload Compression Protocol.	
109	SNP, Sitara Networks Protocol.	
110	Compaq Peer Protocol.	
111	IPX in IP.	
112	VRRP , Virtual Router Redundancy Protocol.	
113	PGM , Pragmatic General Multicast.	
114	any 0-hop protocol.	
115	L2TP , Level 2 Tunneling Protocol.	
116	DDX, D-II Data Exchange.	
117	IATP, Interactive Agent Transfer Protocol.	
118	ST, Schedule Transfer.	
119	SRP, SpectraLink Radio Protocol.	
120	UTI.	
121	SMP, Simple Message Protocol.	

122	SM.	
123	PTP , Performance Transparency Protocol.	
124	ISIS over IPv4.	
125	FIRE.	
126	CRTP, Combat Radio Transport Protocol.	
127	CRUDP, Combat Radio User Datagram.	
128	SSCOPMCE.	
129	IPLT.	
130	SPS, Secure Packet Shield.	
131	PIPE, Private IP Encapsulation within IP.	
132	SCTP , Stream Control Transmission Protocol.	
133	Fibre Channel.	
134	RSVP-E2E-IGNORE .	
135	Mobility Header .	
136	UDP-Lite , Lightweight User Datagram Protocol.	
137	MPLS in IP.	
138	MANET Protocols.	
139	HIP , Host Identity Protocol.	
140 - 252		
253 254	Experimentation and testing.	
255	reserved.	

Header checksum. 16 bits. En kontrollsumma på denna header, alltså inte på datafältet. Eftersom vissa fält (time to live) ändras under transporten mellan destination och sändare måste den räknas om efter varje nivå 3 process.

Source IP address.32 bits. IP-adressen för den sändande noden

Destination IP address.32 bits. IP-adressen för destinationsnoden.

Options. En variabel som kan vara 0 eller innehålla ett värde beroende på vilken typ av val.

00	01	02	03	04	05	06	07
C	Class		Option				

C, Copy flag.1 bit.
Indikerar om option skall kopieras till alla fragmenten.
0 = Ingen kopiering,
1 = Kopiera.

Class.
2 bits.

Value	Description
0	Control.
1	Reserved.
2	Debugging and measurement.
3	Reserved.

Option.5 bits.

Option	Copy	Class	Value	Length	Description	References
0	0	0	0	1	End of options list .	RFC 791
1	0	0	1	1	NOP .	RFC 791
2	1	0	130	11	Security .	RFC 1108
3	1	0	131	variable	Loose Source Route .	RFC 791
4	0	2	68	variable	Time stamp .	RFC 781 , RFC 791
5	1	0	133	3 to 31	Extended Security .	RFC 1108
6	1	0	134		Commercial Security.	
7	0	0	7	variable	Record Route .	RFC 791
8	1	0	136	4	Stream Identifier .	RFC 791 , RFC 1122
9	1	0	137	variable	Strict Source Route .	RFC 791
10	0	0	10		Experimental	

					Measurement.	
11	0	0	11	4	MTU Probe . (obsolete).	RFC 1063
12	0	0	12	4	MTU Reply . (obsolete).	RFC 1063
13	1	2	205		Experimental Flow Control.	
14	1	0	142		Expermental Access Control.	
15	0	0	15		ENCODE.	
16	1	0	144		IMI Traffic Descriptor.	
17	1	0	145		Extended Internet Protocol.	RFC 1385
18	0	2	82	12	Traceroute .	RFC 1393
19	1	0	147	10	Address Extension .	RFC 1475
20	1	0	148	4	Router Alert .	RFC 2113
21	1	0	149	6 .. 38	Selective Directed Broadcast Mode .	RFC 1770
22	1	0	150			
23	1	0	151		Dynamic Packet State.	
24	1	0	152		Upstream Multicast Packet.	
25	0	0	25		QS, Quick-Start.	RFC 4782
26 - 29						
30	0	0	30		EXP - RFC3692-style Experiment.	RFC 4727
30	0	2	94		EXP - RFC3692-style Experiment.	RFC 4727
30	1	0	158		EXP - RFC3692-style Experiment	RFC 4727

30	1	2	222		EXP - RFC3692-style Experiment.	RFC 4727
----	---	---	-----	--	---------------------------------	----------

Padding. Variabel längd. Används för att säkerställa att data startar på en 32 bits gräns.

4.4.2. ARP (Address Resolution Protocol)

Ett protokoll i TCP/IP protokollsviten, som används för att få reda på länkadressen till en viss IP-adress. Nivå 3 är ansvarig för att omvandla logiska adresser, som IP-adressen, till en länkadress som nivå 2 sedan använder för att sända paketet till rätt nod. Om inte nivå 3 vet länkadressen till den IP-adressen så måste ARP användas för att få denna uppgift. Det skickas ett speciellt paket till alla noder, och den nod som känner igen sin IP-adress, svarar med ett paket där länkadressen finns angiven.

4.4.3. RARP (Reverse Address Resolution Protocol)

Detta protokoll är motsatsen till ARP, det vill säga att få reda på en IP-adress för en viss länkadress.

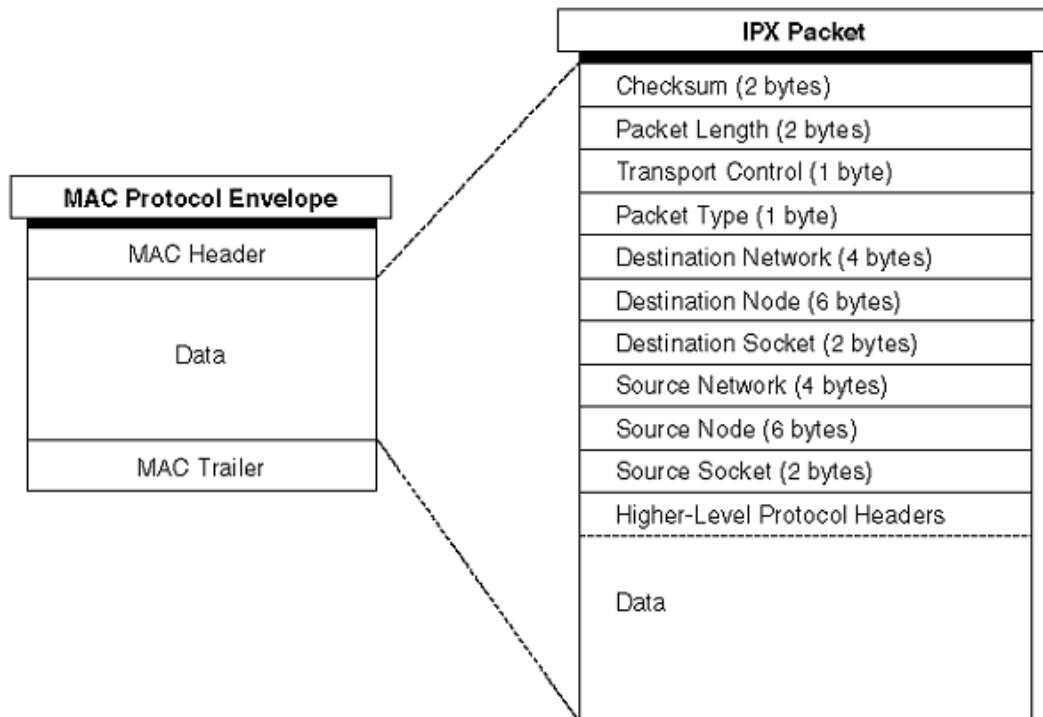
4.4.4. ICMP (Internet Control Management Protocol)

Ett protokoll i TCP/IP protokollsviten, som används för att meddela avvikelser på nivå 3. Till exempel att en nod har skickat ett paket till en router för leverans till en annan nod, men routern upptäcker att det finns en annan router som är bättre för att nå destinationsnoden. Då skickas information om detta till den sändande noden om att i fortsättningen skall alla paket som skall till den avsedda destinationsnoden, skickas via en annan router, och denna finns då angiven.

Det finns en intressant "applikation", eller rättare en funktion, som heter PING, och den använder sig av ICMP för att kontrollera kommunikationen mellan två noder. Dessa noder kan vara åtskilda av routrar. Ping fungerar så, att ett speciellt paket sänds till destinationen som automatiskt returnerar samma paket till avsändaren. Finns det en kommunikationsväg mellan dessa noder så får man reda på detta och även den tid som paketet tog för att göra "resan".

4.4.5. IPX

IPX används av Novell i deras Netware produkter, och är ett förbindelsfritt protokoll.



Checksum.

Checksum används av NetWare SFT IIITM (System Fault Tolerant™) software, NetWare 4, och nyare versioner av NetWare shell (NETx.COM). Äldre versioner av NetWare använder inte IPX checksum och måste ha detta fältet till 0xFFFF.

Packet Length

Längden på hela paketet angiven antal bytes, vilket innebär längden av header plus längden av data. Minimum paket längd är 30 bytes (för IPX headern).

Transport Control

Antalet routers som ett paket har passerat på vägen till sin destination. Sändande nod sätter alltid Transport Control fältet till noll när paketet byggs. När en router tar emot paketet och det krävs vidareändning, alltså skall routas vidare, så räknas detta fält upp med 1 innan det sänds vidare.

Packet Type

Typ av service erbjuden eller begärd av paketet.

Destination Network

Nummer på det nätverk som destinationsnoden är ansluten till. Om detta fält är noll (0), är destinationsnoden på samma nätverk som sändaren. NOTERA: IPX har inte ett broadcastnätverksnummer (som 0xFFFFFFFF). Men ett nätverksnummer av 0xFFFFFFFF har en speciell betydelse i vissa sammanhang (i samband med RIP routing protokoll). Därför skall inte denna nätverksadress användas till något nätverk.

Destination Node

Fysisk adress för destinationsnoden. En nodaddress 0xFFFFFFFFFFFFFFF (alltså sex bytes av 0xFF) blir ett broadcastpaket till alla noder på destinationsnätverket.

Destination Socket

Socketadressen för paketets destinationsprocess. NOTERA: IPX har inte ett broadcastsocketnummer (som 0xFFFF).

Source Network

Nummer på det nätverk som sändande nod är ansluten till.

Source Node

Fysisk adress för den sändande noden. En broadcastadress är inte tillåten.

Source Socket

Socketadressen för den process som sänder paketet.

Higher-Level Protocol Headers

Headers för övre nivåer av NetWare protokoll, som till exempel NCP eller SPX. Dessa headers är en del av data för IPX paketet.

NetWare Socket Nummer och Processer

Socket Number	Process
0x451	NetWare Core Protocol (NCP)
0x452	Service Advertising Protocol (SAP)
0x453	Routing Information Protocol (RIP)
0x455	Novell NetBIOS
0x456	Diagnostics
0x9001	NetWare Link Services Protocol (NLSP)
0x9004	IPXWAN Protocol

Socket nummer mellan 0x4000 och 0x7FFF är dynamiska sockets; dessa används av workstations för att kommunicera med file servers och andra nätverksnoder. Socket nummer mellan 0x8000 och 0xFFFF är "välkända" sockets; dessa har tilldelats sin adress av Novell för speciella processer. Till exempel, 0x9001 är socket nummret som identifierar NLSP.

4.4.6. Datapak (X.25)

X.25 är ett paketförmedlande WAN. Det använder sig av virtuella kretsar som innebär att vid uppkoppling genom nätet skapas en virtuell väg genom nätet, och denna får en identitet som alla paket sedan använder under resten av kommunikationen. På nivå 2 används en variant av HDLC protokollet som kallas LAPD, men med fast paketstorlek som är 128 byte som standard. Paketstorleken kan "förhandlas" till annan paketstorlek vid uppkopplingen. På nivå 1 används X.21 gränssnittet. X.25 kan ha flera förbindelser på samma "linje" samtidigt, genom att ha flera virtuella kretsar. Om inte X.21 finns tillgängligt kan RS232 gränssnittet användas och detta kallas X.32 (uppringd) eller X.28 om noden ej själv kan paketera, och då kopplas noden till en PAD, som då utför paketeringen.

X.25 är byggd för att säkerställa att paketen kommer fram i WAN-nätverket och har därför ganska mycket kontrollfunktioner på nivå 3 för att uppnå detta. Anledningen till detta är att när X.25 specificerades så var inte dessa WAN-nätverk speciellt tillförlitliga. Numera är det en annan situation, och därför har X.25 tappat till frame relay som helt har slopat kontrollen, och alltså endast finns på nivå 2 och 1. Framerelay är betydligt snabbare än X.25 på grund av denna "overhead" som kontrollfunktionerna innebär. Se 4.4.2.6 tidigare i dokumentet.

4.4.7. ISDN

4.4.7.1. Integrated Services Digital Network

Ett digitalt nät från användare till användare, en tjänst som Telia kallar Telia Duo. Telia Duo har 2 B-kanaler med 64 kbps var, plus 1 D-kanal med 16 kbps. B-kanalerna är för användarens användning medan D-kanalen är till för signalering till nätet. D-kanalen kommer i framtiden att kunna användas för korta meddelanden när de andra kanalerna är upptagna. Telia har även en tjänst för företag, Telia Multi, som har 30 B-kanaler á 64 kbps, och 1 D-kanal på 64 kbps. B-kanalerna kan logiskt kopplas ihop så att en Duo kan överföra data med upp till 128 kbps. Denna hopkoppling sker via drivrutinerna till de program som används för dataöverföringen. Två Duo kan kombineras så att det totalt blir 256 kbps, helt beroende på de ingående programkomponenterna.

4.4.7.2. Anslutning

Anslutning sker på det normala telefonnätet med en speciell ISDN adapter, denna har 3 uttag för terminalanslutningar, varav 1 kan expanderas till 8 uttag. ISDN adaptern kan behöva strömförsörjning beroende på de anslutna terminalutrustningarnas effektförbrukning.

I Duo ingår 2 telefonnummer, ett huvudnummer och ett andranummer, båda kan införas i telefonkatalogen. Duo ger möjlighet till samtrafik med det vanliga telefonnätet både fax, modem och tal, det vill säga att med ett ISDN abonnemang krävs en speciell digital telefon, och med denna telefon kan man tala med personer på vanliga telefoner. Gränssnittet är I.420, men för terminaler med andra gränssnitt som V och X finns det adapterar. På marknaden finns speciella ISDN-kort som sätts in i PC datorer och som utnyttjar Telia Duo tjänsten.

4.4.7.3. Tilläggstjänster

Meddelandeöverföring vid uppkoppling

Meddelandeöverföring under samtalet

Upp till 6 extra telefonnummer

Direktuppringning till ett förbestämt nummer efter valbart antal sekunder efter telefonluren lyfts.

Räckviddsbegränsning som spärrar vissa nummer till exempel 071 nummer

Nummerpresentation av den uppringande partens telefonnummer

Skydd mot det egna numrets presentation hos den som rings upp.

4.5. Sammanfattning

Nivå 3 's stora uppgift är att hitta vägar, routing, genom ett nät så att vilken nod som helst kan kommunicera med vilken annan nod som helst, fastän de inte är direkt sammankopplade. I LAN är alla noder direkt sammankopplade, det är ju definitionen på ett LAN. Visserligen kan dessa vara åtskilda av en eller flera repeater eller brygga, men de anses fortfarande som direkt kopplade då dessa enheter inte påverkar kommunikationen nod

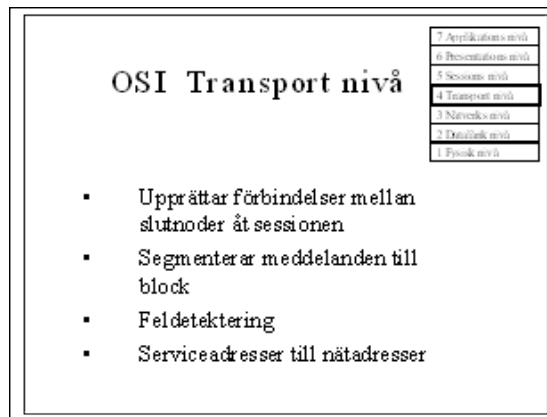
till nod. I LAN har nivå 3 inte mycket att göra mer än att kontrollera om destinationsnoden finns på samma nätverk eller inte. Finns den på samma nätverk så är endast omvandlingen av logisk adress till länkadress det som utföres, och i LAN är den mesta trafiken lokal.

På Internet kan det finnas nätverk som är så stora att de måste delas upp i mindre nätverk och separeras med routrar för att det skall fungera som en autonom enhet. Dessa nätverk blir då egna nät och är sammakopplade med routrar, som vidarebefodrar kommunikationen beroende på var destinationsnoden finns i nätet. Denna typ av router kallas för intern router och har inte samma behov av funktionalitet som de routrar som skall koppla samma autonoma nätverk med varandra. Sammankopplingen av nätverk i Internet har blivit en viktig del i många företags affärsstrategi genom att det ger nya möjligheter för att befrämja affärsidén.

Distinktionen mellan nätverk och nät kan tyckas oväsentlig, men den underlättar förståelsen för de funktioner och problem som är förknippade med nivå 3.

Kapitel 5: Nivå 4, Transport

5.1. Uppgift



Transport nivån ser till att kommunikationen fungerar felfritt mellan slutnoderna som skall utbyta information. Skall se till att felaktig information omsändes, och därigenom garanterar en säker transport av informationen över nätverket för de övre nivåerna. Handhar flödeskontroll för slutnod till slutnod kommunikationen. Delar upp långa meddelanden i segment eller datagram, och ansvarar för att dessa kan sättas ihop till ett meddelande igen. Om paketen ej levereras av protokollen på lägre nivåer så kommer nivå 4 att upptäcka detta.

Omvandlar "humanadresser" till logiska adresser (nätverksadresser). En humanadress är en adress som är lättbegriplig för människor men som inte är användbar för maskiner. Exempel på en humanadress är inom TCP/IP världen, kallas för en domänadress, är www.microsoft.com. Denna adress är lättbegriplig, men måste omvandlas i det här fallet till en IP adress. Denna omvandling sker i denna nivå, omvandlingen från IP adress till länk adress sker sedan i nätverksnivån.

Mellan nivå 4 och nivå 5 är ett speciellt gränssnitt eftersom nivåerna 4 och 3 tillsammans kallas för protokollstacken, och anses som den viktigaste biten av nätverksprotokollen. Ovanför nivå 4 kan flera olika protokoll finnas som alla använder sig av samma eller olika protokollstackar.

5.2. Felupptäckt

Kontrollfunktionen på transportnivån går ut på att alla segment eller paket måste kvitteras, precis som på länknivån. Kvitteringen kan ske med stop & wait, med fast eller "sliding window". Transportnivån kan också dela upp ett meddelande i flera segment, och då måste varje segment identifieras. Genom denna identifiering kommer mottagaren att kunna sätta in segmenten på rätt plats i meddelandet och upptäcka om ett segment saknas. Precis som på länknivån blir även kvitteringen en flödeskontroll så att den sändande noden inte skickar för mycket data som mottagaren inte hinner ta emot. TCP protokollet justerar sitt flöde automatiskt när det upptäcks att mottagaren har svårt att ta emot.

Flera protokoll använder även någon form av kontrollmekanismen och blir då en kontroll på att den data som finns till förfogande är korrekt, se till exempel TCP protokollet i avsnitt 5.3.1. Skillnaden mot länknivån är att kontrollmekanismen fungerar på den data som anlänt till destinationen och jämför med vad som sänts från sändande nod. Länknivån kontrollerar bara länken, som kan vara samma som transportnivån om noderna har direkt

kontakt med varandra.

5.3. Exempel på protokoll

5.3.1. TCP

TCP (Transmission Control Protocol) är ett standardiserat protokoll på Internet, och är ett förbindelseorienterat protokoll som använder IP (Internet Protocol) för nivån under sig. TCP ansvarar för att domänadressen omvandlas till en IP-adress med hjälp av en DNS-server genom att själv initiera ett paket destinerat till denna. I detta paket finns domänadressen och det finns en begäran om att få denna omvandlad till en IP-adress. DNS servern returnerar den begärda adressen i ett paket som svar på förfrågan. Om adressen som TCP fått från nivå 5, redan är en IP-adress används naturligtvis denna. Noden måste vara konfigurerad så att DNS-servern's IP-adress är känd annars kan inte denna omvandling ske från domänadress till logisk adress (IP-adress). Effekten av detta är att man inte kommer vidare ut på nätet såvida man inte använder IP-adresser. Om det inte finns en DNS server tillgänglig kan TCP använda sig av en speciell fil som kan finnas på den lokala noden, och denna fil skall heta hosts. I denna kan den lokala operatören själv ha lagt in omvandlingen av vissa domänadresser till IP-adresser. Filen är en standard text fil som kan editeras med en texteditor, inte en ordbehandlare eftersom den lägger till formateringsinformation som ej kan tolkas av TCP.

Portar används för att hålla reda på vilka protokoll på nivå 5 som använder denna tjänst. Det finns ett antal portar som är fördefinierade och används för vissa protokoll på nivå 5. Telnet och FTP är exempel, där de har port 23 respektive 21.

5.3.1.1. TCP header

TCP Header Format

16-bit						32-bit					
Source Port						Destination Port					
Sequence Number											
Acknowledgement Number (ACK)											
Offset Reserved			U	A	P	R	S	F	Window		
Checksum						Urgent Pointer					
Options and Padding											

Source Port 16 bitar, är en adress till vilket protokoll, eller vilken process, på nivå 5 som har initierat detta paket.

Destination Port 16 bitar, är en adress till vilket protokoll, eller vilken process, på nivå 5 som skall ha detta paket.

Sequence Number 32 bitar, är en pekare till var den första oktetten i detta segment skall vara i meddelandet. I det första segmentet i ett meddelande är detta fält 0, och om det första segmentet innehåller 512 oktetter, så blir detta fält 512 för nästa segment, och om det andra segmentet också är 512 oktetter, så blir det tredje segmentet 1024 i detta fält. Alltså detta fält används till att hålla reda på segmenten som ett meddelande har delats upp i av TCP.

Acknowledgment Number 32 bitar, är ett nummer som kvitterar det sist korrekt mottagna segmentet genom att ange det nästa segment som förväntas komma. Följer samma numrering som Sequence Number fältet, så om det sist korrekt mottagna paketet är det andra, så är detta fält 1024, eftersom det är det nästa segment som förväntas komma efter det andra.

Data Offset 4 bitar, är längden på TCP headern angivet i antal 32 bitars ramar

Reserved 6 bitar, reserverad för framtida behov, måste vara 0 (noll).

Control Bits: 6 bitar,

URG	Urgent Pointer fältet signifikant
ACK	Acknowledgment fältet signifikant
PSH	Push Function
RST	Reset förbindelsen
SYN	Synkronisera sequence numbers
FIN	Inget mer data från sändaren

Window 16 bitar, antalet oktetter i datafältet som sändaren kan acceptera, med start från det som är angivet i acknowledgment fältet.

Checksum 16 bitar, är kontrollinformation på headern + data

Urgent Pointer 16 bitar, innehåller en positiv offset från sequence number fältet och det betyder att den data som finns mellan sequence number och offset inte är del av det ordinarie meddelandet utan är som namnet anger viktig data. Detta fält har giltigt värde endast om URG kontrollbiten är på.

Options variabel längd, och är i multipplar av en oktett, alltså en option är en eller flera oktetter.

Padding variabel längd, används för att headern skall vara i 32 bitars ramar, och är 0 (noll).

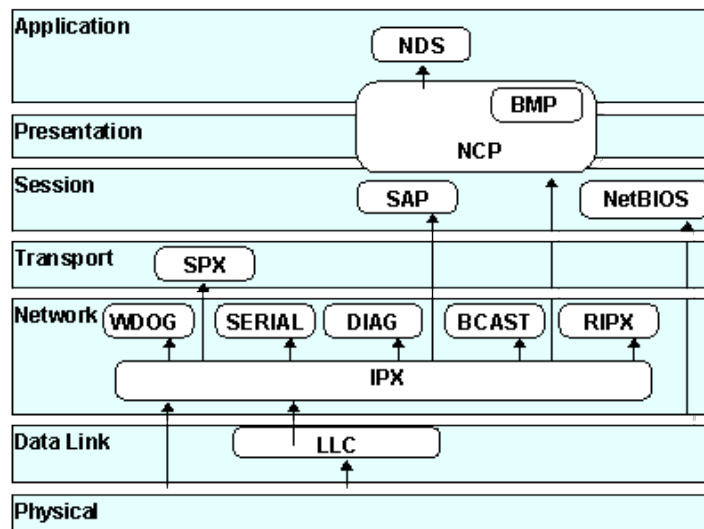
5,3,2, UDP (User Datagram Protocol)

UDP är ett protokoll i TCP/IP protokollsviten, och som inte har någon kontrollfunktion som nivå 4 har. Dessutom kan det inte segmentera ett meddelande, utan används vid de tillfällen när meddelandet kan få plats i ett paket och protokollen på övre nivåer kan hantera eventuella felsändningar.

UDP Header Format

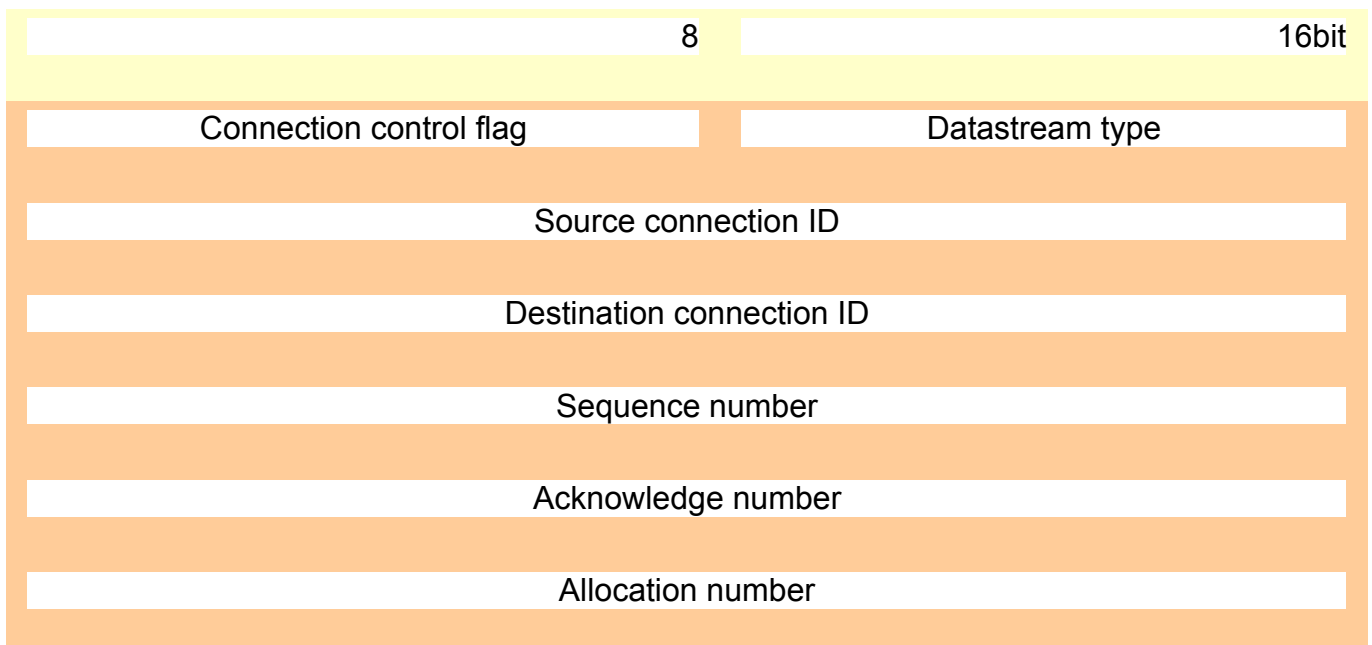
Source Port (16 bits)	Destination Port (16 bits)
Length (16 bits)	Checksum (16 bits)
Data....	

5.3.3. Novell på nivå 4



5.3.4. SPX

Detta är Novell's protokoll på nivå 4 och är ett förbindelseorienterat protokoll



Connection control : Indicates whether the packet is a system or application packet.

Data stream type : Specifies the type of data found in the packet.

Source connection ID : Identification number assigned to the local transport endpoint.

Destination connection ID : Identification number assigned to the remote transport endpoint.

Sequence number : Number of packets exchanged in one direction on the connection.

Acknowledgement number : Sequence number of the next packet SPX expects to receive.

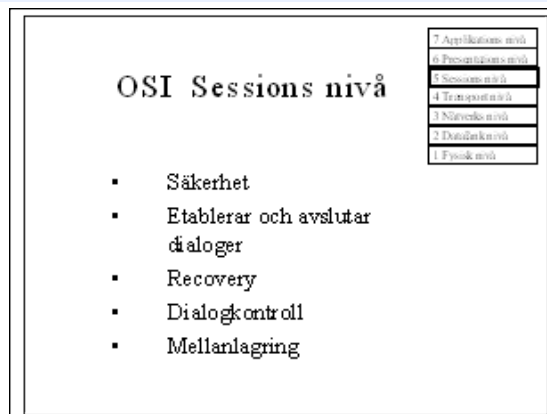
Allocation number : Used to implement flow control between communicating applications. SPX only sends packets until the local sequence number equals the allocation number on the remote machine.

5.4. Sammanfattning

Nivå 4 är den nivå som ansvarar för transporten från sändare till mottagare och därigenom blir den viktigaste länken i kedjan. Men som i alla kedjor måste alla länkar fungera för att hela kedjan skall fungera. Om länknivån eller nätverksnivån ej kan leverera paketen så upptäcks detta av nivå 4. Omsändning av paket är den normala proceduren, men det går inte att sända om paketen hur många gånger som helst. Det finns en gräns när de inte sändes om, denna gräns beror på protokollet, och får till följd att nästa nivå får försöka lösa problemet.

Kapitel 6: Nivå 5, Session

6.1. Uppgift



På sessions nivå etableras kommunikationen logiskt, och hanterar dialogen så att kommunikationen upprätthålles under hela kommunikationsbehovet. Ansvarar även för att avveckla kommunikationen när allt som skall hanteras har behandlats. Sessionen kan ses som ordföranden i ett möte, som skall se till att kalla deltagarna till mötet, ha en agenda över vad som skall tagas upp under mötet, organiserar så att alla får framföra sina punkter av intresse, avslutar mötet när det är färdigt. Skulle mötet avbrytas av någon orsak så skall ordföranden vid ett senare tillfälle kalla till ett nytt möte.

Kalla deltagarna till mötet motsvarar att en applikation eller användare vill etablera kommunikation med en motpart på en annan nod på det egna nätverket eller t.o.m. på ett annat nätverk. I princip kan flera noder delta i en session, (det är inte många applikationer som har sådan funktionalitet), och kontrollen över dessa ingående noder skall då hanteras av nivå 5. Till exempel vilka meddelanden som kommer från vem och att alla får tillgång till samma data. Fördelningen av dialogen mellan de ingående deltagande parterna kan ske på någon av följande sätt:

1 Simplex, till exempel VD på ett företag vill informera alla sina anställda samtidigt men förväntar ingen respons omedelbart på detta.

2 Duplex, till exempel en konferens med deltagare på flera orter och där inte bara röst och/eller bild skall visas utan även bearbetning av data med grupp-programvara.

Sessionen skall säkerställa att allt som har med detta kommunikationstillfälle att göra har sänts och att det har mottagits korrekt. Meddelanden kvitteras så att motparten kan avgöra att dennes kommunikation har fungerat. Om något händer med kommunikationen på någon nivå under, så är det först respektive nivå som skall hantera detta med att begära omsändningar, men om detta misslyckas så rapporteras det uppåt i kedjan och protokoll på högre nivå får avgöra vad som skall ske. Sessionsnivån är den sista som bedömer vad som skall ske om inte nivåerna under (2 och 4) lyckas lösa problemet. Sessionsnivån rapportera i princip till applikationen hur uppdraget har utförts, och denna kan sedan hantera den uppkomna situationen, till exempel genom att meddela användaren.

Vissa implementeringar av datakommunikation använder sessionsnivån som ingång till kommunikationen, till exempel de som använder NetBios. I andra implementeringar ingår sessionsnivån som del i applikationen som till exempel Telnet i TCP/IP sfären av protokoll. Telnet är olyckligtvis namnet på ett protokoll och namn på en applikation som emulerar terminaler över TCP/IP protokollstacken.

I TCP/IP protokollsviten finns det endast denna nivå mellan applikationen och protokollstacken. Med andra ord så de funktioner som OSI-modellen tar upp i nivå 6 och 7 hanteras av de protokoll som finns här eller så hanteras de av applikationerna. I de allra flesta fall så finns funktionerna inbyggda i applikationerna.

6.2. Exempel på protokoll

6.2.1. NetBios

NetBios är egentligen ett gränssnitt för applikationsprogram. Det är först framtaget av IBM men har blivit en defactostandard som många applikationsprogram är skrivna för. NetBios finns i både MicroSofts produkter som i Novell och på Internet. Som namnet anspelar har det för applikationen mot datakommunikation samma uppgift som Bios har för operativsystemet och applikationer vad gäller den lokala hårdvaran i den egna datorn.

6.2.2. Telnet

Telnet är ett terminalemuleringsprotokoll som har tillkommit för att en server inte skall behöva veta all den olika hantering som olika terminaler har, utan ett enhetligt gränssnitt. Det finns applikationer som också heter telnet, men det betyder att de använder Telnet protokollet, och sedan är applikationen användargränssnittet. En server som kan acceptera telnet anslutningar har en serverfunktion som sitter och lyssnar på TCP-port 23, ett nummer som endast är till för telnet. När en klient vill ansluta sig till servern, så anropas den först via denna port, sedan etableras en dynamisk port på server för den fortsatta kommunikationen, och servern återgår till att lyssna på TCP-port 23 för anrop.

Telnet som applikation är normalt ett kommandostyrt gränssnitt och kan startas utan att ange någon fjärrdator. I kommandoläget kan man få veta vilka kommando som finns tillgängliga med kommandot "?". (telnet> är prompten)

```
telnet> ?
```

Commands may be abbreviated. Commands are:

close	close current connection
display	display operating parameters
mode	try to enter line-by-line or character-at-a-time mode
open	connect to a site
quit	exit telnet
send	transmit special characters ('send ?' for more)
set	set operating parameters ('set ?' for more)
status	print status information
toggle	toggle operating parameters ('toggle ?' for more)

- z suspend telnet
- ? print help information

6.2.3. SMTP (Simpel Mail Transfer Protocol)

SMTP är ursprungligen ett protokoll från Unix för elektronsika brev. Unix hade redan från början detta inbyggt som en funktion i operativsystemet. Men det har blivit ett stort protokoll för elektronisk post på Internet, och som namnet säger är det ett ganska enkelt mailprotokoll. Det finns andra nyare och med fler finesser men det används fortfarande väldigt mycket.

6.2.4. POP3 (Post Office Protocol version 3)

POP3 är en modernare variant av SMTP och har fler finesser .

6.2.5. SNMP (Simple Network Management Protocol)

SNMP är ett övervakningsprotokoll, det vill säga att det är detta protokoll som används när övervakningsinformation skall sändas mellan noder. I vissa noder finns det funktioner som samlar på sig data om hur de fungerar och sänder denna information till ett övervakningsprogram på uppmaning. Det kan vara en hub som man kan få reda på hur de olika portarna fungerar, hur mycket data som passerar varje port och så vidare. Det finns en standardiserad "databas" med vilka data som respektive nod behöver samla på sig, och är beskrivet i MIB (Management Information Base)

6.2.6. FTP (File Transfer Protocol)

TP är ett protokoll som används för att flytta eller kopiera filer mellan noder via nätverket. På samma sätt som för telnet finns det en ftpserver som lyssnar på en välkänd port, port 21, och när sedan filöverföringen sker så öppnas en annan port för detta och port 21 är ledig för nya uppdrag.

FTP som applikation är precis som Telnet normalt ett kommandostyrt gränssnitt, och de tillgängliga kommandon listas med kommandot (ftp> är prompten)

```
ftp> ?
```

Commands may be abbreviated. Commands are:

!	cr	macdef	proxy	send
\$	delete	mdelete	sendport	status
account	debug	mdir	put	struct
append	dir	mget	pwd	sunique
ascii	disconnect	mkdir	proxy	tenex

bell	form	mls	quote	trace
binary	get	mode	recv	type
bye	glob	mput	remotehelp	user
case	hash	nmap	rename	verbose
cd	help	ntrans	reset	?
cdup	lcd	open	rmdir	
close	ls	prompt	runique	

6.2.7. DNS (Domain Name Service)

DNS används för att omvandla domänadresser till IP-adresser som sedan användas i kommunikationen mellan noderna. DNS är en distribuerad databas som innehåller information om dessa domäner, bland annat IP-adressen, och denna databas finns på en DNS-server som är en host i nätverket eller på nätet. Förfrågningar till DNS-servern sker med hjälp av DNS-protokollet, liksom svaret tillbaka. För att undvika att frågor skickas till DNS-servern vid varje kommunikation så sparas redan uppslagna adresser i ett cache-minne i respektive nod, för att dels snabba upp sökningar, dels minska nättrafiken.

DNS är uppbyggd i en hierki av servrar, precis som ett filsystem, med en root i toppen. Denna root benäms "." (punkt), och under denna finns DNS-servrarna för COM, MIL, GOV, EDU, NET, ARPA, och respektive lands, som till exempel SE för Sverige. Därunder finns respektive företag eller myndighet som vill använda en egen server. Eftersom det är en distribuerad databas så partitioneras den ut i nätet, och varje partition kallas en zone. Det skall finnas minst två DNS-servrar i varje zone, helst fler, dels för att i händelse av att en av dem skulle falla så finns den andra kvar, dels för att fördela belastningen i nätet på lämpligt sätt.

Domänadressen har en bestämd uppbyggnad för att detta system skall fungera, och följande är definitionen för en domänadress:

```

<domain> ::= <subdomain> | " "
<subdomain> ::= <label> | <subdomain> "." <label>
<label> ::= <letter> [ [ <ldh-str> ] <let-dig> ]
<ldh-str> ::= <let-dig-hyp> | <let-dig-hyp> <ldh-str>
<let-dig-hyp> ::= <let-dig> | "- "
<let-dig> ::= <letter> | <digit>
<letter> ::= A to Z, a to z ( upper and lower case letters a-z)
<digit> ::= 0 – 9

```

Hela domänadressen får inte överstiga 255 tecken, och en label i en subdomän får vara max 63 tecken. Alltså, en domänadress måste börja med en bokstav och avslutas med en bokstav eller siffra, och enda specialtecknet som tillåts är ett bindestreck "-" däremellan. Exempel är www.diadoker.se.

Databasen består av RR (Resource Record) som innehåller följande information:

Owner, domännamnet för denna RR

Type (A | CNAME | HINFO | MX | NS | PTR | SOA), vilken typ av resurs det gäller.

A = *Owner* är en domänadress och *Rdata* är in IP-adress,
 CNAME = det registrerade (kanoniserade) namnet för en alias, det vill säga att *Owner* är aliasnamnet och *Rdata* är det registrerade namnet,
 HINFO = *Rdata* är hostinformation om processor och operativsystem,
 MX = *Rdata* identifierar ett "postkontor" för elektroniska brev,
 NS = *Rdata* är nästa nameserver i hierkin,
 PTR = en pekare till en annan del av domänadressens område,
 SOA = start of authority var denna name server har sitt område, sin zone.

Class (IN | CH), identifierar typ av protokoll

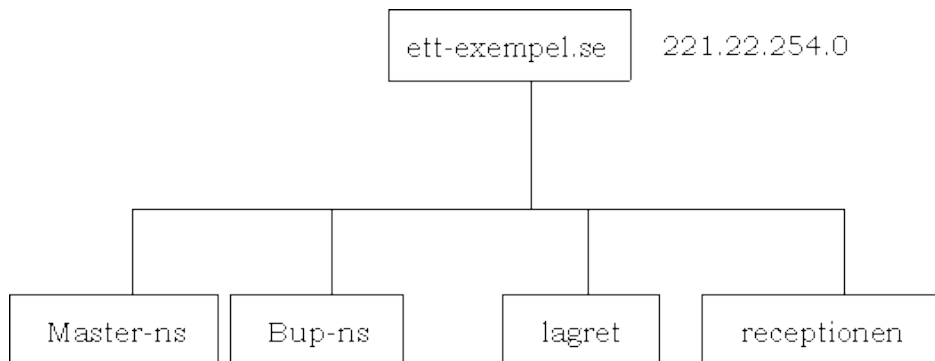
IN = Internet,
 CH = Chaos system

TTL, (Time To Live), hur länge detta RR skall finnas i cacheminnet, detta fält räknas ner efterhand om det inte förnyas.

Rdata, data beroende på type-fältet eller class-fältet. För A är innehållet IP-adressen, för CNAME ett domännamn, för NS ett nodnamn, för PTR ett domännamn, för MX ett 16 bitars värde och ett nodnamn för den nod som skall vara "postkontor" för denna domän, för SOA flera ytterligare värden.

I DNS-databasen för den zone som den har ansvar för skall minst finnas följande RR's, 1 st SOA beskrivande zonen, 1 st NS för varje nameserver som finns för zonen, 1 st A för varje nod i zonen och 1st CNAME för varje alias som finns.

Om vi exemplifierar med en mini-domän med två DNS-servrar och två noder, naturligtvis finns det fler men för det här exemplet räcker det med två. Domänen heter ett-exempel.se, och har IP-adress 221.22.254.0 , DNS-servrarna finns på nodadress 21 (Master-ns, men kallas också för servern) respektive 221 (Bup-ns, men kallas också för Pontus eftersom han sitter på den maskinen), och de två vanliga noderna 12 (receptionen) och 22 (lagret).



Då ser databasen ut på det här viset för denna domän

ett-exempel.se	IN	SOA	ett-exempel.se hostmaster.ett-exempel.se (394876 ; serial 1800 ; refresh every 30 min 300 ; retry every 5 min 604800 ; expire after a week 86400 ; minimum of a day)
		NS	Master-ns.ett-exempel.se

		NS	Bup-ns.ett-exempel.se
Master-ns		A	221.22.254.21
Bup-ns		A	221.22.254.221
receptionen		A	221.22.254.12
lagret		A	221.22.254.22
servern		CNAME	Master-ns
Pontus		CNAME	Bup-ns

6.2.8. NFS (Network File System)

NFS är ett filsystem via nätverket utvecklat av Sun Microsystems, men har blivit en defactostandard. NFS möjliggör att i ett Unix system montera en disk från en nod som finns på en annan maskin än den egna, och därigenom möjliggöra att användaren inte vet var filer finns fysiskt eftersom det är operativsystemet via NFS som gör hämtningen av filerna på den andra noden.

6.2.9. HTTP (Hyper Text Transfer Protocol)

Http används för att sända websidor skrivna i HTML

6.2.10. X-Window

6.2.10.1. Arkitektur

X-Window arkitekturen är uppbyggd på klient - server modellen, och har ett X-protokoll för att överföra fönsterinformation mellan klienten och servern. Servern är en display-server som har faciliteten att visa informationen och hålla reda på användarens inmatning med mus och/eller tangentbord. Klienten är applikationsprogram som skall visa sina resultat i ett fönster eller reagera på användarens inmatning. Denna modell medger att server och klient är skilda åt av ett nätverk, där servern kan vara en PC, Macintosh eller en speciell X-Window terminal. En X-Window terminal är en terminal med en microprocessor, utan disk, programmet är i ROM eller laddas ner via nätverket. Denna konstruktion gör att en X-Window terminal kan vara relativt billig jämfört med en arbetsstation.

6.2.10.2. Display-server

X-displayservern är ett program som håller reda på all inmatning från antingen tangentbordet eller en mus, eller visar information från klientens applikation. Håller reda på vilken klient som har vilket fönster och hanterar kommunikationen mellan klienten och fönstret.

6.2.10.3. Klienter

X-window tillåter att flera klienter körs samtidigt, och kan köras på olika maskiner, men ha samma displayserver. På klienten körs en fönsterhanterare (window manager) som ger karaktär åt X-Windows gränssnitt. Det finns några olika fönsterhanterare på marknaden, t ex Motif från OSF (Open Software Foundation) och Open Look från Sun Microsystem. Idag används X-windows på Linux maskiner för att få en fönsterhanterare, och då blir klient och server på samma maskin. Fönsterhanterare i Linux är t.ex. KDE och Gnome.

6.2.11. SMB (Server Message Block)

Detta protokoll är från mitten av 1980-talet och används i Microsoft Windows för kommunikation mellan klient och server för fildelning och skrivarserver funktioner. Protokollet används fortfarande, men eftersom det ansågs "pratigt" så har version 2 utvecklats av Microsoft. Se närmare beskrivning av protokollet på denna plats : [Översikt av SMB](#), eller om du vill ha en detaljerad beskrivning här : [Detaljerad SMB](#)

6.3. Sammanfattning

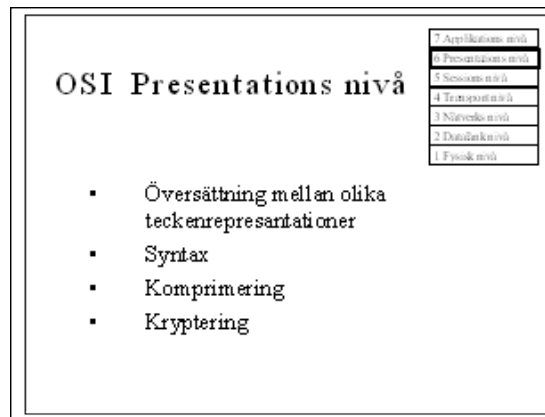
I en TCP/IP protokollstack är sessions nivån den översta, så här finns alla de protokoll som applikationerna kan använda sig av. En del applikationer heter samma som protokollet, som t.ex. telnet eller ftp-klient. Detta kan vara lite förvirrande men man måste skilja på en applikation och ett protokoll, applikationen ligger ovanför protokollstacken, och använder sig av protokollen för att kommunicera. Här listas de protokoll som finns överst i TCP/IP protokollstacken, en del är beskrivna tidigare i kaptilet.

- AFP, Appletalk Filing Protocol
- AIM, AOL Instant Messenger Protocol; för chatt
- APPC, Advanced Program-to-Program Communication
- BitTorrent; för fildelning
- BOOTP, Boot protocol; för konfigurering av IP-adresser
- CFDP, Coherent File Distribution Protocol
- DNS, Domain Name System; för distribution av domännamn
- DHCP, Dynamic Host Configuration Protocol; för konfigurering av nätversanslutningar
- DICOM, Digital Imaging and COmmunications in Medicine; för hantering av medicinska bilder
- FTAM, File Transfer Access and Management
- FTP, File Transfer Protocol; för filöverföring till webbplatser med mera
- Gopher, Gopher protocol
- HTTP, HyperText Transfer Protocol; för World Wide Web med mera
- IMAP, Internet Message Access Protocol; för hämtning av e-post
- IRC, Internet Relay Chat; för chatt
- iTMS, iTunes Music Store Protocol
- LDAP, Lightweight Directory Access Protocol
- Modbus
- MSN Protocol; för chatt

- NIS, Network Information Service
 - NNTP, Network News Transfer Protocol
 - POP3, Post Office Protocol; för hämtning av e-post
 - SIP, Session Initiation Protocol
 - SMTP, Simple Mail Transfer Protocol; för sändning av e-post
 - SNMP Simple Network Management Protocol
 - SSH, Secure Shell
 - TELNET, TELEphone NETwork
 - TFTP, Trivial File Transfer Protocol; för filöverföring till webbplatser med mera
 - TSP, Time Stamp Protocol
 - X.400
 - X.500
 - XMPP, Extensible Messaging and Presence Protoco
 - H.323
-

Kapitel 7: Nivå 6, presentation.

7.1. Uppgift



Presentations nivån är till för att hantera hur information skall presenteras för applikationen så att denna kan förstå vad det är. T.ex om en engelsman och en fransman skall tala med varandra måste de komma överens på vilket språk de skall kommunicera på för att de skall förstå varandra. I datavärlden kan noder som kommunicerar med ASCII och EBCDIC fortfarande kommunicera genom att den ena noden omvandlar sin data så kan den andra förstå den.

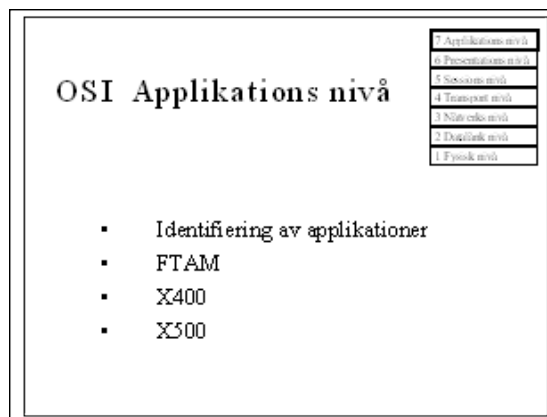
Men här bestäms även om komprimering skall användas, vilken algoritm som skall användas, om informationen skall krypteras och hur detta skall göras. Komprimering och kryptering måste använda någon metod som måste vara känd på båda slutnoderna, och detta kan förhandlas fram vid etableringen av kommunikationen. Kryptering kan ske med symmetrisk teknik som använder privata nycklar, eller asymmetrisk som använder publika nycklar. Symmetrisk kryptering med privata nycklar innebär att båda parter måste inneha var sin kopia av krypteringsnyckeln annars kan meddelandet inte deshifreras. Det är problem med att hålla dessa privata nycklar hemliga, men även hur de skall distribueras så att de inte kommer i orätta händer. Därför har den asymmetriska tekniken tagit över.

Den asymmetriska krypteringen använder två par av nycklar, en publik och en privat. Dessa nycklar har ett sådant inbördes förhållande att om något krypteras med den ena nyckeln kan det deshifreras med den andra och vice versa. De ingående parterna som skall kommunicera utbyter publika nycklar och när någon skall sända ett meddelande så krypteras detta med motpartens publika nyckel. Detta krypterade meddelande kan endast deshifreras med nyckeln som är i par med den publika, och det är den privata nyckeln som ju mottagaren har i tryggt förvar hos sig själv. Den sändande kan också kryptera meddelandet med sin privata nyckel, vilket medför att alla som har tillgång till den publika nyckeln kan se detta meddelande. Detta sätt kan användas när man vill skicka meddelanden till en viss grupp men inga utomstående skall kunna läsa det, då skapar man ett nyckelpar som används vid kommunikationen med denna grupp, och distribuerar den publika nyckeln till gruppen.

Kryptering kan ske på nivå 1 också, men då användes speciell hårdvara som kopplas mellan den sändande parten och nätverket, så att all trafik som går ut på nätet krypteras. Då används den symmetriska tekniken, eftersom det måste finnas en motsvarande "burk" hårdvara hos mottagaren som deshifrerar den mottagna datan. När kryptering används på nivå 6 så kan enstaka meddelanden krypteras och inte nödvändigtvis alla meddelanden.

Kapitel 8: Nivå 7, applikation.

8.1. Uppgift



Applikations nivån är starten på OSI stacken från programmets och användarnas synpunkt. Denna nivå håller reda på vilka applikationer som kommunicerar med vem, och vilken information som hör till respektive applikation.

Applikationsnivån gör olika servicetjänster tillgängliga för utnyttjande. Exempel på servicetjänster/protokoll som finns här är FTAM (File Transfer Access Methods), X400 som är ett MHS och X500 som är ett katalogsystem för bl.a. X400. Dessa är OSI definierade tjänster, men det finns andra som i ett Novell nätverk där vi har till exempel fileserver, printserver, dessa är tjänster som nätverket erbjuder. Dessa servicetjänster har service adresser, och i vissa protokoll kallas de ports eller sockets.

MHS betyder Message Handling System, ett meddelandehanterings system för forward&store meddelanden som till exempel elektronisk post.

När datakommunikationstjänster skall utnyttjas av en applikation så används nivå 7 som gränssnitt, och för applikationen är det som att hantera filer. En rutin används, som till exempel READ, och till denna adderas parametrar som anger vad som rutinen skall arbeta med. Vid alla sådana här gränssnitt vill applikationen få ett resultat av sin begärda tjänst, och detta fås som den begärda data eller ett meddelande som anger varför det inte finns någon data. Detta meddelande skall ses som en avrapportering av hur tjänsten utfördes, och det är via detta meddelande som applikationen får reda på om det har blivit något fel vid överföringen.

Som vi har sett tidigare finns det tre nivåer som ansvarar för att kontrollera överföringen, nivå 2, 4 och 5, och de utför denna uppgift från olika perspektiv. Men om dessa tre nivåer inte kan få det att fungera trots de metoder som står dem tillbuds, så meddelas resultatet uppåt till nivån ovanför, och eventuellt får applikationen reda på det. Hur applikationen skall hantera denna uppkomna situation, alltså att det inte gick att kommunicera som tänkt, beror på hur applikationen är gjord och ligger helt utanför OSI-modellens sfär av intresse.

Här på denna länk kan du få mer information om applikationsnivån och dess protokoll: http://en.wikipedia.org/wiki/Category:Application_layer_protocols

Kapitel 9: Design av kommunikation

9.1. Bedöma kommunikationsbehoven

För att designa datakommunikation finns det många faktorer att ta hänsyn till. Här är en sammanställning som kan vara intressant att ta del av, men detta skall ses som en hjälp att reda ut behoven, inte som en karta eller mall över hur man gör när man designar ett nätverk.

Som i de flesta designprocedurer måste först en analys göras på de behov som redan finns och det som redan existerar. Sedan försöka blicka in i framtiden och se vad man vill åstadkomma på längre sikt. I de teknikorienterade områdena förändras situationen snabbt, och då kan man med fel teknik ha "låst in sig i hörnet". Därmed inte sagt att man skall ha en rädsla för att göra något. Om man väntar lite för att få del av den nya tekniken, så sitter man där och väntar och väntar, eftersom det med all sannolikhet kommer bättre teknik, och den blir också billigare hela tiden.

9.1.1. Faktorer

9.1.1.1. Inter/intra-organisatorisk datakommunikation

Inom en organisation kan lösningar väljas fritt (normalt anpassning till redan gjorda val)

Vid kommunikation med andra organisationer måste anpassning ske (gränssnitt/protokoll på samtliga OSI nivåer)

9.1.1.2. Intern/Extern datakom

Vid extern kommunikation (till exempel databassökningar, geografiskt spridda företag, användning av portabla PC för resande) krävs portar/gateways mot allmänna nät

9.1.1.3. Trafik

Bedömning av volymer: antal transaktioner / tid * tecken / transaktion, dag och säsongsvariationer

Trafiktyper: satsvis / interaktiv kommunikation, transaktion- / fil- / bildöverföring

Önskade eller begärda överföringstider och svarstider

Identifiera flaskhalsar

9.1.1.4. Tillgänglighet / säkerhet

Krav på :

dubblering av linjer, noder

nätövervakning

val av nättyp

9.1.1.5. Sekretess

Krav på :

kryptering, sigill

identifiering (motringning, lösenord)

speciella nättjänster / nättyper

9.1.1.6. Användarvänlighet

generella funktioner för nätanvändning (adressering, inloggning, identifiering)

terminalkonverterings program

9.1.1.7. Drift och underhåll

enkel installation

utrymmessnålhet

okänslighet mot elektriska störningar

fysiskt skyddade kablar

enkla anslutningsmöjligheter

hög kapacitet

9.1.1.8. Kostnads / intäcksanalys

Hur kostnader skall fördelas på olika tjänster

9.2. Logisk design av nät

9.2.1. Indelning i delsystem

9.2.1.1. Geografisk indelning

Lokalt begränsad datakom, (punkt - punkt, enkla lan, PC-lan, korthållsmodem)

inom en byggnad

inom ett rum

närliggande byggnader

Datakom inom ett företagsområde, (växlar, korthållsmodem, lan stamnät)

en stad

mellan byggnader
mellan våningar

Lång distans datakom, (allmänna nät)

geografiskt spridda företag
andra organisationer

Internationell trafik, (flera allmänna nät, fasta linjer)

9.2.1.2. Trafikmässig indelning

Lokal lösning för de som kommunicerar mest, ofta alla som arbetar med samma applikation

Olika delnät kan skiljas med bryggor.

Kommunikation med datorer eller med servers.

9.2.1.3. Indelning efter trafikmiljö

Blanda inte synkron och asynkron trafik, en leverantörs nät med en annan leverantörs.

Förena via konverteringsutrustning

9.2.2. Välj eventuell växelfunktion

Växelfunktion behövs om en terminalenhet (terminal, PC, dator) skall kommunicera med eller använda tjänster hos minst två andra enheter eller servers.

Exempel på växelfunktioner:
omkopplare, kopplingstavlor = manuellt,
terminalväxlar,
PABX,
dator i ett nät,
nät (lan, wan)

9.2.3. "Optimera" kopplingar

Minskad ledningskostnad genom användning av till exempel:

multiplexorer
koncentratorer
linjedelare
multidrop / slingnät

9.2.4. Välj kopplingar

9.2.4.1. Kablage

Beror av krav på kapacitet (volym), överföringstider, säkerhet, sekretess

9.2.4.2. Konverteringsutrustning

Beror på avstånd, hastighetskrav, trafikform (asynkron/synkron, duplex)

9.2.5. Inför eventuella konverteringsfunktioner

Nätets alla delar som skall kommunicera (terminaler, datorer, servers, växlar...) måste passa ihop på relevanta ISO-nivåer. Om olika protokoll/gränssnitt används så krävs konverteringsprogram/utrustning, till exempel:

kommunikationskort och program i PC
synkron/asynkron omvandlare
termnalemulering
PAD

Kapitel 10: Referenser

William Stallings, 1997, *Data and computer communications*, Prentice Hall, Fifth edition, ISBN 0-13-571274-2

J.Postel, 1980, *RFC 0768 User Datagram Protocol*, IETF Standard #6

J.Postel, 1981, *RFC 0791 Internet Protocol*, IETF Standard #5

J.Postel, 1981, *RFC 0792 Internet Control Message Protocol*, IETF Standard #5

J.Postel, 1981, *RFC 0793 Transmission Control Protocol*, IETF Standard #7

J.Postel, 1982, *RFC 0821 Simple Mail Transfer Protocol*, IETF Standard #10

J.Postel, J.K.Reynolds, 1983, *RFC 0854 Telnet Protocol specification*, IETF Standard #8

J.Postel, J.K.Reynolds, 1983, *RFC 0855 Telnet option specification*, IETF Standard #8

J.Postel, J.K.Reynolds, 1985, *RFC 0959 File Transfer Protocol*, IETF Standard #9

P.V.Mockapetris, 1987, *RFC 1034 Domain names – concepts and facilities*, IETF Standard #13

P.V.Mockapetris, 1987, *RFC 1035 Domain names – implementation and specification*, IETF Standard #13

J.D.Case, M.Fedor, M.L.Schoffstall, C.Davin, 1990, *RFC 1157 Simple Network Management Protocol*, IETF Standard #15

W.Simpson, 1994, *RFC 1661 The Point-to-Point Protocol*, IETF Standard #51

W.Simpson, 1994, *RFC 1662 PPP in HDLC – like framing*, IETF Standard #51