Teoria Reti di Calcolatori

Cos’è internet? Internet è una rete di calcolatori che connette miliardi di dispositivi(host) in tutto il mondo. Gli host sono connessi tra loro mediante una rete di collegamenti(cavi coassiali, fibre ottiche..). Ogni collegamento si occupa di trasferire un pacchetto da un mittente ad un preciso destinatario passando per i commutatori di pacchetto(router e switch) i quali prendono un pacchetto dal collegamento in entrata e lo ritrasmettono su uno di quelli in uscita(in base al destinatario).

ISP(Internet Service Provider): Gli host accedono ad internet tramite i cosidetti ISP. Un ISP è un insieme di commutatori di pacchetto e collegamenti che forniscono all’host una rete per comunicare con gli altri.

Tipi di trasmissione(RICCOBENE)
Punto a Punto: collegamento più semplice, due macchine collegate da un solo cavo direttamente. Può essere full-duplex nel caso in cui entrambi gli host possono parlare contemporaneamente, half-duplex se possono scambiarsi messaggi uno alla volta(ricetrasmittente) oppure unidirezionale se la comunicazione è unidirezionale.
Broadcast: tutti possono ascoltare e parlare contemporaneamente. Al fine di non sovrapporre i segnali bisogna fornire delle regole di comunicazione.
Bus Condiviso: tutti possono leggere tutti possono scrivere. L’informazione si propaga dalla macchina generata a tutte le altre.
Anello: l’informazione viene immessa nel circuito e salta da una macchina all’altra fino a quando non ritorna in quella mittente.
A stella: consiste di un hub centrale collegato punto a punto ad altre macchine. Quando una macchina invia un segnale, l’hub lo ritrasmette a tutte le altre. Si dice che è stupido in quanto non converte i segnali in informazione ma li ritrasmette semplicemente. Non potendo convertire in bit se riceve due segnali contemporaneamente, ne ritrasmette la collisione.

Trasmissione store and forward: La maggior parte dei commutatori di pacchetto usa questo tipo di trasmissione che consiste nel fatto che il commutatore deve ricevere l’intero pacchetto prima di poterlo cominciare a trasmettere in uscita. A tal fine il commutatore mantiene un buffer di output per conservare i pacchetti. Una volta ricevuto dal buffer il pacchetto si mette in coda prima di essere rispedito(ritardo di accodamento). Se la coda supera la grandezza del buffer si verificherà una perdita di pacchetto.

Commutazione di Circuito
In queste reti le risorse richieste lungo il percorso(buffer e velocità di trasmissione(banda)) per consentire la comunicazione, sono riservate per l'intera durata della sessione di comunicazione. Un esempio sono le linee telefoniche. Quando due host desiderano comunicare la rete stabilisce una connessione end-to-end(punto a punto) dedicata a loro con una certa banda garantita.
Ex. Abbiamo 4 commutatori connessi tramite 4 collegamenti con una velocità di trasmissione massima pari a 1Mbps. Questo significa che ogni collegamento avrà una banda pari a 250Kbps garantita. Se due di questi commutatori vogliono comunicare allora, nonostante gli altri collegamenti non siano usati, potranno al più scambiarsi dati a 250Kbps.

Commutazione a livello di pacchetto
La differenza con la commutazione di circuito sta nel fatto che in questo modello, le risorse riservate alla comunicazione vengono stanziate solo nel momento in cui si vuole stabilire una comunicazione. Nell'esempio precedente, se due commutatori vogliono comunicare e gli altri collegamenti sono liberi, allora la connessione avrebbe un'ampiezza di banda pari a 1MBps: ovvero viene sfruttata tutta la banda disponibile. Dal punto di vista delle risorse la commutazione di circuito è molto più dispendiosa in quanto non è detto che tutti e 4 i collegamenti vengano usati(non è detto che tutti gli utenti accedono ad internet) e quindi risulta inutile riservare banda a connessioni inutilizzate.

Gitter o Jitter come minchia si scrive: quantità di tempo che trascorre tra la trasmissione del pacchetto n e quella del pacchetto n+1

Tipi di Ritardo

Ritardo di elaborazione
Tempo richiesto per esaminare l'intestazione del pacchetto e per determinare dove dirigerlo. A ciò si aggiunge anche il tempo richiesto per controllare eventuali errori. Ordine di microsecondi.

Ritardo di accodamento
Dipende dal numero di pacchetti arrivati precedentemente ed accodati in attesa di ritrasmissione sullo stesso collegamento. Se la coda è vuota il ritardo è nullo.

Ritardo di trasmissione
Tempo richiesto per trasmettere tutti i bit del pacchetto sul collegamento.

Ritardo di propagazione
Una volta immesso nel collegamento il pacchetto deve propagarsi fino ad un router(o fino al destinatario). Il ritardo di propagazione è dato dal rapporto fra la distanza che

intercorre fra mittente e destinatario e la velocità di propagazione del collegamento(propria del mezzo fisico).

Ritardo ent-to-end
Somma dei ritardi complessivi che si verificano su ogni nodo del collegamento.

Protocollo: definisce il formato e l'ordine dei messaggi scambiati tra due o più entità di comunicazione così come le azioni intraprese in fase di trasmissione e di ricezione di un messaggio o di un altro evento. Definisce le regole della comunicazione. I progettisti organizzano i protocolli e l'hardware e software che li implementano in livelli o strati. Ciascun protocollo appartiene ad uno di essi.
Ogni livello fornisce il suo servizio utilizzando i servizi del livello immediatamente inferiore(e quindi per transitività usa tutti quelli sottostanti). Considerati insieme i protocolli dei vari livelli costituiscono la pila dei protocolli(approccio top-down).

Pila di protocolli di internet→ Modello TCP/IP

Livello di applicazione: è la sede delle applicazioni di rete e dei relativi protocolli(HTTP,DNS,FTP). Un'applicazione in un host tramite il protocollo scambia pacchetti(messaggi) con l'applicazione di un altro host.
Livello di trasporto: trasferisce i messaggi del livello di applicazione. In internet troviamo TCP e UDP. In questo contesto i messaggi prendono il nome di segmenti.
Livello di rete: si occupa di trasferire i pacchetti a livello di rete, detti datagrammi, da un host ad un altro attraverso una serie di router tra sorgente e destinazione. Il protocollo a livello di rete di internet è IP che comprende anche diversi protocolli adibiti all'instradamento.
Livello di collegamento(Data Link): per trasferire un pacchetto da un nodo a quello successivo sul percorso il livello di rete si affida ai servizi del livello di collegamento. In particolare ad ogni nodo il ivello di rete passa il datagramma al livello sottostante che lo trasporta al nodo successivo. Il pacchetto a livello di collegamento è detto frame.
Livello fisico: il suo ruolo è quello di trasferire i singoli bit del frame da un nodo a quello successivo.

Modello OSI
L'ISO(International Organization for Standardization) propose che le reti di calcolatori fossero organizzate in sette livelli. Il modello OSI comprende i livelli sopracitati con l'aggiunta del livello di presentazione e di sessione(posti fra il livello di applicazione e quello di trasporto). Il livello di presentazione fornisce servizi che consentono ad applicazioni di interpretare il significato dei dati scambiati(cifratura dei dati), mentre quello di sessione fornisce la sincronizzazione dello scambio di dati. In particolare il livello di sessione, in caso di interruzione forzata della connessione, permette di riprendere da dove si aveva lasciato.
Il modello OSI è solo TEORICO.

Architettura Client-Server
Consiste di un host sempre attivo(SERVER) che risponde alle richieste da parte di altri host(client). Il client non ha bisogno di essere sempre attivo: potrebbe inviare una richiesta al server e poi spegnersi. Per le grandi aziende un singolo host server non è in grado di rispondere a tutte le richieste dei vari client. Per tale motivo ad oggi esistono dei complessi chiamati data center che ospitano diversi host server.

Architettura P2P(Peer to Peer)
In questo tipo di architettura si sfrutta la comunicazione diretta tra coppie arbitrarie di host. Ogni host(peer) comunica direttamente con un altro suo pari(peer) senza che i messaggi passino attraverso dei server intermedi. Quest'architettura viene spesso utilizzata per applicazioni che si occupano di messaggistica istantanea, videoconferenza, telefonia e trasferimento file.
A differenza dei data center questo tipo di architettura non ha bisogno di grandi infrastrutture(soldi) e di manutenzione, inoltre ogni peer che richiede file, aggiunge a sua volta capacità di servizio al sistema, rispondendo alle richieste di altri peer.

Comunicazione tra Processi
Un processo è un'istanza di un programma in esecuzione su una macchina. La comunicazione fra processi nello stesso calcolatore avviene mediante il SO. Quando i due processi sono situati su macchine diverse allora essi devono comunicare scambiandosi messaggi attraverso la rete.

Client-Server
Per ciascuna coppia di processi comunicanti, generalmente etichettiamo come client quell'host che richiede un servizio, mentre indichiamo come server colui che fornisce il servizio richiesto.

Socket
Una socket è un'interfaccia software che permette di collegare il livello applicativo con quello di trasporto. Essa rappresenta una porta mediante la quale il processo applicativo spedisce i pacchetti al livello di trasporto.
Indirizzamento
Per spedire un pacchetto da un host ad un altro è necessario conoscere due dati fondamentali: l'indirizzo IP dell'host destinatario cui il pacchetto deve essere spedito, ed il numero di porta di destinazione che specifica la socket(e quindi il processo) che dovrà ricevere il dato. Alle applicazioni più note sono stati assegnati dei numeri di porta specifici. Se ad esempio viene spedito un pacchetto con numero di porta 80 allora siamo certi che tale pacchetto è destinato ad un processo web server.

Proprietà dei protocolli di trasporto
Trasferimento dati affidabile: un protocollo offre un trasferimento dati affidabile se i dati inviati sono consegnati corretti e completi. Ovvero noi abbiamo l'assoluta certezza che quei dati arriveranno senza errori al processo ricevente. Loss-to-tolerant sono quelle applicazioni che tollerano delle perdite(multimediali). Esistono inoltre

delle applicazioni che, in caso di perdite, utilizzano algoritmi di interpolazione per recuperare i dati mancanti.
Throughput: è il tasso al quale il processo mittente può spedire i bit a quello ricevente. In soldoni è la larghezza della banda. Garantire un throughput per una connessione è un qualcosa di molto difficile in quanto dipendente dal traffico in rete.
Temporizzazione: avere la certezza che i dati arriveranno entro un determinato lasso di tempo. Questa proprietà interessa particolarmente le applicazioni interattive.
Sicurezza: fornire riservatezza tra i due processi comunicanti impedendo che host terzi possano leggere i dati scambiati.

HTTP(Hypertext trasfer protocol)→ protocollo di pull ovvero di richiesta di dati.
HTTP è il protocollo a livello di applicazione del Web. Definisce il modo in cui i client web richiedono le pagine ai web server e come questi ultimi le trasferiscono ai client. Il client web è rappresentato dal browser che l'utente usa mentre il web server rappresenta un host che ospita oggetti web indirizzabili tramite URL(Apache e Microsoft Internet Information Server).
HTTP utilizza TCP ed ha numero di porta 80.
Descrizione della comunicazione:

1. Il processo client HTTP,in esecuzione sull'host client, inizializza una connessione TCP con il processo server HTTP in esecuzione sull'host server.
2. Una volta instaurata la connessione, il client HTTP tramite la propria socket invia un messaggio HTTP che include l'URL e quindi il percorso per l'oggetto richiesto.
3. Il processo server HTTP riceve il messaggio, recupera l'oggetto richiesto, lo incapsula in un pacchetto e lo spedisce al client
4. Il server comunica a TCP di chiudere la connessione cosa che avverrà solo quando il client avrà ricevuto il messaggio
5. Il client riceve il messaggio e la connessione termina.

Connessioni Persistenti e non persistenti:
Nelle connessioni persistenti(HTTP 1.1) il server web lascia la connessione aperta dopo aver inviato la risposta al client. In tal modo il server può inviare più oggetti(e quindi pagine web) alla volta durante la stessa connessione.
Nelle connessioni non persistenti invece il server chiude la connessione dopo aver inviato l'oggetto richiesto. Se la pagina web richiesta consiste di 10 oggetti allora saranno necessarie 10 connessioni TCP

Messaggi HTTP

I messaggi HTTP sono scritti in testo ASCII . Quelli di richiesta consistono di cinque righe ciascuna seguita da un carattere di ritorno a capo ed un carattere di nuova linea. La prima riga è detta di richiesta mentre quelle successive sono di intestazione. Nella riga di richiesta ci sono tre campi: il campo metodo che può assumere diversi valori(PUT,GET,DELETE), il campo URL ed il campo versione di HTTP.

Le righe di intestazione comprendono la riga host che specifica l'host in cui risiede l'oggetto(server), la linea Connection che indica se la connessione è persistente o meno, la linea User-agent che indica il browser che sta effettuando la richiesta, infine la linea Accept-language indica la lingua in cui si preferisce che l'oggetto sia spedito. Dopo le linee di intestazione si trova un corpo che può essere riempito nel caso in cui l'utente sta cercando dei contenuti specifici all'interno della pagina web richiesta.
Messaggio di Risposta
La prima riga è quella di stato e presenta tre campi: la versione del protocollo, un codice di stato ed un corrispettivo messaggio di stato.
La riga Connection indica che il server ha intenzione di chiudere la connessione dopo l'invio del messaggio. La riga Date indica l'ora e la data di creazione e invio della risposta. La riga Server indica da quale server è stato generato il messaggio. La riga Last-Modified indica l'istante e la data in cui l'oggetto è stato creato o modificato l'ultima volta. La riga Content-Type indica il formato dell'oggetto. Infine abbiamo un corpo il quale contiene l'oggetto richiesto.
Il codice di stato ed il relativo messaggio indicano il risultato che ha avuto la richiesta: ex. 404 Not Found.

DHTML
Estensione di HTTP in cui il web server al momento della richiesta costruisce la pagina richiesta direttamente online e la spedisce al client.

FTP(File Transfer Protocol)→ TCP
Si tratta di un protocollo per il trasferimento di file tra due host remoti. Il protocollo instaura tra i due host due connessioni, sulla porta 20 e 21, una mediante la quale scambiano i dati(porta 20)ed un altra adibita allo scambio di messaggi(porta 21)al fine di monitorare lo stato della connessione(ad esempio se mantenerla o meno). Tale protocollo si differenzia da HTTP in quanto mantiene uno stato ovvero delle informazioni sui due host: prima di poter cominciare a strasmettere i dati i due host devono autenticarsi. Per sopperire a questa mancanza HTTP ha implementato l'utilizzo dei cookie. I cookie permettono ai web-server di tener traccia degli utenti al fine di fornire dei contenuti in funzione delle loro ricerche più frequenti. Non sono altro che dei codici identificativi.

Web Cashing
Una web-cache è un proxy-server ovvero un'entità che soddisfa richieste HTTP al posto del web-server effettivo. Il proxy ha una propria memoria in cui conserva copie di oggetti richiesti recentemente. Quando un client effettua una richiesta esse vengono dirette innanzitutto verso il proxy server, se quest'ultimo ha in cache l'oggetto richiesto allora egli lo invia al client, altrimenti effettua una richiesta al server e ritrasmette la risposta del server al client, conservando in cache una copia dell'oggetto richiesto(agisce sia da client che da server). I proxy server hanno una grande importanza perché permettono di non sovraccaricare i server ed allo stesso tempo di fornire delle risposte più veloci.

SMTP(Simple mail transfer protocol) → protocollo di push in quanto invio messaggi
SMTP rappresenta il principale protocollo per la posta elettronica a livello applicativo. Esso è basato su TCP al fine di garantire un trasferimento dati affidabile. Oltre che dal protocollo applicativo il sistema postale è costituito dallo user-agent(Outlook, gmail..) e dal server di posta.
Comunicazione: siano due host A e B e supponiamo che A voglia spedire una mail all'host B:

1. A invoca il proprio user agent, fornisce l'indirizzo di posta di B, scrive il messaggio e lo invia.
2. Lo user agent di A invierà il messaggio al suo mail server.
3. Il lato client di SMTP eseguito sul mail server di A vede il messaggio e apre una connessione TCP verso il lato server SMTP in esecuzione sul mail server di B ed invia il messaggio.
4. Il lato server di SMTP riceve il messaggio e lo ripone nella casella di posta di B il quale quando lo ritiene opportuno, invoca il proprio user agent per leggere il messaggio.

Messaggi SMTP
Dopo una breve presentazione(handshake) in cui il client indica l'indirizzo di posta del destinatario, quest'ultimo invia i dati che costituiscono la mail più un messaggio terminale che consiste solo di un punto. Successivamente sempre il client invia un comando QUIT che chiude la conversazione. Ad ogni comando del client il server risponde con un codice ed una qualche spiegazione in inglese.

POP3 e IMAP
Sono due protocolli usati dagli host per trasferire i messaggi dal web server al PC locale. SMTP infatti essendo di push trasferisce il messaggio dallo user agent del mittente al suo mail server e successivamente fra i mail server mentre non può essere usato per prelevare messaggi dal mail server.
POP3 è un protocollo piuttosto semplice che permette di prelevare dal server le mail e scaricarle sul proprio PC locale. Ciò risulta essere molto limitante soprattutto nel caso in cui noi volessimo nuovamente accedere alle mail però da un altro dispositivo. A tal fine si è introdotto IMAP il quale ci permette di salvare la posta sul mail server e conservarne una copia in locale, offrendo dunque la possibilità di recuperarla successivamente anche tramite altri dispositivi.

DNS → UDP porta 53
Quando andiamo ad effettuare una richiesta ad un web server, usiamo dei nomi come www.amazon.it oppure www.google.com. In realtà, nonostante questi siano i nomi degli host, essi forniscono ben poche informazioni sulla loro collocazione in Internet. Al fine di ottenere queste informazioni è necessario tradurre questi nomi nei corrispettivi indirizzi IP: codice di 4 byte (in cui ogni byte rappresenta un numero che va da 0 a 255) che permette di identificare in modo univoco un dispositivo in rete. Ad ogni host è assegnato uno o più indirizzi IP tramite cui ottenere informazioni sulla loro collocazione in rete. DNS rappresenta sia un database distribuito costituito da

centinaia di server contenenti indirizzi IP, e sia un protocollo che permette agli host di interrogare questi server per ottenere le traduzioni.

DNS aggiunge ulteriore ritardo all'accesso in rete infatti supponendo che un host A voglia contattare il web server www.amazon.it allora:

1. Il client HTTP di A prima di stabilire una connessione TCP deve conoscere l'indirizzo IP del web server amazon
2. A tal fine invia una richiesta al DNS client per tradurre l'URL usato dall'host
3. Il client DNS invia un'interrogazione ad un server DNS
4. Ricevuta la risposta il client DNS comunica l'indirizzo IP al client HTTP
5. Il client HTTP stabilisce una connessione con il web server.

Struttura DNS

Se il DNS fosse costituito da un unico server si incorrerebbe in diversi problemi:

1. Un solo database dovrebbe far fronte a milioni di richieste al giorno
2. Essendo un solo database ad avere tutte le informazioni, se subisse un guasto internet collasserebbe e nessuno riuscirebbe a navigare in rete
3. Un unico database centralizzato risulterebbe vicino a certe regioni, ma lontano rispetto al resto del mondo→ questo provocherebbe dei tempi di accesso alle informazioni lunghissimi.
4. Problemi di manutenzione

Considerando questi aspetti negativi quindi si è optato per un database distribuito e gerarchico. Distinguiamo tre tipi di DNS server:

1. Root server: rappresentano il livello più alto nella gerarchia, e si occupano di fornire gli indirizzi IP dei server top-level domain .
2. Server top-level domain(TLD): si occupano dei domini come org, com, edu, gov. I server TLD forniscono gli indirizzi IP dei server autoritativi.
3. DNS server autoritativi: ogni azienda od organizzazione dotata di host pubblicamente accessibili in internet(web server) deve fornire i propri indirizzi IP i quali vengono ospitati proprio dai server DNS autoritativi dell'azienda.

Un ultima categoria riguarda i DNS server locali. Questi sono associati a ciascun ISP come università, dipartimenti, aziende, ISP residenziali. Questi server locali sono particolarmente importanti in quanto fanno da proxy fra l'host e la gerarchia dei DNS server. In DNS ha grande importanza il servizio di caching: ogni server DNS mantiene in cache gli indirizzi IP delle ricerche recenti. In tal modo se un host richiede un IP chiesto poco tempo prima, il server può comunicarglielo direttamente senza che l'host invii richieste ad altri DNS server. Grazie a questo servizio il DNS può essere visto più che come un albero, come un grafo: ogni DNS server cui effettuiamo una richiesta di traduzione può avere già la risposta in cache, altrimenti ci indirizzerà verso il DNS server che può farcela trovare.

NON È NECESSARIO PASSARE DAI DNS SERVER ROOT.

Il servizio di cache quindi ci permette di diminuire il numero di richieste da effettuare ed al contempo velocizzare il processo di traduzione. Non solo, ma quando ricerchiamo un traduzione i DNS server ci fornisce non uno ma una lista di indirizzi dalla quale scegliere. Tale lista, ad ogni richiesta, viene modificata al fine di non fornire ad ogni richiesta la stessa traduzione: in questo modo si evita la congestione

di un solo(ad esempio) web server ma si “spalma” su tutti quelli disponibili(pensiamo ad esempio ad i server di google).
Importante: gli indirizzi IP variano col tempo quindi il database ha bisogno di aggiornarsi.

RICCOBENE E LA SUA VOGLIA DI SPIARE GLI ALTRI
SNMP: protocollo per il monitoraggio (SIMPLE NETWORK MANAGMENT PROTOCOL)→ UDP
Voglio monitorare un server, un dispositivo, bene si usa questo. Non solo monitoraggio ma anche azione sui dispositivi. Devo avere un agente che una volta installato può chiedere tutto al sistema operativo(le info di sistema) le chiede e le mette in un suo database chiamato MIB. Le info possono essere chieste da un manager esterno tramite SNMP. Il manager può interrogare ma anche effettuare operazioni di set(cambiare un parametro ->pericolo) Nella versione 3 ci sono i meccanismi di sicurezza login e password. È possibile mettere anche meccanismi di trap. Permette di realizzare la gestione remota di una macchina.

Livello di TRASPORTO
(Ric_Start)Il livello di trasporto fornisce una comunicazione logica tra gli host, saltando tutte le
problematiche della connessione di rete. Anche con il livello di trasporto parliamo di un protocollo end-to-end, perché non coinvolge nessuna delle macchine sottostanti, anche se ciò rappresenta un problema dato che non è possibile reperire informazioni utili dal livello sottostante. Il livello di trasporto piò parlare solamente con il livello di trasporto corrispondente. TSAP e NSAP. TSAP sta per Transport Service Access Point, ossia punto di accesso al servizio di trasporto. Il suo corrispondente è NSAP, ossia Network Service Access Point. Il TSAP coincide l’identificativo della porta, NSAP coincide con l’indirizzo IP della macchina(Ric_End).

Idee Generali sul come avviene la comunicazione

I messaggi a livello di trasporto vengono chiamati segmenti. Lato mittente il livello di trasporto converte i messaggi che riceve da un processo applicativo in segmenti: questo avviene spezzandoli in parti più piccole, e aggiungendo ad ognuno una propria intestazione. Il livello di trasporto passa i segmenti al livello di rete, il quale incapsula il segmento in un datagramma e lo invia al destinatario. Lato ricevente il livello di rete estrae dal datagramma il segmento, lo passa al livello di trasporto, il quale elabora a sua volta il segmento, estrae il pacchetto e lo rende disponibile al processo corrispondente(in base alla socket).
Internet fornisce due protocolli per il livello di trasporto UDP/TCP mentre basa il proprio livello di rete sul protocollo IP. IP è un protocollo inaffidabile che si basa sulla politica di best-effort delivery service: IP fa del suo meglio per trasferire i file con successo, ma non assicura una ceppa.

DEMULTIPLEXING/MULTIPLEXING
Lato ricevente il livello di trasporto esamina l'intestazione del segmento per capire a quale socket associarlo. Il compito di trasportare i dati dei segmenti verso la giusta socket prende il nome di demultiplexing.
Lato mittente, il compito di radunare i pacchetti provenienti dalle diverse socket ed incapsularli in segmenti a livello di trasporto per poi passarli al livello di rete, è detto multiplexing.
Tutto ciò avviene grazie all'intestazione che il livello di trasporto assegna a ciascun pacchetto(trasformandolo in segmento). Tale intestazione varia tra UDP(più semplice) e TCP(più pesante) ma entrambe presentano due campi fondamentali:
il numero di porta dell'host di origine ed il numero di porta dell'host destinazione.

!!!!DA RIVEDERE CON SAROOO
Differenza nel multiplexing
UDP stabilisce delle connessioni su una singola socket(per quel specifico processo), mentre TCP per lo stesso processo ha una socket padre,a partire dalla quale crea della socket figlia tramite cui stabilire connessioni col mittente.
Questo significa che al contrario di UDP, se due segmenti TCP in arrivo, hanno indirizzi IP di origine o numeri di porta di origine diversi, vengono diretti a due socket differenti.
!!!!DA RIVEDERE CON SAROOO

UDP
UDP è un protocollo a livello di trasporto minimale: non aggiunge nulla a IP se non la funzione di multiplexing/demultiplexing ed una forma di controllo degli errori molto semplice. UDP non usa handshaking e per tale motivo si dice che non è orientato alla connessione.
UDP non è un protocollo end-to-end in quanto non viene stabilita alcuna connessione. L'host mittente invia messaggi, UDP li consegna. Se i dati arrivano(cosa improbabile) l'host ricevente può usare l'intestazione del pacchetto per rispondere.
Il motivo per cui alcune applicazioni usano UDP è sicuramente quello che esso non introduce alcun ritardo nello stabilire una connessione: rispetto a TCP è più veloce nello spedire i pacchetti, i quali avendo una intestazione minimale possono includere più dati, inoltre non mantenendo alcun bit di stato, una connessione UDP può generalmente supportare più client attivi.
UDP viene usato in SNMP, DNS ma anche in molte applicazioni multimediali, lo streaming in tempo reale, le teleconferenze e così via. Solitamente si decide di usare UDP per applicazioni loss-tolerant in cui è sostenibile perdere dei dati durante la trasmissione.

Struttura segmenti UDP
L'intestazione UDP presenta solo 4 campi di due byte ciascuno. Oltre alla coppia

[numero di porta di origine, numero di porta di destinazione], abbiamo un campo lunghezza che specifica il numero di byte del segmento. Il campo checksum invece viene usato dall'host ricevente per verificare se sono avvenuti errori nel segmento. Lato mittente UDP effettua il complemento a 1 della somma di tutte le parole a 16bit del segmento e lo pone nel checksum. Lato destinatario si calcola la somma delle parole a 16bit e si sommano al checksum. Se il pacchetto è arrivato senza subire alterazioni, allora i bit saranno tutti a 1, altrimenti si è verificato un errore. Anche se UDP rivela l'errore in realtà non fa nulla per gestirlo. Spesso il pacchetto viene scartato o trasmesso con un avvertimento.

Principi Trasferimento Dati Affidabile
Premessa: tratteremo mittente e destinatario come automi a stati finiti

RDT 1.0 → trasferimento dati su un canale perfettamente affidabile(banale)
Il lato mittente di rdt accetta semplicemente dati dal livello superiore,crea un pacchetto e lo invia sul canale. Lato ricevente, raccoglie i pacchetti dal sottostante canale, rimuove i dati dai pacchetti e li passa al livello superiore. Con un canale perfetto non c'è bisogno che mittente e ricevente comunichino tra loro in quanto nulla può andare storto.

RDT 2.0 → trasferimento dati affidabile su un canale con errori sui bit
Un modello più realistico del canale è quello in cui i bit di un pacchetto possono essere corrotti. L'idea è quella di far si che il mittente prima di spedire un ulteriore pacchetto, debba ricevere dal destinatario un messaggio di verifica: un ACK costituisce una verifica positiva(i dati sono arrivati integri),un NAK invece rappresenta una notifica negativa(i dati sono stati corrotti).
Il mittente risulta quindi composto da due stati. Uno che attende dati dal livello superiore per creare un pacchetto e spedirlo, ed un altro che(dopo aver spedito il pacchetto) attende dal destinatario un messaggio di notifica. Se riceve un ACK allora il mittente sa che il pacchetto precedentemente trasmesso è stato ricevuto correttamente e quindi può procedere alla spedizione di un altro; se riceve un NAK allora il mittente ritrasmette l'ultimo pacchetto e attende una risposta alla trasmissione. Si usa il principio dello stop and wait: il mittente spedisce un pacchetto alla volta: se non riceve una notifica per quel pacchetto non può trasmetterne un altro.
In realtà bisogna tener conto anche della possibilità che anche i pacchetti ACK e NAK possano essere a loro volta alterati. La soluzione a questo problema consiste nel fatto che il mittente ritrasmetta il pacchetto di dati corrente a seguito della ricezione di un pacchetto NAK/ACK alterato. Tuttavia questo approccio introduce il problema di pacchetti duplicati(in quanto non è detto che il pacchetto sia stato realmente perduto). Una soluzione a questa problematica viene introdotta nel modello RDT 2.1: l'idea consiste nell'obbligare il mittente a numerare i propri pacchetti dati con un numero di sequenza(0/1). Al destinatario sarà sufficiente controllare questo numero per sapere se il pacchetto rappresenti una ritrasmissione o meno.

Il mittente esegue le stesse azioni di quello in RDT 2.0 ma presente ora 4 stati al fine di gestire i pacchetti con numero di sequenza 0 e quelli con numero di sequenza 1. Anche lato destinatario il numero di stati raddoppia.
Un altra modifica viene introdotta con RDT 2.2 : il destinatario, per segnalare al mittente di aver ricevuto un pacchetto corrotto, spedisce un ACK per l'ultimo pacchetto ricevuto correttamente, invece di inviare un segnale NAK. Così facendo il mittente che riceve due ACK per lo stesso pacchetto(ACK duplicati) sa che il destinatario non ha ricevuto quello successivo e quindi lo ritrasmette.

Trasferimento dati affidabile su un canale con perdite ed errori sui bit
Supponiamo ora che il canale oltre a danneggiare i bit possa anche perdere pacchetti. L'idea in questo caso è quella di utilizzare un timer: il mittente inizializza un contatore ogni volta che invia un pacchetto, se non si riceve un ACK per quel pacchetto prima che il timer scada allora lo ritrasmette.
Stimare la grandezza di un timer è molto difficile in quanto non si può mai avere la certezza che questo sia abbastanza grande da non introdurre troppi pacchetti duplicati ed abbastanza piccolo da non perdere troppo tempo prima di ritrasmetterlo.
RDT 3.0 introduce un timer e le primitive ad esso associate al fine di gestire la perdita di pacchetti.

Pipelining
Il protocollo RDT 3.0 risulta corretto dal punto di vista dell'affidabilità, ma risulta inefficiente in termini di prestazioni. Il problema sta nella strategia stop and wait: il mittente non potrà mai sfruttare tutta la banda disponibile se, prima di spedire un pacchetto, deve attendere un ACK per quello precedente.
Il pipelining è una metodologia che si basa sul continuo invio di pacchetti senza aspettare l'ACK di ricezione. Se ad esempio consentissimo al mittente di trasmettere tre pacchetti alla volta senza dover aspettare ACK di verifica, RDT 3.0 triplicherebbe il suo throughput.
Esistono due approcci per implementare il pipelining: GO-Back-N e ripetizione selettiva.

GO-Back-N(GBN)
Secondo questo modello il mittente può spedire più pacchetti senza dover attendere alcun ack, ma non può avere più di un numero massimo consentito di N pacchetti in attesa di un ack. All'interno della pipeline avremo 4 intervalli: il primo che comprende i numeri di sequenza per quei pacchetti per cui abbiamo ricevuto un ack di ritorno, il secondo comprende i numeri di sequenza dei pacchetti per cui non abbiamo ricevuto un ack di ritorno; il terzo comprende i numeri di sequenza dei pacchetti che possono essere inviati ma che non lo sono ancora stati; il quarto che comprende i numeri di sequenza dei pacchetti che non possono essere inviati.
N viene spesso chiamato ampiezza della finestra. Tale finestra è scorrevole: quando si riceve un ack per il pacchetto più vecchio all'interno della finestra, essa scorre verso destra,annettendo uno fra i pacchetti che possono essere spediti.
Il mittente in GBN deve rispondere a tre diversi tipi di evento:

Invocazione dall’alto:
Se la finestra non è piena, crea e invia un pacchetto, altrimenti, mantiene il pacchetto nel buffer in attesa.
Ricezione di un ack:
In questo modello la ricezione di un ack per il pacchetto con numero di sequenza n indica che tutti i pacchetti con un numero di sequenza minore o uguale a n sono stati correttamente ricevuti dal destinatario: in tal senso si parla di ack cumulativi.
In seguito alla ricezione di un ack la finestra di ricezione si sposta verso destra facendo così spazio a pacchetti che erano in attesa di essere spediti.
Timeout:
Se il timer per un pacchetto(quello meno recente) scade allora il mittente ritrasmette tutti i pacchetti per cui non abbiamo ancora ricevuto un ack di verifica.
Lato destinatario se un pacchetto con numero di sequenza n viene ricevuto correttamente ed è in ordine, il destinatario invia un ack per quel pacchetto. In tutti gli altri casi il destinatario scarta il pacchetto e rimanda un ACK per il pacchetto più recente ricevuto correttamente. La scelta di non conservare pacchetti non in ordine deriva dal fatto che GBN, lato mittente, in caso di smarrimento del pacchetto n rispedirà tutti i pacchetti a partire da n in avanti(quindi anche n+1). Ricordiamo che il destinatario non invierà ack in caso di ricezione del pacchetto n+1 in mancanza del pacchetto n, in quanto essendo gli ack cumulativi, inviare un ack per il pacchetto n+1 significherebbe dire al mittente di aver ricevuto anche il pacchetto n quando in realtà non è così. Se da un lato la ritrasmissione di tutti i pacchetti risulta dispendiosa,lato ricevente, egli deve memorizzare solo il numero di sequenza del pacchetto successivo.

Ripetizione Selettiva
Il principale problema di GBN è dato dal fatto che quando l’ampiezza della finestra è grande, nella pipeline si possono trovare numerosi pacchetti. Se uno viene perso, questo porterà ad un grande numero di ritrasmissioni(spesso inutili in quanto non abbiamo la certezza che tutti gli altri siano stati persi). I protocolli a ripetizione selettiva, ritrasmettono solo quei pacchetti su cui esistono sospetti di errori. Si userà sempre una finestra di ampiezza N, al cui interno però(a differenza di GBN) si troveranno sia pacchetti per cui si è ricevuto un ack e sia quelli che non ne hanno ricevuto. Il destinatario infatti invia un riscontro anche per i pacchetti ricevuti fuori sequenza: questi vengono memorizzati in un buffer del ricevente finché non sono stati ricevuti quelli mancanti. Quando il ricevente riceve un pacchetto con numero di sequenza uguale alla ‘base’ della finestra di ricezione, allora lo spedisce al livello superiore insieme a tutti i pacchetti nel buffer con numeri di sequenza consecutivi.
Lato mittente la finestra scorre solo nel caso in cui si è ricevuto un ack per il pacchetto più vecchio, rimane ferma nel caso in cui si riceve un ack per tutti gli altri.

TCP
TCP viene detto orientato alla connessione in quanto prima di effettuare lo scambio di dati i processi effettuano un handshake, ossia devono inviarsi dei segmenti preliminari per stabilire i parametri del successivo trasferimento dati.
TCP va in esecuzione solo sui sistemi periferici, i router intermedi sono completamente ignari delle connessioni TCP.
TCP offre una connessione full-duplex: i dati possono fluire da mittente a destinatario e viceversa. Essa è anche punto a punto(end-to-end) ossia ha luogo esclusivamente fra due host(il multicast non è ottenibile tramite TCP)

MSS(maximum segment size)
Definisce la massima quantità di dati inseriti in un segmento. Questo valore sommato all'intestazione TCP(20 byte) deve essere minore dell'unità trasmissiva massima(MTU), ovvero la grandezza massima di un frame che può essere inviato su quel collegamento.

Struttura segmenti TCP
Il segmento TCP consiste di campi di intestazione e di un campo contenente i dati provenienti dal processo applicativo.
Intestazione

1. Come UDP l'intestazione include i numeri di porta di origine e di destinazione ed di un campo checksum → totale 32 bit.
2. Campo numero di sequenza e campo numero di ack entrambi di 32 bit.
3. Campo finestra di ricezione di 16 bit.
4. Campo lunghezza dell'intestazione 4bit che specifica la lunghezza dell'intestazione in multipli di 32bit.
5. Campo opzioni usato per stabilire una connessione massima(opzionale)
6. Campo flag(6 bit). Contiene il bit di ACK, i bit RST, SYN e FIN che vengono usati per impostare e chiudere la connessione. I bit CWR e ECE usati per il controllo di congestione. Il bit PSH, se ha valore 1 indica al destinatario di spedire immediatamente i dati al livello superiore. Il bit URG indica la presenza di dati urgenti.
7. Campo puntatore dati urgenti: indica la posizione dell'ultimo byte di dati urgenti.

In TCP il numero di sequenza indica il numero del primo byte del segmento. Se un host sta inviando segmenti di 1000byte ciascuno allora il primo segmento ha numero di sequenza 0, il secondo 1000 , il terzo 2000 e così via.

Timeout_Come TCP reagisce alle perdite
TCP usa un timer per far fronte alla possibile perdita di pacchetti.
Si definisce RTT il tempo che intercorre tra l'invio del segmento e la ricezione del suo ack di verifica. Ovviamente il timer deve avere una valore maggiore di RTT ma di quanto? Al fine di scegliere un valore adatto, che riesca a far fronte alle fluttuazioni ed alla congestione della rete, TCP effettua una media ponderata tra il

precedente valore del timer ed il nuovo valore di RTT. Tale media prende il nome di EWMA ovvero media pesata esponenziale mobile.
Funzionamento
Prendo un campione(Sample RTT), lo peso con un fattore alpha e lo vado a sommare alla stime precedentemente pesata(Estimated RTT), la quale viene a sua volta pesata con un fattore 1-alpha. Tale media attribuisce maggiore importanza ai campioni recenti rispetto a quelli vecchi in quanto riflettono meglio la congestione attuale della rete. Ad ogni passaggio infatti, la media stimata in precedenza viene pesata con un fattore minore di 1. Oltre alla media, viene stimata anche la distanza(DevRTT)tra la media pesata ed ogni campione misurato. Per farlo si effettua un altra media pesata: la distanza(sample – estimated) viene pesata con un fattore beta, mentre la precedente stima viene pesata con un fatto 1-beta, e si sommano. Per calcolare il timer si somma il valore di Estimated RTT con il valore della distanza DevRTT moltiplicato per 4. Inizialmente il timeout viene impostato ad 1s, quando viene ricevuto il primo ack viene ricalcolato.
Esistono tre eventi principali relativi alla ritrasmissione:
Dati provenienti dall'applicazione: TCP incapsula i dati e quando spedisce il segmento, se il timer non è già in funzione lo avvia quando il segmento viene passato a IP.
Timeout: TCP risponde ritrasmettendo il segmento che lo ha causato e quindi riavviando il timer.
Ricezione ACK: TCP riavvia il timer se ci sono altri segmenti in attesa di ack
Se riceve un ACK
Nota: ogni volta che TCP ritrasmette un segmento per cui è scaduto il timer, imposta quest'ultimo ad un valore doppio del precedente(anzichè calcolarlo come definito in precedenza).
****
TCP usa ACK cumulativi.
TCP funziona come GBN ma senza ritrasmettere tutti i pacchetti in caso di scadenza del timer. Inoltre il destinatario TCP ha un buffer nel quale conserva i pacchetti che arrivano fuori sequenza: quando ne arriva uno, invece di spedire un ACK per quel pacchetto ne spedisce uno duplicato indicando il numero di sequenza del prossimo byte atteso.
Se arriva un pacchetto con numero di sequenza pari a SendBase, ovvero il numero di sequenza del pacchetto più vecchio spedito,allora TCP può far avanzare la finestra di trasmissione perché è certo che i pacchetti precedenti siano stati ricevuti correttamente. Potrebbe capitare che un ACK di conferma per il pacchetto con numero di sequenza SendBase venga perso. Se il mittente TCP riceve l'ACK per il pacchetto con numero di sequenza SendBase+1, capisce che l'ACK per il pacchetto precedente è stato perso e quindi aggiorna la sua variabile SendBase e fa scorrere la finestra: se il pacchetto con numero di sequenza SendBase non fosse stato ricevuto il destinatario TCP non avrebbe inviato l'ACK di conferma per SendBase+1 ma avrebbe inviato un ACK duplicato indicando il numero di sequenza SendBase, ovvero il pacchetto atteso.
****

Ritrasmissione Rapida

Al fine di diminuire i tempi di attesa nella ritrasmissione, TCP metta a disposizione un ulteriore servizio. Se il mittente riceve tre ACK duplicati per un segmento, allora egli considera il segmento che lo segue perduto e quindi lo ritrasmette, ancor prima che il timer per il segmento smarrito, scada. Ad esempio potrebbe accadere che il pacchetto n venga perso ma il destinatario riceva i pacchetti n+1, n+2 ed n+3 prima che scada il timer. Come conseguenza, conserva i pacchetti nel buffer ed invia un ACK duplicato per il pacchetto atteso, n appunto. Quando il mittente TCP riceve i tre ACK duplicati capisce che il pacchetto n si è perso e quindi lo ritrasmette.

Controllo di flusso

TCP offre un servizio di controllo di flusso affinché il mittente non saturi il buffer del ricevente.

****

ATTENZIONE: non confondere controllo di flusso con controllo di congestione, quest'ultimo si riferisce al rallentamento che TCP applica sul mittente a causa dei ritardi dovuti al traffico in rete e non al riempimento del buffer del ricevente che non riesce più a smaltire i pacchetti che riceve con conseguenti perdite.

****

TCP offre controllo di flusso facendo mantenere al mittente una variabile chiamata finestra di ricezione la quale fornisce al mittente un'indicazione dello spazio libero disponibile nel buffer del destinatario. La finestra di ricezione viene impostata uguale alla quantità di spazio disponibile nel buffer(scritto nel campo apposito del segmento TCP). Per garantire che il buffer non venga saturato, è necessario mantenere la quantità di segmenti spediti,per cui non si è ricevuto un ack di verifica, sotto al valore della finestra di ricezione. Supponiamo che il buffer si riempia. Ad un certo punto si libera dello spazio ed il destinatario lo comunica al mittente tramite un ACK: ma se questo messaggio viene perso? Tutti e due risultano bloccati.
Per evitare questo si usa un altro timer detto Persistent Timer che viene attivato quando viene ricevuto un messaggio di chiusura della finestra. Se il mittente non riceve più messaggi, allora il timer manda un messaggio di sblocco/sveglia al destinatario "Sei ancora vivo?".Se non dovesse arrivare risposta allora la connessione viene chiusa.

Sindrome della finestra stupida

Per evitare che ad ogni minimo spiraglio di buffer libero, il destinatario aggiorni il campo finestra, si cerca di mettere una pezza . La soluzione di Clark prevede che il ricevente non aggiorni subito la sua finestra di ricezione in caso di piccole variazioni.

Algoritmo di Nagle

Al fine di prevenire l'invio di pacchetti in cui la quantità di dati era irrilevante, Nagle propose un algoritmo la cui idea fondamentale è: non spedire un pacchetto per ogni tasto che è stato premuto, ma se ancora non hai ricevuto un ACK, accumula i byte che hai da spedire e appena possibile li spedisci. Questo ovviamente fa sì che, da un

lato miglioriamo le prestazioni del sistema perché non spediamo pacchetti troppo frequentemente, d'altro canto però l'interattività dell'utente è abbastanza scarsa perché non vede subito quello che sta facendo. L'algoritmo funziona bene quando il tempo di andata e ritorno(RTT) è elevato, perché altrimenti non ha nemmeno il tempo di mettersi in funzione. L'algoritmo è comunque disattivabile dall'applicativo.

Gestione della connessione TCP
Handshake a tre vie_Inizio della connessione
Il processo applicativo client informa il livello di trasporto client di voler stabilisce una connessione verso un processo nel server.
Step 1: TCP lato client invia uno speciale segmento al TCP lato server. Tale segmento non contiene dati ma ha il bit SYN posto a 1(viene infatti chiamato segmento SYN). Inoltre il client sceglie a caso un numero di sequenza iniziale.
Step 2: Il server TCP riceve il segmento, alloca variabili e buffer TCP e risponde con un altro segmento speciale, il quale non contiene dati, ma pone il bit SYN a 1, il campo ACK assume il valore del numero di sequenza successivo a quello scelto, ed infine il server sceglie il proprio numero di sequenza iniziale. Tale segmento è detto SYNACK.
Step 3: Ricevuto il segmento, anche il client TCP alloca variabili e buffer per la connessione TCP ed invia un terzo ed ultimo segmento speciale il quale ha il bit SYN posto a 0 ed il campo ACK con il numero di sequenza successivo(quello del server). A differenza dei primi due segmenti, il terzo può contenere PAYLOAD ovvero dati a livello applicativo.
In realtà i numeri di sequenza di partenza non sono scelti a caso ma secondo alcuni criteri.

Fine della connessione
Supponiamo che il client decida di chiudere la connessione. Allora il processo applicativo client invia un comando di chiusura che forza il client TCP ad inviare un segmento TCP speciale al server il quale contiene il bit FIN con valore 1. Il server, ricevuto il segmento, spedisce a sua volta un segmento speciale con il bit FIN a 1. Infine a sua volta il client manda un ack di verifica al server. A questo punto tutte le risorse per la connessione risultano deallocate.
La chiusura è unidirezionale,rispetto all'apertura che invece è bidirezionale. Nel senso che, possiamo chiudere la connessione da una macchina verso l'altra, e tenere aperta la connessione dati nel senso opposto. Questo vuol dire che se il client chiude la connessione verso il server, il server può continuare a spedire dati al client, e ottenerne le relative risposte(gli ACK)senza però inviare dati. In realtà non esiste un vero protocollo di chiusura e potrebbe accadere che da un lato la connessione rimanga aperta e dall'altro no. In realtà se il pacchetto FIN iniziale viene perso, l'host che vuole chiudere la connessione lo fa lo stesso, mentre l'altro continua ad inviare dati: quando riesce a capire che l'altro non vuole più comunicare allora la chiude.

Concetto di sfida

Host 1 vuole aprire una connessione con Host 2. E Host 2 risponde. Dato che Host 1 non vede l'Host 2 e Host 2 non vede l'Host 1, ma semplicemente possono scambiarsi il messaggio; chi c'è dall'altra parte? Ci sono messaggi vecchi provenienti da Host 2 che si sono persi? Che spuntano improvvisamente senza motivo? Oppure c'è un reale tentativo di aprire la connessione? Dato che non si vedono, la sfida è "io ti mando un numero, se tu ora sei presente, sei online, allora mi devi rispondere con un valore legato al numero che ti sto mandando". Poi la sfida viene invertita. L'Host 2 sfida l'Host 1 a fare la stessa operazione.
Tale sfida avviene al fine di garantire che vecchi messaggi che circolano nella rete,per qualunque motivo, non vadano ad aprire connessioni,quindi a occupare risorse in maniera inutile.

Controllo della congestione
Protocollo ATM
ATM permetteva ai router di comunicare quando si stavano riempiendo criticamente. Ma TCP non vede il livello di rete quindi non può essere applicato

Quando la congestione in rete risulta molto alta, è possibile che si verifichino degli overflow nei router causando perdita di pacchetti.
La congestione spesso è dovuta alla presenza in rete di copie di copie di pacchetti ritrasmessi perché il collegamento ha generato del ritardo sui router o perché andati persi. Sostanzialmente si arriva ad un situazione in cui, la maggior parte dei pacchetti in rete sono copie che non hanno alcuna utilità se non intasarla. TCP deve usare un controllo di congestione end-to-end dato che a livello di rete il protocollo IP non fornisce alcun feedback relativo alla congestione. L'idea è quella di limitare il mittente in base alla congestione "percepita": se il mittente si accorge di condizioni di scarso traffico, incrementa il proprio tasso trasmissivo, altrimenti lo diminuisce. Al fine di capire il livello di traffico, TCP fa tenere ai sistemi periferici una variabile aggiuntiva chiamata finestra di congestione. La quantità di segmenti per cui non si ha ancora ricevuto un ack deve essere minore del minimo tra, la finestra di congestione e la finestra di ricezione(associata al buffer del ricevente). Nelle nostre trattazioni assumiamo che la finestra di ricezione non ci dia problemi e concentriamoci su quella di congestione.
Il mittente verifica la congestione in rete in base al verificarsi o meno di eventi di perdita ovvero un timeout o la ricezione di tre ack duplicati.

***

TCP si dice auto-temporizzato perché se gli ack arrivano ad una frequenza alta, allora la finestra di ricezione(e quindi il tasso trasmissivo) viene ampliata velocemente, altrimenti, se la frequenza di ack è bassa, la finestra verrà ampliata piuttosto lentamente.

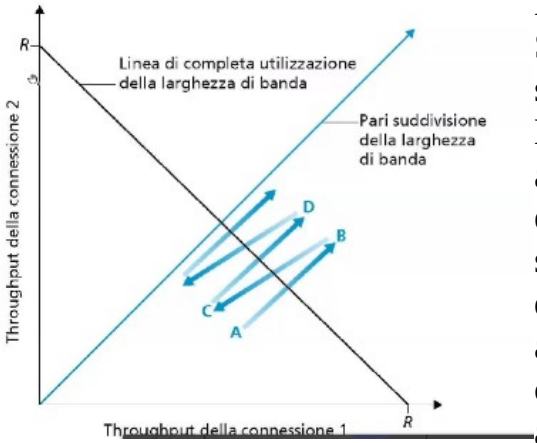
***

AIMD
L'algoritmo di controllo di congestione TCP si basa su tre componenti fondamentali: slow start, congestion avoidance e fast recovery.

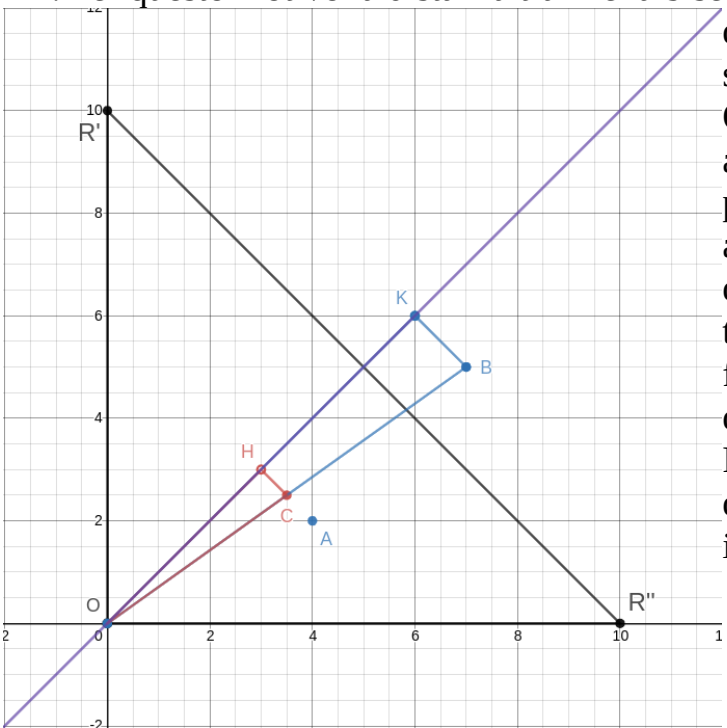
Slow start
Quando si stabilisce una connessione TCP la finestra di congestione viene settata a 1 MSS il che comporta una velocità di invio pari a MSS/RTT.
Durante la fase iniziale detta di slow start, il valore della finestra si incrementa di 1 MSS ogni volta che un segmento trasmesso riceve un ack(prima che si verifichi un evento di perdita). Se il valore della finestra è pari a 2 allora il mittente può inviare 2 segmenti. Se riceve gli ack per quei segmenti allora diventa 4, poi 8 e così via: la crescita è esponenziale. Per il controllo della congestione si fa riferimento ad un altra variabile aggiuntiva, una soglia. Quando si verifica un evento di perdita, la finestra di congestione viene settata a 1, mentre la soglia viene settata alla metà del valore che aveva la finestra prima dell'evento.
Congestion avoidance
La crescita esponenziale della finestra potrebbe risultare molto pericolosa(da 16 arriviamo a 32, 64, 128..) e causare un forte aumento della congestione. Per prevenire ciò, quando il valore della finestra supera quello della soglia, la crescita passa da esponenziale a lineare: invece di raddoppiare il valore della finestra ad ogni RTT, esso viene incrementato di 1 ad ogni blocco(RTT) di trasmissione. Nel caso in cui si verifichi un evento di perdita, dopo aver aggiornato i valori della soglia e della finestra, quest'ultima riprende a crescere in maniera esponenziale.
Fast recovery
Questo componente non è obbligatorio. In una prima versione TCP detta TCP Tahoe, qualunque fosse l'evento di perdita, il valore della finestra di ricezione veniva fatto ripartire da 1(per poi crescere esponenzialmente). In TCP Reno tramite Fast Recovery, quando l'evento di perdita è la ricezione di tre ack, invece di far ripartire la finestra di ricezione da 1(e rientrare nella fase di slow start), essa riparte dal valore della soglia, seguendo un incremento lineare(congestion avoidance).
Il controllo di congestione di TCP è spesso indicato come incremento additivo, decremento moltiplicativo(AIMD) in quanto, in condizioni stabili la finestra viene incrementata linearmente, quando si verifica un evento di perdita, viene dimezzata.
TCP Reno e Tahoe generalmente si comportano in maniera simile. Quando però essi si contendono una connessione, grazie al fast recovery, è TCP Reno ad avere la maggior porzione di banda, andando ad appropriarsi di quella che Tahoe rilascia quando avviene un evento di perdita(anche quando Reno parte dopo, vince sempre)

Fairness
Si dice che un meccanismo di congestione sia equo(fair) se la velocità trasmissiva media di ciascuna connessione,presente sul collegamento in esame, risulta uguale(o quasi) per ognuna di esse, ovvero se ciascuna ottiene la stessa porzione di banda.
Se TCP sta suddividendo la banda in modo uguale fra due connessioni, allora il throughput dovrebbe cadere sulla bisettrice del primo quadrante. L'obbiettivo dovrebbe essere ottenere throughput situati vicino all'intersezione fra la bisettrice e la linea di massimo utilizzo della banda.
I punti R1(sotto)=(R,0) ed R2(sopra)=(0,R) indicano rispettivamente che la connessione 1 o la connessione2 hanno preso tutta la banda disponibile. Infine il punto di coordinate (R/2, R/2) indica l'equa suddivisione della banda. Il segmento

disegnato tra i punti R1 E R2 ha punti di coordinate (x,y) tale che x+y=R ovvero y=-x+R. Il coefficiente angolare di questa retta è quindi pari a -1 e risulta perpendicolare alla bisettrice. Se superiamo la bisettrice, tenderemo di nuovo verso il basso(zona a minimo potenziale). Tutti i punti che sono nel triangolo rettangolo di vertici OR1R2 sono punti tale che la somma delle coordinate è minore o uguale a R. Quindi il segmento R1R2 indica la linea di completa utilizzazione possibile della banda. Anche se teoricamente non dovrebbe accadere, è possibile avere un throughput maggiore di R nel caso limite in cui ci sono dei buffer di ingresso in cui si sono accumulati dei pacchetti → può capitare di andare oltre la linea del massimo throughput.



Dimostriamo che c'è Fairness.
Supponiamo di trovarci in un punto sotto il segmento R1R2, ad esempio nel punto A oppure C. Dato che mi trovo al di sotto, le due connessioni aumentano entrambe il throughput. Supponendo di essere nel punto P(x,y) allora il successivo punto sarà P1(x+1,y+1). Per trovare la retta passante per i due punti si fa il sistema e si trova che il coefficiente angolare di questa retta sarà 1. Quindi i punti della connessioni si spostano su un segmento parallelo alla bisettrice(proprio per il valore del coefficiente angolare. La distanza dalla bisettrice rimane uguale quindi non miglioriamo ma nemmeno peggioriamo. Una volta che siamo nel punto B, supponiamo ci sia una perdita,per entrambe le connessioni. Allora le due dimezzano i loro valori e si spostano al punto C. Questo punto C si trova nelle coordinate(x/2 e y/2). Anche qui facciamo un sistema e viene fuori che il termine noto dell'equazione della retta è zero. Quindi questa retta sarà del tipo y=mx. Dato che il termine noto è 0 allora la retta passa per l'origine. Quindi il punto C si trova esattamente a metà della retta che parte da B e passa per l'origine. Inoltre C è più vicino alla bisettrice rispetto ad A infatti: guardando il triangolo OBK ed il triangolo OCH ci accorgiamo che sono simili dato che hanno gli angoli uguali tra loro.: l'angolo retto e l'angolo in H ed in K. Per questo motivo la distanza tra B e la bisettrice è esattamente il doppio della distanza tra C e la bisettrice. Quindi ci siamo avvicinati all'equità.



Quando si verificano eventi di perdita allora la banda utilizzata si riduce(A), per poi ricominciare ad aumentare fino ad arrivare a B. Se a B si verificano degli eventi di perdita il tasso trasmissivo viene nuovamente ridotto fino ad arrivare a C, per poi aumentare di nuovo.
Risulta importante sottolineare che le due connessioni TCP non comunicano in nessun modo. Esse sono infatti

indipendenti, ma tutte le trattazioni ed accorgimenti precedentemente affrontati hanno portato alla realizzazione di un sistema che funziona da solo e che mantiene una sorta di rispetto reciproco fra le connessioni.

A differenza di TCP, le connessioni UDP non sono fair. Le applicazioni multimediali infatti, invece di dover ridurre il proprio traffico, preferiscono
usare UDP e perdere pacchetti purché continuino a continuare a sostenere un alto tasso trasmissivo. Senza fairness non potrebbe esistere internet, infatti se non ci fosse questa proprietà, tutte quelle connessioni che arrivano dopo, non riuscirebbero a trovare spazio e non potrebbero comunicare.

Network Layer(Livello di rete per Andrea Bellomo)

Il livello di rete è il primo livello che si occupa della comunicazione attraverso i dispositivi intermedi presenti in rete.
Principalmente abbiamo due tipi di servizio: datagram e circuito virtuale
Il servizio datagram fa si che ogni pacchetto segua la strada indicata dal router(la più conveniente). Questo significa che potenzialmente per ogni pacchetto si potrebbe cambiare percorso → i pacchetti possono arrivare in ordine non corretto a destinazione. Nel sistema a circuito virtuale invece la strada che i pacchetti dovranno percorrere è una e viene settata a priori. Questo significa che si fa routing solo una volta: successivamente tutti i pacchetti seguiranno esclusivamente quella strada.
Vantaggi/Svantaggi
Nel sistema datagram non c'è necessità di creare a priori il circuito in quanto ogni pacchetto può seguire strade differenti, mentre nel sistema a circuito virtuale ciò avviene. Nel sistema datagram ogni pacchetto deve portare l'informazione sulla destinazione in modo che il router possa scegliere la strada giusta, mentre nel sistema virtuale, una volta creato il circuito non c'è più necessità di avere l'informazione completa. In datagram ogni router deve fare parecchio lavoro per capire quale strada scegliere, mentre in virtuale il router non deve fare nessuna operazione in quanto tutto già stabilito. In datagram un guasto ad un router non è un grosso problema, perché in quel caso, esso verrà evitato ed il pacchetto seguirà un'altra strada. In virtuale un guasto invece il circuito deve essere ricreato e non si potrà avere comunicazione.
Nel sistema a circuito virtuale la congestione viene gestita in modo semplice. Il primo pacchetto trasporterà ad ogni router le informazioni relative alla quantità di dati che arriveranno: in tal modo ogni router può dare consenso all'instradamento(se ha abbastanza risorse) altrimenti negarlo(se non ha risorse sufficienti).

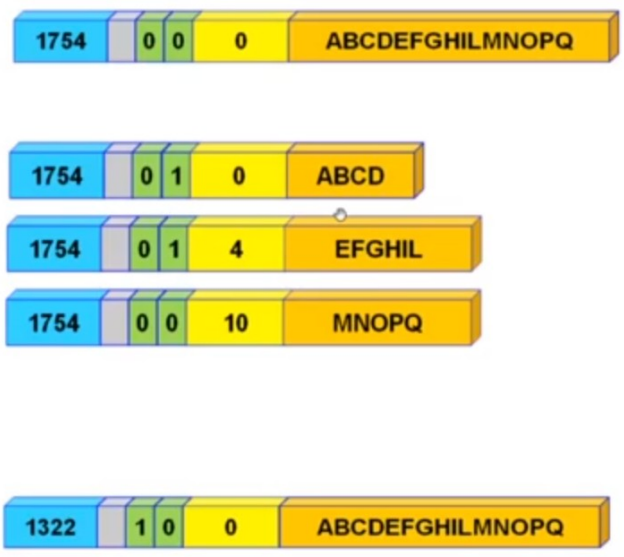
Protocollo IPv4

Formato datagrammi

I principali campi dei datagrammi IPv4 sono:

1. Numero di versione: sono 4 bit che specificano la versione del protocollo IP del datagramma la quale consente al router una corretta interpretazione di esso.
2. Lunghezza dell'intestazione: questi 4 bit indicano dove iniziano effettivamente i dati del datagramma. Essendo infatti il campo opzioni variabile, l'intestazione IP può avere lunghezza variabile anche se generalmente è di 20byte.
3. Tipo di servizio: viene usato per distinguere fra vari tipi di datagrammi quali datagrammi in tempo reale(telefonia) oppure da alto traffico(FTP).
4. Lunghezza del datagramma: misura in byte della grandezza(intestazione + dati) del datagramma. Teoricamente la massima dimensione dei datagrammi IP è 65.535 byte, ma generalmente non si va oltre i 1500 byte.
5. Identificatore, flag, offset di frammentazione: campi utili al processo di frammentazione dei datagrammi. L'identificatore, identifica il datagramma(ma non mi dire) , i flag sono DF(Don't fragment) e MF(More fragment)., mentre l'offset indica la sua posizione.
6. Tempo di vita: variabile inclusa per assicurare che i datagrammi non restino in circolazione all'infinito. Ogni volta che incontrano un router tale variabile viene diminuita di 1. Quando arriva a 0 il datagramma viene scartato.
7. Protocollo: indica il protocollo di trasporto al quale il datagramma è destinato. Ad esempio il valore 6 indica TCP, quello 17 indica UDP. Tale valore permette di collegare il livello di rete a quello di trasporto.
8. Checksum dell'intestazione: consente ai router di rilevare gli errori sui bit nei datagrammi ricevuti(nel caso vengono scartati). Tale verifica riguarda solo i bit dell'intestazione.
9. Indirizzi IP sorgente/destinazione: necessari per identificare mittente e destinatario.
10. Opzioni: questi campi consentono di estendere l'intestazione IP aggiungendo ulteriori informazioni. Essi costituiscono un problema in quanto possono avere lunghezza variabile e quindi il tempo per la loro elaborazione è variabile.
11. Dati(payload): contiene il segmento TCP/UDP da consegnare.

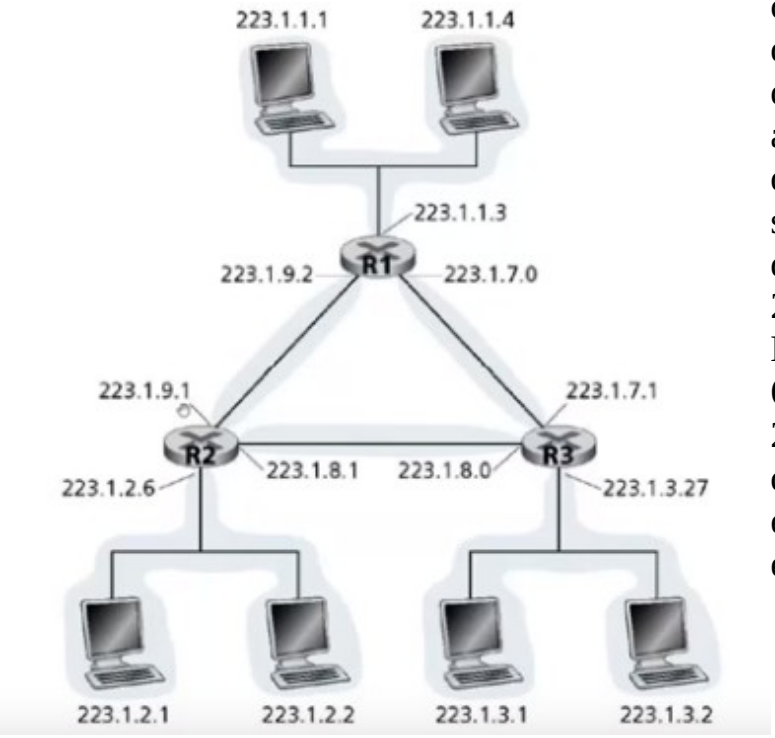
Frammentazione datagrammi IPv4



Per il trasporto da un router ad un altro i datagrammi vengono incapsulati in frame la cui grandezza deve essere minore dell'MTU. Può capitare che durante l'invio di un file, il percorso abbia dei collegamenti con diversi MTU. Potrebbe quindi accadere che un router riceva un datagramma, ma risulta impossibilitato ad incapsularlo in un frame ed inoltrarlo sul collegamento in uscita perché supera l'MTU. La soluzione a tale problema è

la frammentazione: consiste nello scomporre un datagramma in dei datagrammi più piccoli detti frammenti per poi trasferirli sul collegamento in uscita.
Prima di raggiungere il livello di trasporto i frammenti devono essere riassemblati. Per fare ciò si sfruttano i campi di identificazione,flag e offset.
Quando il router frammenta il datagramma assegna ai frammenti lo stesso indirizzo sorgente/destinazione e lo stesso identificatore numerico del datagramma originario.
Il destinatario per avere la certezza di aver ricevuto tutti i frammenti utilizza il campo offset per stabilire l'ordine corretto e capire se c'è qualche frammento mancante(un valore 0 indica che è il primo). L'offset indica il numero del byte precedente a quel frammento. Se il flag MF è posto a 0 allora il frammento in questione è l'ultimo.
Se un datagramma ha invece il flag DF posto a 1 allora non è possibile frammentarlo(bisogna trovare un'altra strada).

Indirizzamento IPv4
Generalmente un host ha un solo collegamento con la rete. Il confine fra host e collegamento fisico viene detto interfaccia. Dato che ogni router è agganciato almeno a due collegamenti egli presenta più interfacce(una per ciascun collegamento).
Il protocollo IP richiede che ogni interfaccia abbia un proprio indirizzo IP(univoco). Essi sono lunghi 4 byte ovvero 32 bit, per un totale di 2^32 ovvero circa 4 miliardi di indirizzi IP. Essi sono scritti seguendo la notazione decimale puntata: ciascun byte viene rappresentano con un numero nel range [0-255] e separato dagli altri byte con un punto.
Ogni interfaccia è collegata ad una sottorete:una sottorete(detta anche rete IP o semplicemente rete) è l'insieme delle interfacce associate ad un collegamento.
Ogni indirizzo IP ci fornisce delle informazioni relative alla sottorete cui l'interfaccia è collegata.
Ogni indirizzo IP dell'interfaccia avrà una parte comune, detta maschera di sottorete, ed una parte relativa all'host in sé e per sé. Ad esempio nella figura, la sottorete associata all'insieme delle interfacce dei tre host in alto e del router R1 collegato ha come indirizzo IP 223.1.1.0/24 dove la notazione /24 è la maschera di sottorete e ci dice che i 24bit più a sinistra forniscono l'indirizzo della sottorete. Ogni altro host che si vuole aggiungere alla sottorete dovrà avere un indirizzo IP del tipo 223.1.1.xxx.



Nell'esempio abbiamo un totale di 6 sottoreti: quella con indirizzo IP 223.1.1.0/24 ,
quella 223.1.9.0/24 tra il router R1 ed R2, 223.1.7.0/24 tra il router R1 ed R3, 223.1.8.0/24 per i due router

in basso R2 e R3, 223.1.2.0/24 per la sottorete che collega i due host + il router R2, 223.1.3.0/24 per la sottorete a destra.
In realtà per gli host/router non dovrebbe essere possibile avere degli indirizzi come 223.1.8.0 in quanto non validi a meno che non si tratti di una connessione punto a punto(così come non sono validi indirizzi con .255 tipico dei broadcast).

Assegnazione indirizzi IP
***La VLSM (*Variable Length Subnet Masking*) è una tecnica che permette la suddivisione ricorsiva dello spazio di indirizzi di un'organizzazione, al fine di utilizzarlo in maniera più efficiente. Tramite questa pratica, una subnet può essere partizionata in ulteriori subnet, attraverso l'utilizzo di parte dei bit destinati all'host number; si crea così un subnetting a più livelli. ***

La strategia odierna di assegnazione degli indirizzi IP è detta classless interdomain routing CIDR(sider). Ogni indirizzo è della forma a.b.c.d/x ed è diviso in due parti. I primi x bit costituiscono la parte di rete(maschera della sottorete) e sono spesso detti prefisso. I rimanenti 32-x bit possono essere usati per distinguere i vari dispositivi all'interno della rete.
Prima dell'adozione di CIDR le parti di rete di un indirizzo IP dovevano essere lunghe 8,16 o 24 bit. Tale schema era noto come classful addressing in quanto in base al numero di bit si distinguevano le classi A,B,C. Tale schema però risultò particolarmente ostico da adottare. Un indirizzo di classe C(/24) poteva infatti ospitare 2^8 – 2=254 host(due indirizzi sono riservati), mentre un indirizzo di class B(/16) poteva ospitare 65.634 host, uno sbalzo enorme.

Alcuni indirizzi sono riservati.
This host: rappresenta l'host stesso
Host della stessa subnet: se voglio contattare un host, e so per certo che questo si trova dentro la mia stessa subnet(ad esempio la mia stessa LAN) allora possono mettere a 0 tutta la parte relativa alla rete e lasciare solo la parte di indirizzo dedicata all'host.
Broadcast(255.255.255.255): tutte le macchine riceveranno il pacchetto.
Loopback: servono per identificare realmente this host. Quasi sempre l'indirizzo si trova nel formato 127.0.0.1 ma in realtà va bene qualunque indirizzo che inzi per 127.
RICORDA: 255 equivale a 1.
Se nell'indirizzo della subnet sostituisco i valori finali con 000.255.255.255 ottengo un broadcast nella subnet. Lo stesso avviene se faccio 127.255.255.255 ovvero un broadcast nella stessa macchina(inutile).
IP privati: iniziano con 10(classe B).Tutti i router bloccano questo tipo di indirizzi dai quali è dunque impossibile navigare in rete. Possono servire per fare LAN protette.
Zero Configuration Network: configurazione fatta dal sistema operativo in automatico in mancanza di informazioni dal mondo esterno. Anche questo non naviga su internet.

MAC Address(palese favoritismo verso Apple)
A livello LAN(dentro la rete) abbiamo i MAC Address che vengono scritti dal costruttore della scheda di rete, ed hanno un formato diverso. I MAC Address sono indirizzi A LIVELLO DI COLLEGAMENTO ed identificano i dispositivi all'interno della rete(LAN). Essi sono lunghi 6 byte il che consente di avere 2^48 possibili indirizzi. I primi tre byte rappresentano la localizzazione di chi ha prodotto la scheda mentre gli ultimi tre rappresentano la scheda prodotta. Sono espressi in notazione decimale, scrivendo due cifre esadecimali per ogni byte. Al giorno d'oggi è possibile cambiare indirizzo MAC via software. Una proprietà interessante è che sono univoci: non esistono due schede con lo stesso indirizzo MAC.
Differenze
Gli indirizzi IP hanno una struttura gerarchica(parte di rete e parte host) e devono essere cambiati quando un host si sposta, cioè quando cambia rete cui è collegato. L'indirizzo MAC invece non varia ed ha una struttura orizzontale
A differenza degli indirizzi IP l'indirizzo MAC non dà nessuna informazione sulla LAN cui è collegato. Anche qui esiste uno speciale indirizzo broadcast, costituito da una stringa i cui 48bit sono 1 ovvero FF-FF-FF-FF-FF-FF in notazione decimale. Quando un host vuole spedire un messaggio, deve fornire alla scheda di rete non solo il datagramma IP ma anche l'indirizzo MAC della macchina.

In sostanza: le macchine comunicano tra di loro tramite indirizzo IP, ma le frame(a livello di collegamento) viaggiano usando il MAC address, quindi serve un protocollo che permetta di passare dal MAC Address, all' IP e viceversa.
Il protocollo che ci permette di fare questo è ARP

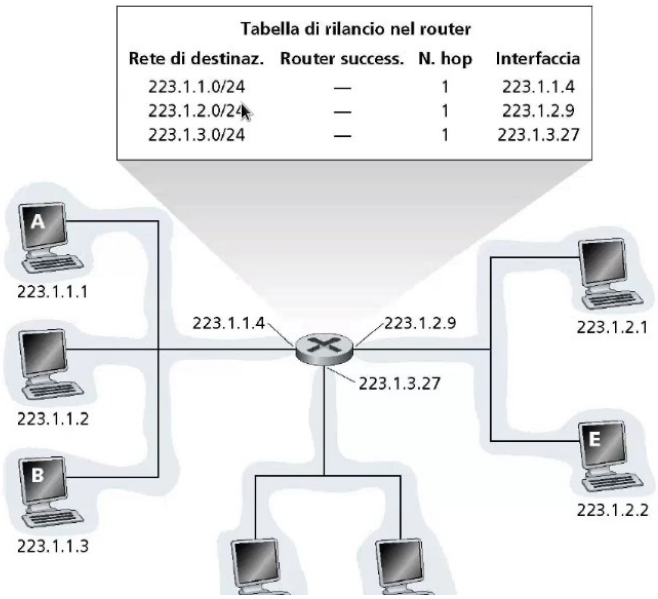
ARP(Address Resolution Protocol)
Supponiamo che un host A voglia inviare un datagramma ad un host B presente nella sua stessa sottorete(LAN). Per farlo A deve trasmettere alla sua scheda di rete, oltre al datagramma IP, anche il MAC di B. Per determinare l'indirizzo MAC di B, A usa il protocollo ARP il quale fornirà la traduzione dell'indirizzo IP in indirizzo MAC. ARP è paragonabile a DNS: la differenza sostanziale sta nel fatto che DNS effettua una traduzione di nome host-indirizzo IP in tutto il mondo, mentre ARP effettua traduzione indirizzo IP-l'indirizzo MAC solo per host nella stessa sottorete; inoltre DNS è un insieme di server, mentre ARP è un protocollo distribuito.

Funzionamento
La conversione da indirizzo IP ad indirizzo MAC avviene grazie alla così detta tabella ARP presente nella RAM degli host. La tabella oltre a contenere una corrispondenza fra indirizzi IP ed indirizzi MAC, contiene anche un valore relativo al TTL(time to live) ovvero il tempo di vita che indica quando bisognerà eliminare una voce all'interno della tabella(tipicamente 20 minuti).

Indirizzo IP	Indirizzo LAN	TTL
222.222.222.221	88-B2-2F-54-1A-0F	13:45:00
222.222.222.223	5C-66-AB-90-75-B1	13:52:00

Se la tabella del modulo ARP di un nodo non possiede un record per un determinato indirizzo IP allora, il nodo trasmittente, costruisce uno speciale pacchetto chiamato pacchetto ARP con lo scopo di interrogare gli altri nodi della stessa sottorete.
La scheda di rete trasmette dunque il pacchetto speciale ARP con indirizzo MAC broadcast FF-FF-FF-FF-FF-FF, al fine di raggiungere tutti gli host collegati. Il frame viene ricevuto da tutti gli host, ma solo quello che ha l'indirizzo IP corrispondente, invia all'host mittente un frame di risposta ARP con l'indirizzo MAC. Il nodo mittente aggiornerà la sua tabella ARP e potrà ora trasmettere il messaggio.

Supponiamo di avere un router che collega due gruppi di host(appartenenti a due sottoreti distinte). Quest'ultimo presenterà dunque due interfacce, due moduli ARP , due schede di rete, due indirizzi IP e due indirizzi ARP. Supponiamo che l'host A, nella sottorete 1, voglia inviare un datagramma all'host B nella sottorete 2. Affinchè un datagramma possa passare da una sottorete all'altra si usa l'indirizzo MAC del router: inizialmente l'host A invia al router il datagramma(tramite l'uso di ARP per la traduzione del suo indirizzo IP); ricevuto il datagramma il router vede l'indirizzo IP di destinazione del datagramma, consulta la propria tabella ARP e crea un frame da spedire all'host corretto nella sottorete 2 tramite la sua scheda di rete(per la sottorete 2).
Il mittente può a priori capire se l'indirizzo IP cui deve spedire il datagramma è nella sua sottorete o meno.
Facendo un AND con il suo IP e la sua Mask egli ottiene una stringa contenente l'indirizzo di tutta la sottorete di appartenenza(e poi tutti zeri). Poi, effettua la stessa operazione con l'IP dell'host destinazione e con la sua sottorete. Se il risultato è identico allora il destinatario appartiene alla sua stessa sottorete, altrimenti no.
Per capire se è identico si effettua una XNOR tra i due risultati ottenuti: se ottengo tutti 1 allora la destinazione si trova nella mia stessa LAN.
Quando si configura una macchina in una rete sarebbe utile darle almeno tre informazioni di base: il suo indirizzo IP, la sua maschera di sottorete e l'indirizzo IP del router di uscita(Gateway).
Tabella di Routing
Ogni macchina(non solo il router) ha una tabella di routing, utile nel caso in cui si vogliano contattare delle destinazioni fuori dalla propria sottorete. Esse contengono(in base all'implementazione): una rete di destinazione con una maschera, L'IP del router da contattare per uscire verso la destinazione ed una metrica per capire quanto convenga quell'uscita(se non ci sono router successivi verso quella destinazione, allora è nella sua stessa sottorete).
Nel caso di un solo gateway di uscita



Nella tabella manca l'informazione relativa al router successivo ed inoltre la metrica avrà per tutte le destinazioni lo stesso valore in quanto si passa solo dal router centrale

Problemi ARP

<u>Il protocollo ARP non prevede alcuna autenticazione</u>
Quando viene effettuata una ricerca tramite broadcast con il pacchetto speciale ARP al fine di ricevere il giusto indirizzo MAC, un solo nodo risponde, ma è quello giusto?Non c'è alcuna possibilità di verificare che quella macchina sia realmente quella che cercavamo.

<u>ARP è un protocollo senza stato</u>
Potrebbe accadere che un host nella sottorete, invii automaticamente(con brutte intenzioni) una risposta, per il suo indirizzo MAC, senza alcuna richiesta. Questa informazione verrà accettata da tutte le macchine,le quali aggiorneranno le loro tabelle ARP,e data per corretta senza nessun controllo. Quindi non possiamo fidarci della risposta che ci arriva.

Possibili scenari:
Potrebbe accadere che una macchina riceva due risposte per lo stesso indirizzo IP(durante la ricerca del MAC). Solitamente è il secondo messaggio che va a sovrascrivere il primo. A questo punto ogni messaggio diretto a quell'indirizzo IP sarà ricevuto dalla macchina ingannatrice realizzando così un attacco Man in the Middle. Per tale motivo si ritiene molto inefficace basare l'autenticazione solo sul MAC Address.
Un altro scenario che può accadere potrebbe essere quello secondo il quale una macchina si finga il router di uscita. Il tal modo essa può avere accesso(modificarli, ritardarli, far cadere la connessione, filtrarli) a tutti i pacchetti che erano destinati al router. Questo tipo di attacco risulta molto più pericoloso rispetto a quello precedente, in quanto può portare a conseguenze più gravi.

Facendo solo sniffing banale, l'attaccante potrebbe essere saturato da tutto il traffico in rete, mentre con l'attacco Man in the Middle allora il suo compito di analisi del traffico diventa molto più semplice(gli arriva solo quello che gli interessa).

RARP
Reverse ARP, ovvero fa la traduzione opposta, da indirizzo MAC a quello IP. A differenza di ARP non è un protocollo distribuito in quanto prevede la presenza di un server RARP che conosce tutti gli host nella rete.
La richiesta viene fatta sempre tramite broadcast ed a rispondere è il server RARP. Tale protocollo non viene usato spesso proprio per la necessità di avere un server che risponde alle richieste.

BOTP
Protocollo pensato per effettuare bootstrap in presenza di macchine senza sistema operativo. L'obbiettivo è quello di caricare, installare un sistema operativo da rete in una macchina che, nella versione più banale, non ha niente(disco fisso, memoria) ma conosce solo il proprio MAC Address. Deve esistere un server che permette di mandare il kernel dell'OS per effettuare il boot del sistema. Fatto questo ed avendo un MAC address la macchina può ricevere un IP da un server RARP: tramite l'IP la macchina potrà connettersi in rete e chiedere al server di mandare tutto il sistema operativo. L'operazione di trasferimento file viene fatta con TFTP, dove la T sta per Trivial. Essenzialmente è una versione di FTP senza autenticazione, viene bypassato il controllo dell'utente. BOOTP era utilissimo quando gli hard disk avevano un costo non indifferente ed era utile per creare delle macchine "terminale" senza un hardware dedicato.

DHCP(Dynamic host configuration Protocol)
DHCP è un protocollo che consente ad un host che si collega ad una rete di ottenere un indirizzo IP in modo automatico, così come di apprendere informazioni aggiuntive, quali la sua maschera di sottorete , l'indirizzo del router per uscire dalla sottorete(gateway) e l'indirizzo del suo DNS server locale. DHCP può essere configurato in modo da fornire ad un dato host lo stesso indirizzo IP ogni qualvolta si collega, oppure un indirizzo IP temporaneo, che sarà diverso tutte le volte che l'hoost si collega alla rete.
DHCP viene spesso detto protocollo plug-and-play o zero-conf in quanto permette di automatizzare la connessione degli host alla rete.
DHCP è adatto alla situazione nella quale ci sono molti utenti che vanno e vengono da una rete, fornendo degli indirizzi che sono necessari solo èer una quantità di tempo limitata: in tal modo si evita di riconfigurare il portatile a ogni nuova connessione.

DHCP è un protocollo client-server. Un client è di solito un host appena connesso che desidera ottenere informazioni sulla configurazione della rete. Ogni sottorete dispone di almeno un server DHCP.

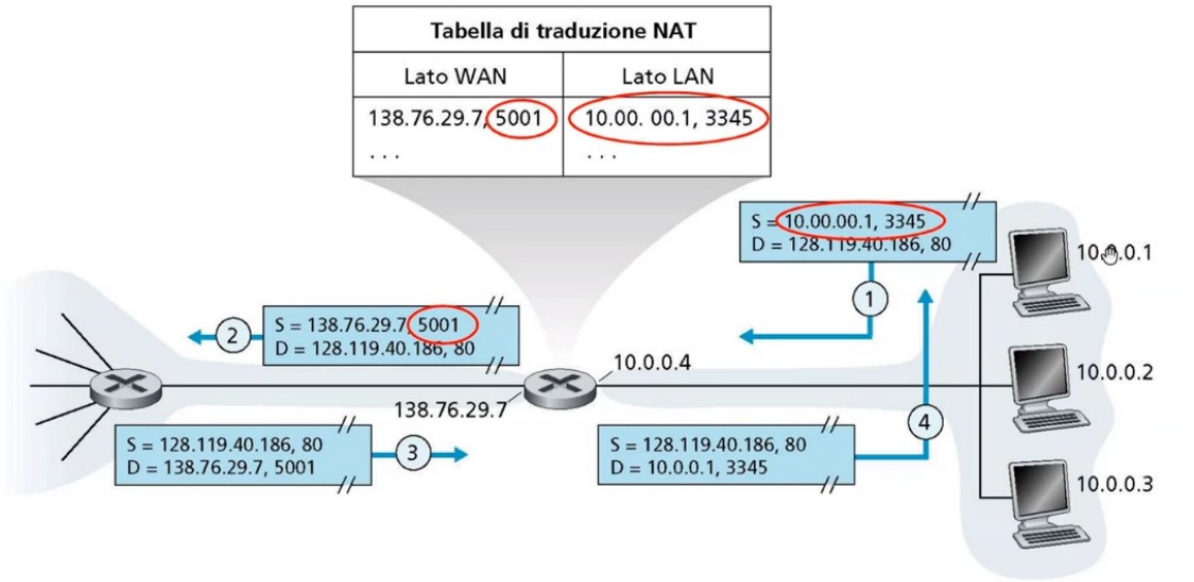
Funzionamento(UDDPPPPP)
Supponiamo che un host A, si sia appena collegato ad una sottorete. Tutti i messaggi vengono per semplicità e trasparenza, inviati in broadcast.

1. Individuazione del server DHCP: l'host invia un messaggio chiamato DHCP discover in un pacchetto UDP tramite la porta 67. Tale pacchetto ha indirizzo IP d'origine quello dell'host da cui è generato(0.0.0.0) ed un codice detto transaction ID.
2. Offerta del server DHCP: il(o i) server DHCP risponde al client con un messaggio detto DHCP offer che contiene: l'indirizzo IP del server DHCP, lo stesso transaction ID,l'indirizzo IP proposto al client, la maschera di sottorete ed il lifetime(o lease time) ovvero la quantità di tempo per cui l'IP sarà valido.
3. Richiesta DHCP: il client appena collegato sceglierà tra le offerte dei server(l'idea è quella di scegliere l'IP con lifetime maggiore) e risponde a quello scelto con un DHCP request che riporta i parametri di configurazione. Il transaction ID viene aumentato di 1.
4. Conferma DHCP: il server risponde al messaggio di richiesta DHCP con un messaggio DHCP ACK di conferma.

Nel caso in cui il client voglia continuare ad operare in rete, DHCP offre un meccanismo per aumentare il lifetime dell'indirizzo IP assegnatogli.
Problemi
Cosa succede se scade il lifetime?
La regola dice che o viene rinnovata la richiesta da parte del client, oppure l'IP viene riciclato. Se l'host non rinnova la richiesta(quindi continua ad usare l'IP scaduto) ed il server DHCP(non sapendolo) affida quell'IP ad un nuovo host collegato alla rete, nella rete saranno presenti due macchine distinte con lo stesso IP.
Questo causa dei problemi ad esempio con le richieste ARP, in quanto una macchina che effettua una richiesta ARP, si ritrova con due risposte da parte di due macchine che dicono di avere l'IP cercato ma con indirizzo MAC differenti. Il tutto è ancora più problematico nel caso di un server DHCP abusivo o malevolo, in più rispetto a quello legittimo. In questo caso alcune macchine riusciranno a connettersi alla rete e altre no.

NAT(network address translation)
Spesso all'interno della LAN conviene vere indirizzi privati(10.0.0.0/8) in modo da non avere nessun obbligo verso le politiche globali ed evitare qualsivoglia errori. Gli indirizzi privati hanno però significato solo per gli i dispositivi interni, mentre non possono scambiare messaggi col mondo esterno. Il servizio che permette agli IP privati di scambiare messaggi con l'esterno è fornito dal protocollo NAT.
In figura tutto il traffico dalla rete domestica, cui sono associati i tre host + il router, al mondo esterno ha come indirizzo IP sorgente quello dell'interfaccia che il router ha verso il lato WAN, il quale viene usato anche come indirizzo IP di destinazione per il

traffico dal mondo esterno alla rete domestica. Il router, lato LAN ha un indirizzo IP privato facente parte della sottorete 10.0.0.0/24 mentre lato WAN ha un indirizzo IP pubblico: egli non fa altro che nascondere i dettagli della rete domestica al mondo esterno dirigendo su di se tutto il traffico per poi inoltrarlo all'host corrispondente.
Ma come fa il router a dirigere il traffico verso un determinato host se il suo indirizzo è sconosciuto dai dispositivi esterni? Viene effettuata una sorta di traduzione tramite la tabella di traduzione NAT in cui oltre agli indirizzi IP vengono usati anche i numeri di porta.



Supponiamo che l'host con iP 10.0.0.1 voglia effettuare una richiesta verso il server con IP 128.119.40.186. L'host assegna un numero di porta di origine arbitrario, ed invia il datagramma alla rete locale. Il router NAT riceve il datagramma, genera un nuovo numero di porta di origine e sostituisce l'indirizzo IP di origine dell'host con il proprio IP sul lato WAN(ossia quello riconosciuto dal mondo esterno).
Il NAT del router aggiunge un nuovo record alla propria tabella di traduzione NAT che associa all'IP del router con quel numero di porta, l'IP dell'host client con il suo numero di porta. Il router infine spedisce il datagramma al server, il quale ignaro della manipolazione avvenuta, risponde con l'oggetto richiesto. Quando il datagramma originato dal server arriva al router, questo consulta la tabella di traduzione NAT per ottenere l'indirizzo IP corrispondente al numero di porta del pacchetto, ed invia all'host client il datagramma ricevuto dal server esterno.

Problemi di NAT

1. In pratica per quanto concerne il mondo esterno, è come se tutte le operazioni venissero eseguite dalla stessa macchina.
2. Utilizzo della porta: essendo il numero di porta un numero a 16bit il router può scegliere tra circa 65000 porte, ma in caso di LAN molto grandi, egli potrebbe saturarsi per via delle troppe connessioni.
3. Una volta usato un numero di porta per una connessione, esso non potrà essere utilizzato per comunicare fintanto che non sarà cancellato dalla tabella NAT.

4. Se un pacchetto arriva dal mondo esterno il NAT, non essendoci un record in tabella, non sa che traduzione fare e quindi lo scarta. Con NAT la comunicazione deve sempre partire dalla LAN al mondo esterno.
   Soluzioni:
- Mettere una riga statiche nella tabella, indicando un aparto di entrata che gira in automatico il pacchetto ad una data macchina nel lato LAN: UpnP.

UpnP
Protocollo che permette ad un host di mandare un messaggio al routwr NAT in modo settare di default che il traffico proveniente dall'esterno verso la LAN venga indirizzato all'host stesso.
Tali record nella tabella NAT hanno una scadenza.

IPv6
Si è deciso di passare da IPv4 ad IPv6 per tre motivi principali:
1. Esaurimento dello spazio degli indirizzi dato da un'allocazione iniziale inefficiente e dall'estrema diffusione di internet.
2. Scalabilità del routing: dovrebbe essere fatto in maniera geografica in cui indiirizzi IP numericamente vicini sono poco distanti geograficamente.
3. Implementazione di nuovi servizi.

Nuovi servizi di IPv6:
1. Sicurezza
2. Autoconfigurazione
3. Gestione della Quality of Service
4. Indirizzamento multicast(tanti host in una stessa rete)
5. Indirizzamento host mobili

Principali differenze
1. Si passa da indirizzi IP di 32 bit a 128 bit in modo tale che essi diventino praticamente inesauribili.
2. Intestazione fissa di 40 byte che consente una più rapida elaborazione dei datagrammi IP.
3. Etichettatura dei flussi che permette di evitare di fare routing per ogni pacchetto in quanto tutti quelli appartenenti allo stesso stream seguiranno tutti la stessa strada in ordine. È un modo per implementare il circuito virtuale.

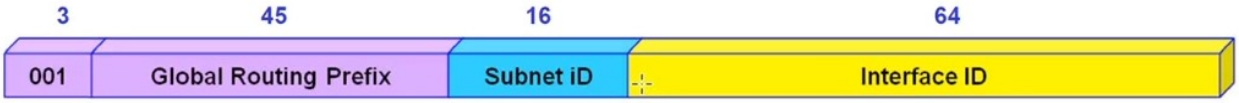
Formato e Campi
Versione: campo a 4 bit che indica il numero di versione IP.
Classe di traffico: campo a 8 bit usato per attribuire priorità a determinati datagrammi.
Etichetta di flusso(Flow Label): campo a 20 bit che indica l'appartenenza ad un determinato stream.
Lunghezza del payload: valore a 16 bit che indica il numero di byte nel datagramma.
Intestazione successiva: identifica il protocollo di trasporto cui è diretto il datagramma.

Limite di hop: indica il lifetime del datagramma. Viene decrementato di 1 ogni volta che viene inoltrato da un router, fino a quando non raggiunge 0 venendo così scartato.
Indirizzi sorgente e destinazione: pesano molto di più
Dati: Payload.

Sono stati eliminati i campi:
Frammentazione/Riassemblaggio: IPv6 non consente la frammentazione sui router intermedi in quanto è ritenuta un'operazione troppo dispendiosa. Essa viene affidata all'host sorgente o destinazione. Se il router riceve un datagramma troppo grande, esso viene scartato ed il router invia al mittente un messaggio di errore.
Checksum: data la presenza di un checksum sia a livello di trasporto che a livello di collegamento si è optato per eliminare tale campo in modo da alleggerire il compito dei router.
Opzioni: il campo opzioni non fa più parte dell'intestazione in modo che essa abbia una grandezza fissata.

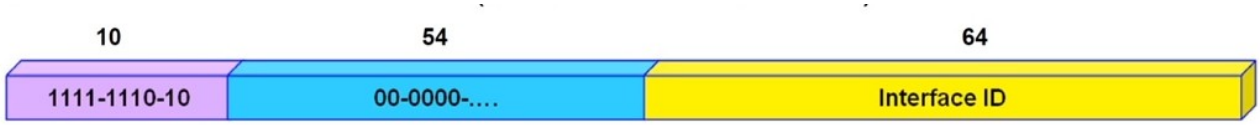
Struttura indirizzo IPv6
Ogni indirizzo è diviso in 8 blocchi da 16 bit l'uno: ogni cifra è capace di rappresentare 16 valori ed è quindi rappresentabile in esadecimale.
Per convenzione un indirizzo del tipo:
8000:0000:0000:0000:0123:4567:89AB:CDEF
può essere compresso nella forma:
8000 :: 0123:4567:89AB:CDEF, dove i doppi due punti indicano tutti zeri fino a quando non c'è un cambiamento.

Global Unicast Addresses



Come per IPv4 anche qui possiamo suddividere l'indirizzo in una parte di rete ed in una parte di host(spesso 64 e 64).
Indirizzo Unicast signfica che indica una singola macchina. I primi 48 bit rappresentano il Global Routing Prefix: i primi 3 bit indicano che tipo di indirizzo è, gli ulteriori 45 indicano invece qual'è il sito, ovvero la regione cui è stato assegnato questo ID. Altri 16 bit vengono utilizzati per identificare la subnet.
Infine abbiamo 64 bit per l'interfaccia.

Indirizzi Locali: FE80::/64

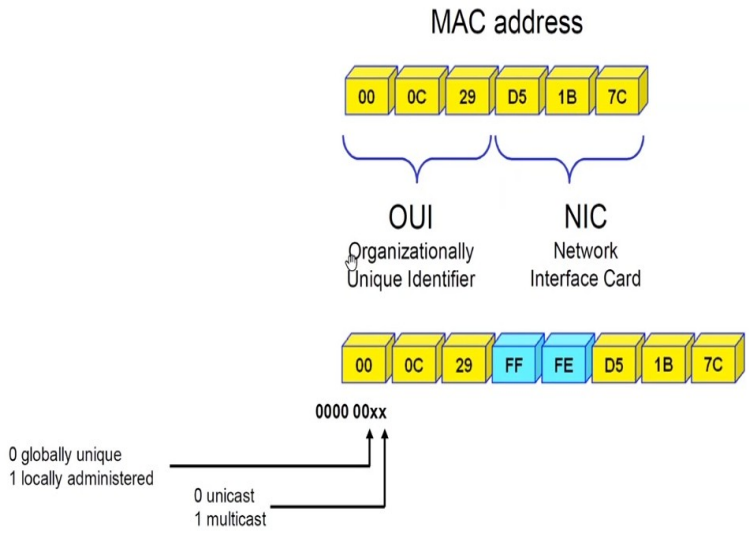


Vengono utilizzati solo tra nodi dello stesso link. FE80 occupa 10bit.

Indirizzi di sito FEC0::48

Si possono usare solo tra nodi di uno stesso sito.
Entrambi, indirizzi locali e di sito non possono essere usati su Internet.

Indirizzi IPv6 EUI-64



Vengono presi i 3 byte iniziali e ricopiati, si aggiunge una coppia FF:FE ed infine si copiano gli ultimi 3 byte. In tal modo si passa da 6 a 8 byte. Il primo byte è della forma 0000 00xx.
La prima x: se 0 indica che l'indirizzo è Globally Unique, se 1 indica che è un indirizzo di sito.
La seconda x: se 0 indica che è unicast, se è 1 indica che è multicast.

Il protocollo forza ad 1 il penultimo bit(la prima x): in tal modo viene creato un indirizzo IPv6 valido che però non può navigare su internet perché indirizzo di sito.
Ogni macchina può quindi crearsi il suo indirizzo IPv6 univoco in quanto gli indirizzi MAC di partenza sono univoci.

Unicast_Multicast_Anycast
Unicast: gli indirizzi Unicast iniziano con 2 o 3 come primo bit, quindi iniziano con 001(che può diventare 0010 o 0011). Indirizzano il pacchetto verso una singola macchina.
Multicast
Gli indirizzi Multicast iniziano con FFxx, dove i byte xx possono variare andando a differenziare tra varie tipologie. Uno di quelli che ci può interessare FF02::2 → All router address. L'idea è che a differenza del broadcast, indirizziamo il pacchetto solo

ad un gruppo di host interessati. In tal modo si riduce anche il traffico in rete e l'invio di dati ad host che non sono interessati.

Anycast.
Hanno la parte finale composta solo da zeri.
L'idea è: Il primo router che riuscirà a gestire l'indirizzo Anycast, lo indirizzerà verso la macchina più conveniente. Quindi io non vado ad indirizzare una singola macchina, ma indirizzo a un Router che poi smisterà il pacchetto. È uguale all'unicast, ma qui la macchina destinatario è random.

IPv6 over Ethernet
I pacchetti IPv6 vengono incapsulati in frame Ethernet esattamente come per IPv4, l'unico cambiamento è nel campo Ethertype: 86DD invece di 0800. In tal modo permette di distinguere che tipo di pacchetto contiene il frame Ethernet.

Neighbour Discovery Protocol
Protocollo utile per scoprire quali siano gli host vicini presenti nella nostra rete ed i router. Il tutto viene gestito grazie agli indirizzi Multicast. Va ad implementare quello che era il protocollo ARP dentro il protocollo IPv6.

Classi di Traffico in Ipv6
000-111= time insensitive
I primi tre bit indicano quanto è sensibile al tempo la comunicazione. Servono per comunicare al router di dare priorità ai pacchetti più sensibili al tempo.
1000-111= priotirà dei pacchetti
Indica cosa scartare e cosa non scartare assolutamente.

Header Opzionali
Sono quei campi che in IPv4 venivano inseriti nel campo Opzioni rendendo variabile la dimensione dei datagrammi.
Il campo Next Header specifica quanto è grande il pezzo che segue.

In generale la struttura sarà sempre:
Header IPv6 Next Header=something > something Header Next Header = somethingelse...
Questo genere di struttura agevola la frammentazione in IPv6.

Gli header opzionali sono dotati di un numero, in particolare:
6 indica TCP, 58 indica ICMP, 59 indica NULL -> nessun header a seguire.

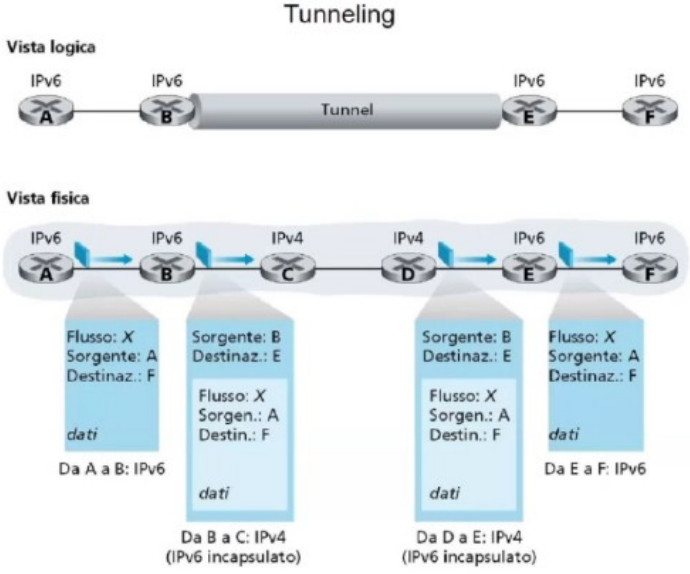
Frammentazione
A differenza di IPv4, in IPv6 la frammentazione è affidata solo agli host periferici.
In generale conviene capire qual'è l'MTU massima che il canale può sopportare è frammentare di conseguenza.
Bisogna ricordare che può essere frammentato solo ciò che si trova dopo il Fragment Header ovvero l'header opzionale che si trova prima dell'intestazione TCP e del Payload.

Sicurezza
Tra i vari header opzionali esiste un Authentication Header che garantisce autenticità e correttezza del pacchetto ed un Encrypted Security Payload Header che permette di criptare i dati.

Migrazione da IPv4 a IPv6
Ancora oggi esistono strutture che lavorano con indirizzi IPv4.
Per effettuare il passaggio da IPv4 ad IPv6 e viceversa, esistono diversi approcci.
Dual Stack
Prevede di avere nella rete domenstica due stack, uno per i datagrammi IPv4 ed uno per quelli IPv6. Quando arriva un frame Ethernet, in base al suo campo Ethertype, sapremo se la frame dovrà salire la pila protocollare con IPv4 o quella con IPv6

Tunneling
Il problema si pone quando una macchina IPv6 vuole parlare con un'altra Ipv6 passando però attraverso una rete IPv4.

La soluzione consiste nel creare un tunnel. Il router di transito B prende il pacchetto v6 e lo pone nel campo dati di in un altro pacchetto v4. A questo punto, non sapendo cosa si cela nel campo dati, i router di transito IPv4 instradano il datagramma normalmente fino ad arrivare al router E Ipv6. E, esaminando che il valore nel campo
numero di protocollo nel pacchetto Ipv4 è 41, estrae il pacchetto Ipv6 e lo instrada come se l'avesse ricevuto da un qualsiasi router Ipv6.



Il problema contrario non si pone perché gli indirizzi IPv4 sono mappabili dentro IPv6.

ICMPv4(Internet Control Message Protocol)
Permette di mandare messaggi di controllo che non richiedono porte di comunicazione(ping). Essendo parte di questo protocollo possiamo disabilitare il ping: la macchina è on, ma non risponde al ping, risultando invisibile alla rete.
Un altro comando interessante è il Traceroute che ci dà informazioni sullo stato di rete verso la destinazione cui dovremmo spedire il datagramma.
Il mittente invia un datagramma privo di dati, con un TTL pari ad 1, verso il primo router. Quest'ultimo si accorge che il pacchetto ha il proprio TTL scaduto e lo scarta, mandando una notifica al mittente. Il processo viene iterato, aumentando di volta in volta il TTL in modo da arrivare fino all'ultimo router sul percorso verso la destinazione.
Esiste anche una versione di ICMP per IPv6.

DHCPv6
Il protcollo ha lo stesso funzionamento e scopo di quello IPv4 ma vengono introdotti alcuni cambiamenti:
Discovery: manda un messaggio Multicast per sollecitare i server DHCP a rispondere. In tal modo il messaggio non viene mandato a tutte le macchine ma solo ai server. Il mittente non è identificato dall'IP 0.0.0.0 ma ha un indirizzo local link valido.
Offer: grazie alla presenza dell'indirizzo local link la risposta è unicast.
Selection(Request): la scelta dell'indirizzo IP viene nuovamente mandata in multicast ai server.
ACK(Risposta): viene mandata una conferma dei dati tramite unicast.

Algoritmi di Routing
La rete può essere rappresentata sotto forma di un grafo pesato i nodi sono i router mentre gli archi sono i collegamenti fisici.
Gli algoritmi di routing vanno a modificare ed aggiornare le tabelle di routing, e quindi il processo di instradamento dei pacchetti lungo la rete.
Gli algoritmi di routing possono essere suddivisi in 2 categorie:
Statici
Viene effettuata una 'fotografia' del grafo in un dato istante dalla quale partire per decidere i percorsi di routing. L'informazione calcolata viene poi distribuita a tutti i nodi. Questo approccio risulta stabile, ma poco propenso a possibili variazioni di rete.
Dinamici
Esamina di volta in volta la situazione dei pesi per poter prendere la decisione migliore in quell'istante di tempo. Questo approcciò riesce ad ottenere dei risultati migliore, soprattutto nel caso di congestione di rete. Monitorare i cambiamenti ed

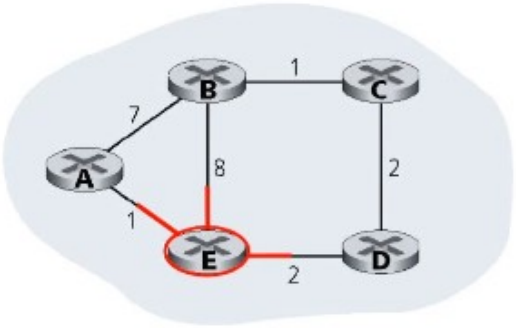
inoltrare i nuovi percorsi potrebbe appesantire la rete e causare congestione se fatto troppo di frequente, per tale motivo, risulta molto più pesante e costoso.
In base a come le informazioni vengono scambiate gli algoritmi si dividono poi in:
Locali
Ogni nodo prende delle decisioni sulla base delle informazioni scambiate con i nodi vicini in maniere autonoma. Questo causa diversi problemi(ad esempio un pacchetto che rimbalza all'infinito in un ciclo vizioso).
Globali: ogni nodo del grafo scambia informazioni con tutti gli altri nodi per prendere le proprie decisioni. Questo metodo è si efficace, ma poco efficiente in termini di prestazioni, per la quantità di informazioni che il nodo deve ricevere.

Flooding
Un algoritmo quasi perfetto che riesce a trovare la strada migliore nel minor tempo possibile e senza aver bisogno di particolari informazioni.
Il nodo mittente fa una copia del pacchetto da spedire e lo manda a tutti i nodi a cui è connesso. Questi a loro volta faranno una copia e lo manderanno a tutti i nodi cui sono connessi. In questo modo vengono generati pacchetti duplicati i quali con tutta probabilità ritorneranno anche al mittente che a sua volta li ri-inoltrerà sui suoi collegamenti in una seconda volta.
In un numero di passi che coincide col diametro del grafo, otteniamo tutti i possibili percorsi disponibili e quindi il percorso migliore. Il problema sta nel bloccare il fooding prima che arrivi ad inondare e congestionare la rete di pacchetti duplicati.

Primo metodo per il controllo del flooding:
Settiamo un HOP limit(simil TTL) pari o poco superiore al diametro del grafo, in modo da bloccare il flooding quando un pacchetto ha fatto troppi salti.

Secondo metodo per controllare il flooding:
Consiste nel far capire alla destinazione che i pacchetti che possono arrivargli sono tutte copie, ad esempio mettendo un ID al pacchetto, in modo da diminuire il numero di pacchetti in circolazione. Il flooding verrebbe bloccato con un numero di salti superiore di 1 al diametro del grafo. In tal modo si avrebbe solo un piccolo e breve picco di congestione. Tale approccio è però molto costoso in termini di memoria e tempo di elaborazione in quanto prevede che ogni nodo abbia una tabella su cui sono registrati i nodi inoltrati in modo da confrontare ogni nodo che arriva con quelli già presenti nella tabella(per riconoscere i duplicati). Funziona solo se il flooding viene usato di rado.

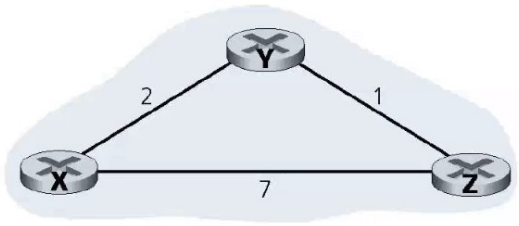
Distance Vectors
Nelle prime implementazioni di internet si pensò ad un approccio locale e statico(in quanto non teneva conto dell’evoluzione dei link).
L’algoritmo distance vectors si basa sull’algoritmo di Bellman-Ford. I costi minimi per i percorsi sono correlati dalla formula: $dx(y)=\min\{c(x,v) + dv(y)\}$ dove $dx(y)$ è il costo del percorso minimo dal nodo x al nodo y. Tale formula viene calcolata per tutti i vicini di x.



costo verso la destinazione via

destinazione $D^E()$	A	B	D
A	1	14	5
B	7	8	5
C	6	9	4
D	4	11	2

Considerando il nodo E del grafo, si costruisce una tabella(solitamente rettangolare) dove ci sono tante colonne quante sono le possibili uscite e tante righe quanti sono i nodi presenti nella rete(ad eccezione del nodo stesso). Per ogni riga, si cerca di arrivare al nodo specificato dalla riga,passando per quello specificato dalla colonna, partendo da E. Ad esempio per andare da E ad A passando per A basta percorrere il collegamento diretto tra E ed A. Per andare da E ad A passando per D, percorriamo il collegamento diretto fra E e D, poi torniamo indietro ed infine ci dirigiamo verso A.



Destinazione / Uscita

	X	Y	Z
Y	∞	∞	
Z	∞	∞	

	X	Y	Z
Y	∞	∞	
Z	∞	∞	

	X	Y	Z
Y	2	∞	
Z	∞	7	

	Y	X	Z
X	∞	∞	
Z	∞	∞	

	Y	X	Z
X	2	∞	
Z	∞	1	

	Z	X	Y
X	∞	∞	
Y	∞	∞	

	Z	X	Y
X	7	∞	
Y	∞	1	

La costruzione della tabella procede inizialmente ponendo tutti i valori ad infinito. Inizialmente vengono considerati solo i collegamenti diretti: partendo da X si arriva al nodo Y passando per Y con costo 2 mentre si arriva a Z passando per Z con costo 7 e così via partendo da Y e da Z.

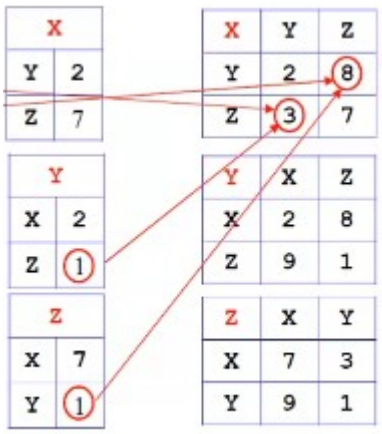
Successivamente i nodi adiacenti comunicano tra loro una versione sintetizzata della propria tabella, ovvero un vettore con i pesi migliori che il nodo stesso conosce verso gli altri nodi. Nel nostro esempio :

Y	
X	2
Z	1

X	
Y	2
Z	7

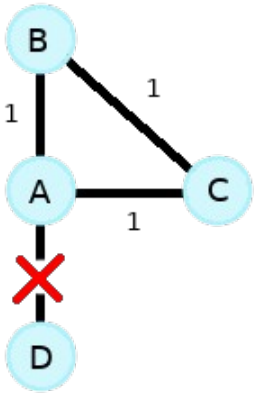
Z	
X	7
Y	1

Ottenute le informazioni, e sapendo ad esempio che Y riesce a raggiungere Z con un costo di 1, il nodo X può aggiornare la propria tabella: il costo per arrivare da X a Z è pari al costo per arrivare da X a Y(2) + il costo per arrivare da Y a Z(1) per un totale di 3.



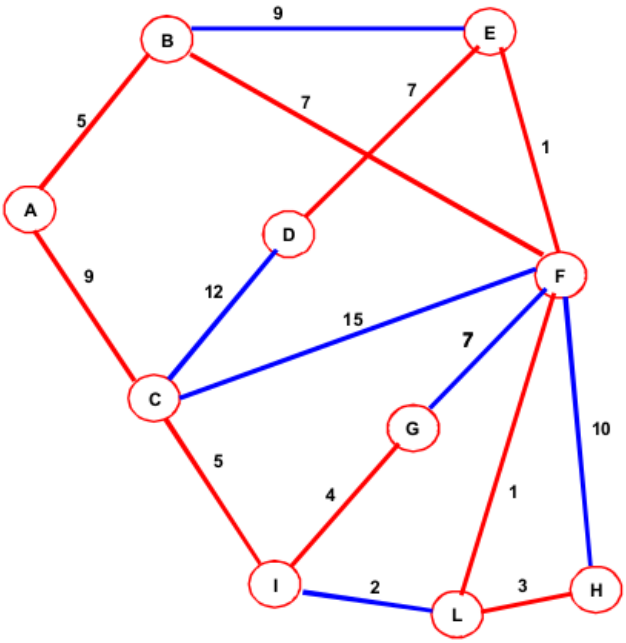
A questo punto tutti e tre i nodi, dopo aver aggiornato le proprie tabelle, generano un nuovo vettore con i pesi migliori.
L'algoritmo si ferma quando si arriva alla situazione migliore.
Un problema di questo algoritmo è dato dal fatto che i nodi si preoccupano solo di andare a trovare il percorso migliore, non curanti del fatto che stanno implicitamente dirigendo tutto il traffico verso il nodo Y: l'algoritmo non tiene conto delle conseguenze future sullo stato della rete.
Nel caso di miglioramenti nel peso di alcuni nodi, l'algoritmo si adegua in pochi passi. Al contrario in caso di peggioramenti(il peso di un arco aumenta), i nodi cominceranno a cambiare i propri vettori introducendo però(nelle fasi intermedie) dei valori falsi, che si basano ancora sulle stime precedenti. Dopo un certo numero di iterazioni, il sistema converge.
Potrebbero però presentarsi delle situazioni in cui il numero di passi da effettuare affinchè il sistema converga sia talmente alto da arrivare ad una situazione definita come Conteggio all'infinito. In questo caso la convergenza non dipende dal diametro del grafo bensì dalla combinazione dei pesi degli archi.

Poisoned Reverse
Una possibile soluzione al problema prevede che se un nodo Z deve comunicare ad un nodo Y la sua distanza verso un altro nodo X allora tale distanza avrà valore infinito se il percorso migliore da Z verso X passa per Y. Questo perchè non ha senso comunicare ad un nodo un percorso se quest'ultimo passa per il nodo stesso.
Z quindi, sta di fatto mentendo al fine di bloccare un'eventuale informazione falsa.



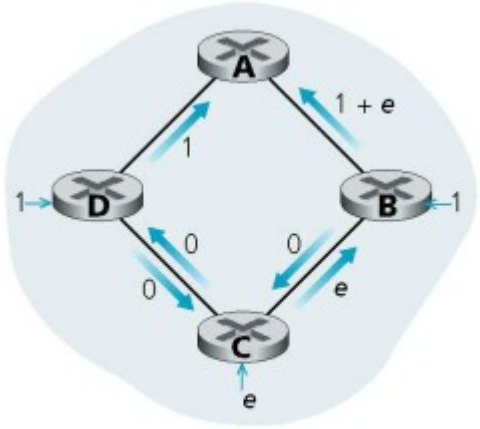
Esiste però una situazione in cui neanche la Poisoned reverse funziona.
A dice a B e C che D è irraggiungibile.
B fa un nuovo percorso verso D attraverso C:
DB(D) = C(B,C)+DC(D)=1+2 = 3.B ora dice a C che D è irraggiungibile a causa della poisoned, mentre dice ad A che ha un percorso verso D di costo 3. A adesso calcola un nuovo percorso attraverso B e dice a C che D è raggiungibile sempre attraverso B.

Algoritmo Link-State
Il Distance vector era stato pensato per gestire grafi in cui i pesi non variavano spesso ma soprattutto senza tenere conto dello stato della rete. L'algoritmo Link State mira a risolvere questi problemi cercando di realizzare una fotografia completa del grafo.
Tale algoritmo si basa a sua volta sull'algoritmo di Dijkstra e per tale motivo non può essere applicato ad un grafo contenente cicli di peso negativo.



A
B = 5 A B
C = 9 A C
F = 12 A B F
L = 13 A B F L
E = 13 A B F E
I = 14 A C I
H = 16 A B F L H
G = 18 A C I G
D = 20 A B F E D

Partendo da un nodo si vanno a trovare tutti i percorsi minimi verso i nodi ad esso adiacenti; successivamente si aggiunge all'insieme dei nodi percorribili il nodo il cui percorso alla sorgente ha costo minimo rispetto agli altri; si itera il procedimento andando ad aggiungere a mano a mano tutti gli altri nodi, trovando così anche i percorsi minimi.



a. Routing iniziale

Nonostante l'algoritmo funzioni abbastanza bene anche lui ha dei problemi. In una situazione come quella in figura, partendo dal nodo C come sorgente, essendo i percorsi verso B e D equivalenti,l'algoritmo sceglierà uno dei due in maniera arbitraria, tramite cui arrivare al nodo A. Se sceglie B, il traffico sul link ignorato potrebbe arrivare a zero, All'analisi successiva, la prossima 'fotografia' del grafo mostrerà il link che collega C a B sovraccarico, mentre quello che collega C a D libero: l'algoritmo agirà di conseguenza spostando tutto il traffico in direzione del nodo D. Tale procedimento verrà ripetuto ad ogni iterazione causando delle oscillazioni.

Confronto tra DV-LS

Il Distance Vector è un algoritmo che lavora a livello locale mentre il link state è un algoritmo globale: un'unica macchina prende le decisioni su come trovare il percorso migliore in base ad una fotografia dell'intero sistema. Per tale motivo DV si basa su messaggi dei nodi adiacenti, mentre LS si basa sullo stato dell'intera rete ed ha bisogno di messaggi broadcast, inoltre per DV i messaggi vengono scambiati solo in caso di variazioni, mentre LS ha bisogno di un continuo scambio di informazioni. Dato che in LS ogni nodo ha tutte le informazioni sulla rete, a differenza di DV, può capire o meno se alcuni nodi maliziosi stiano cercando di imbrogliare gli altri con delle informazioni errate.

RIP-Routing Information Protocol.
Protocollo di routing di tipo distance vector che impiega il numero di HOP(numero di salti effettuati da un nodo all'altro)come metrica. Adottando un limite massimo di HOP viene così evitato il routing loop.
Ad oggi poco utilizzato.

RIPv1
Il numero massimo di HOP è 15. Questo perchè il numero massimo di HOP pone anche un limite al diametro del grafo, e per un grafo con un diametro troppo ampio, la convergenza dell'algoritmo risulterebbe troppo lenta.
Le tabelle di routing vengono scambiate ogni 30 secondi. Se un percorso non viene aggiornato per 180 secondi, la sua distanza viene posta ad infinito, se trascorrono altri 120 secondi, ovvero allo scadere del garbage-collection timer, il nodo viene eliminato dalla tabella di routing.

Messaggi RIPv1
Utilizza due tipi di messaggi:
REQUEST: per chiedere info ai nodi adiacenti;
RESPONSE per inviare info di routing
Una tabella di routing RIP contiene:

1. Indirizzo di destinazione.
2. Distanza dalla destinazione (in HOP).
3. Next HOP: router adiacente cui inviare i pacchetti.
4. Timeout.
5. Garbage-collection timer.

La lunghezza di tali messaggi può arrivare fino a 512 byte ed inoltre non c'è alcune informazione circa la maschera della sottorete(sono indirizzi Classful).

RIPv2
Include il trasporto di informazioni relative alla maschera di sottorete supportando così il CIDR. Implementa un'autenticazione dei messaggi per garantire sicurezza, lo split horizon per prevenire routing loop ed il poison reverse per evitare lo scambio di informazioni false.

RIPng
Estensione di RIPv2 che permette di supportare IPv6

Autonomous System(AS)
Si definisce autonomous system un gruppo di router e reti sotto il controllo di un'autorità amministrativa come un ISP. Quando vogliamo comunicare all'interno di un AS viene usato un Interior Gateway Protocol(IGP) come RIP, OSPF ed altri, mentre per comunicare all'esterno dell'AS uso un Exterior Gateway Protocol(EGP) come BGP. Quando voglio comunicare con un dispositivo esterno al mio AS ciò che

mi interessa sapere è quali sono i punti di uscita(Gateaway), verso gli altri AS, più efficienti per raggiungere la mia destinazione.
In tal senso, fare una fotografia della rete per un algoritmo come LS, si limita a fotografare il proprio AS e non l'intero Internet, cosa che sarebbe impossibile.
Spesso i protocolli EGP tengono conto, oltre che del percorso migliore, anche di vari fattori come ad esempio i rapporti politico/economici che intercorrono fra i vari paesi.

OSPF(Open short path first)
Implementa Link State(quindi Dijkstra)e può essere applicato anche su reti di dimensioni notevoli.
Ogni nodo manda un messaggio ad un nodo vicino. In base al tempo di risposta può essere determinata la bontà del link. Questo tempo tiene conto sia della capacità del link, ma anche del traffico presente sul link in quel momento.
Tale misura viene effettuata sia dal nodo mittente che dal destinatario, prevenendo in tal modo eventuali imbrogli.
Ogni nodo prepara poi una sua tabella comunicando chi siano i suoi vicini e quali siano i costi dei link verso di essi.
Questa tabella viene mandata in Flooding, ponendo un ID all'inizio della tabella in modo tale che il messaggio non torni indietro e non faccia circoli viziosi.
Generalmente abbiamo tre fasi:
HELLO: scoperta e verifica dei vicini
EXCHANGE: sincronizzazione iniziale del Database
FLOODING: aggiornamento del Database

Per bloccare il Flooding prima di congestionare la rete si usano gli LSA: ogni router ha un database composto da Link State Records e vengono aggiornati e sincronizzati tramite i Link State Advertisement.
Gli LSA sono emessi:

1. Quando un router riscontra un nuovo router adiacente
2. Quando un router o link si guasta
3. Quando il costo di un link cambia
4. Periodicamente ogni 30 minuti

L'algoritmo in generale non presenta problemi se non durante le fasi di aggiornamento dei database ma solo in casi eccezionali potrebbe verificarsi un loop infinito per un messaggio, il quale verrà scartato una volta esaurito il suo TTL.

BGP
Protocollo di routing tra AS e quindi esterno alla propria rete di appartenenza, il cui scopo principale è quello di realizzare una sorta di ponti di comunicazione fra AS.

Firewall
Esistono due tipologie di firewall: software e hardware.
Un firewall software non è altro che la composizione di una serie di regole che vengono date ad un unico dispositivo per fare transitare o meno pacchetti dall'esterno verso l'interno e viceversa. Il problema è che se un attaccante riesce tramite una

backdoor od un bug, ad accedere al sistema, potrebbe cambiare le regole del firewall e controllare tutto il traffico.
Mentre un firewall software è un processo che gira all'interno del SO, quindi se il SO è compromesso lo è anche il firewall, un firewall hardware è un device(Application Gateway) fisico che filtra il traffico in arrivo su un host. Tale device è posto fra il router interno alla rete ed il router esterno alla rete. Oltre alle opzioni di filtraggio è possibile inserire altre regole che permettono di aumentare i fattori di sicurezza.

DMZ
Zona Delimitarizzata. Zona al confine tra due stati in rapporti tesi, ovvero una zona cuscinetto dove non c'è presenza di forze militari.
Dal punto di vista informatico è una rete posizionata tra la rete interna ed internet ,non protetta da firewall ma che appartiene ancora all'amministratore di rete. Lo scopo di questa rete è quello di aggiungere un ulteriore livello di sicurezza ad una rete locale. Gli host posti su tale rete infatti sono quelli che forniscono servizi sia ad utenti interni che esterni alla rete locale, ossia quelli potenzialmente più vulnerabili ad attacchi esterni. Tale rete permette a chi si connette dall'esterno di usufruire soltanto dei servizi degli host posti in essa, impedendo così l'accesso alla rete interna. Questo significa che se un attaccante riesce a prendere il controllo di un host nella DMZ egli si ritrova in un vicolo cieco oltre il quale non potrà andare.

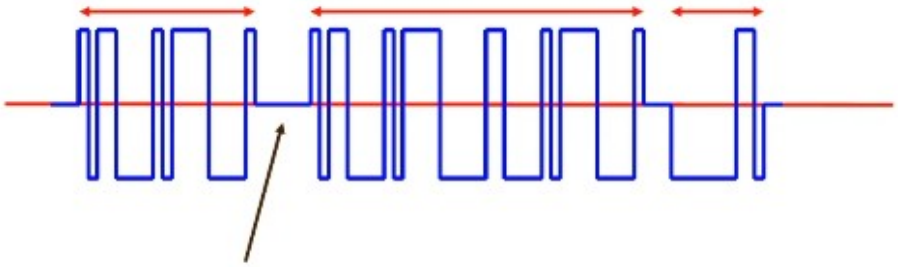
Data Link Layer1
Si occupa di fornire al livello di rete un servizio di trasmissione di flussi di bit.
I suoi compiti principali sono:

- Framing: raggruppare i bit provenienti dal livello fisico in modo da formare pacchetti.
- Mac Sublayer: gestire l'accesso al mezzo fisico specificando le regole con cui immettere i frame nel collegamenti(molto semplificato o assente nelle connessioni punto a punto).
- Fornire un recapito affidabile se richiesto.
- Gestire gli errori dovuti al canale di trasmissione.
- Regolare il flusso dei dati tra sorgente e destinazione.

Il Data-Link-Layer è fondamentale in quanto è quel livello che permette lo scambio reale di informazioni tra macchine adiacenti.
In alcune tipologie di reti è diviso in due parti ben distinte, il Medium Access COntrol Sublayer(strato di software) ed il Logical Link Control che è il DLL.

Framing
In una fibra ottica possiamo immaginare di avere due livelli, uno in cui passa la luce ed uno in cui non c'è. Le fasi di assenza di luce corrispondono a sequenze di zeri: il problema è quello di capire se in quelle fasce c'è un'assenza di comunicazione oppure se rappresentano una sequenza molto lunga di bit a 0. Interpretare tali segnali in un modo piuttosto che in un altro fà tutta la differenza del mondo al livello comunicativo in quanto altera di molto il segnale.

Un sistema banale per risolvere il tutto prevede di avere tre livelli, in cui usiamo due

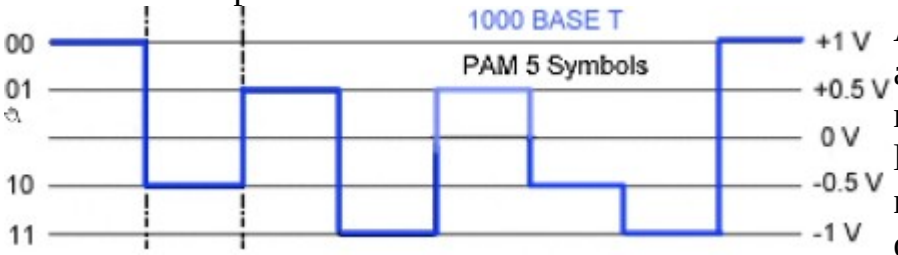


livelli per segnalare sequenze di 1 e di 0, ed un ulteriore terzo livello per segnalare un'assenza di comunicazione: Stiamo di fatto sprecando un livello che potremmo invece usare per trasportare più comunicazione, ed è quello che viene chiamato ridondanza nella comunicazione.

La ridondanza quindi non è altro che uno spreco di bit che risulta però essere necessario alla comunicazione

Per limitare la ridondanza è possibile usare un sistema basato sempre su tre livelli nel quale se dopo un bit c'è 1 allora si ha un cambio di livello, altrimenti se ci sono bit 0 mantengo lo stesso livello->non c'è quindi un livello che viene usato esclusivamente per segnalare assenza di comunicazione.

La stessa cosa può essere fatta con 5 livelli:



Associo il livello 0V all'assenza di segnale, mentre ad ogni altro livello utilizzabile associo non un bit ma una coppia di bit.

Codifica 4B5B

Nome	4B	5B	Descrizione
0	0000	11110	hex data 0
1	0001	01001	hex data 1
2	0010	10100	hex data 2
3	0011	10101	hex data 3
4	0100	01010	hex data 4
5	0101	01011	hex data 5
6	0110	01110	hex data 6
7	0111	01111	hex data 7
8	1000	10010	hex data 8
9	1001	10011	hex data 9
A	1010	10110	hex data A
B	1011	10111	hex data B
C	1100	11010	hex data C
D	1101	11011	hex data D
E	1110	11100	hex data E
F	1111	11101	hex data F

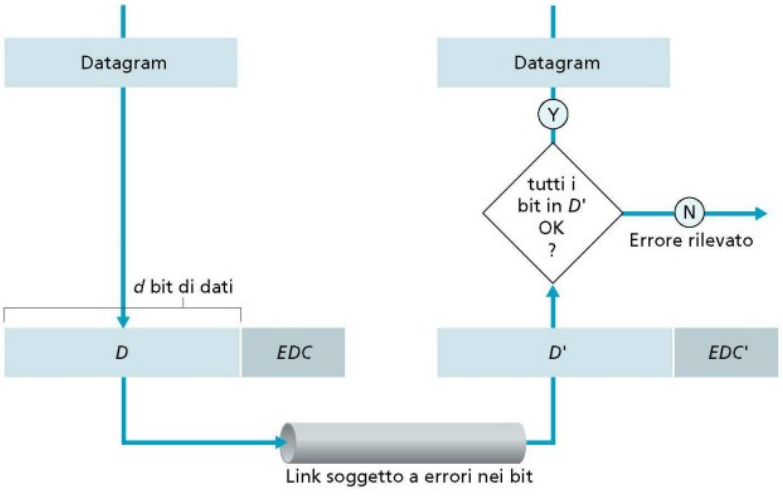
L'idea è quella di inserire uno strato software per capire quando termina una sequenza di trasmissione. A livello fisico abbiamo sequenze di 5 bit mentre a livello logico abbiamo sequenze di 4 bit codificate secondo una tabella prestabilita. Ad esempio se vogliamo trasmettere il valore 7 codificato con 0111, la tabella mi dice che devo trasmette 01111, quindi in pratica ogni 4 bit introduco una ridondanza di un bit. Ancora, per trasmettere 0 ossia 0000, una volta convertito trasmetteremo 11110. In questo modo non ci saranno mai sequenze con più di 3 zero consecutivi(e nemmeno con 8 bit ad 1 consecutivi): la conseguenza di tale approccio è che se abbiamo molti zeri consecutivi sicuramente non c'è comunicazione.
La mappatura non è quindi casuale ma è stata fatta con lo scopo di evitare le sequenze ripetute.

Con 4B5B usiamo dunque un sistema di comunicazione a due livelli risolvendo il problema del framing, sprecando però 1/5 della banda, che è comunque meglio di andare ad aggiungere un solo livello per caratterizzare la ‘non comunicazione’.
Un obbiettivo secondario della tabella è quello di andare a sincronizzare mittente e destinatario(come non l’ho capito).

Codifica 8B10B
Codifica simile a 4B5B ma elettricamente neutra(quindi il numero di bit a 1 è pari al numero di bit a 0). Tale codifica viene usata in vari standard quali SATA, USB 3.0 e HDMI.
L’idea è quella di usare 8 bit di dati con 10 bit di segnale, ottenendo sempre lo stesso spreco di ridondanza.
Il fatto che la codifica sia neutra impedisce che ci sia un trasferimento di energia tra un dispositivo e l’altro.
L’obbiettivo è quello di garantire un numero di bit uguale(0 e 1) a meno di 2.
Gli 8 bit sono suddivisi in un primo gruppo di 5b ed un secondo di 3b i quali vengono trasformati rispettivamente in un gruppo da 6bit e da 4bit.
Tali sequenze non sono però fisse, in particolare la seconda dipende dal risultato dell’operazione precedente.
Alcune codifiche inoltre non sono uniche: se al passo precedente avevo un eccesso di bit ad 1, allora al passo successivo uso la codifica con eccesso di bit a 0. Le codifiche che non hanno alternative sono uniche proprio perchè già equilibrate.

Rilevazione e Correzione di errori

La rilevazione consiste nell’individuare la presenza di errori nella trasmissione in un frame, senza però correggerlo. Se viene rilevato un errore infatti il DLL si limita a scartare quel frame.
La correzione consente sia di rilevarlo che di correggerlo.
Entrambi i metodi sono basati sulla presenza di ridondanza e sulla possibilità di falsi positivi.



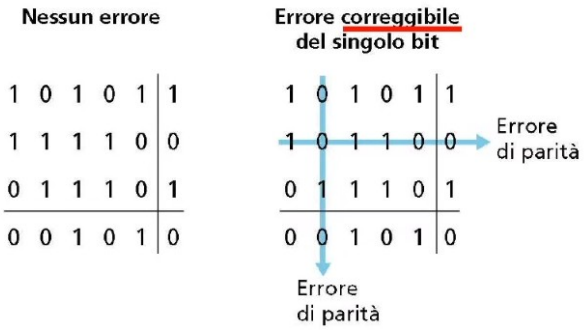
Supponiamo di avere una sequenza di dati D formata da un numero ‘d’ di bit. A tale sequenza aggiungiamo un’altra sequenza detta EDC(error detection and correction) calcolata tramite una funzione su D.
Il nodo ricevente deve determinare se D’ coincide con D potendo fare affidamento solo su D’ ed ECD’. Per farlo esistono diverse tecniche.

Controllo di Parità

Il modo più semplice è il controllo di parità ad 1 bit. Consiste nell'aggiungere un bit di parità a fine sequenza in modo da far risultare il numero di bit ad 1 un numero pari. Per esempio 01101110 diventa 011011101. Ciò che deve fare il ricevente è contare il numero di bit a 1 tra quelli ricevuti. Se trova un numero dispari di bit ad 1 sa che si è verificato almeno un errore: in particolare il numero di errori sarà dispari.
Il problema di questo approccio è che un numero pari di errori non viene rilevato. Inoltre più aumenta la lunghezza più aumenta la probabilità che ci sia un numero pari di errori.

Parità bidimensionale

Secondo questo modello le sequenze vengono poste su una riga a formare una matrice. Per ogni riga e per ogni colonna si va a verificare la parità. Se in un riga rivelo un errore di parità all'ora ci sarà sicuramente un errore all'interno della sequenza. Inoltre se trovo una colonna in cui si verifica un errore di parità, il punto di intersezione tra la riga e la colonna nelle quali si è verificato un errore, indica il luogo in cui risiede quest'ultimo. Il sistema non è però adatto alla situazione in cui si presentano più errori.



CRC(Codice di Ridondanza Ciclica)

$D \cdot 2^r$ XOR R Formula matematica

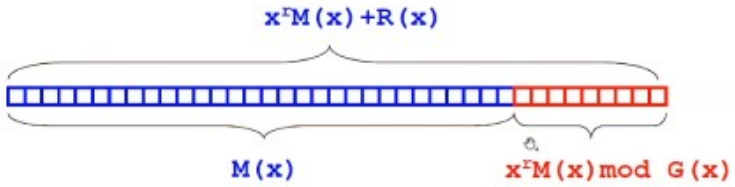
1 0 0 1 0 1 1 1

$x^7+ \quad x^4+ \quad x^2+ \; x^1+ \; 1$

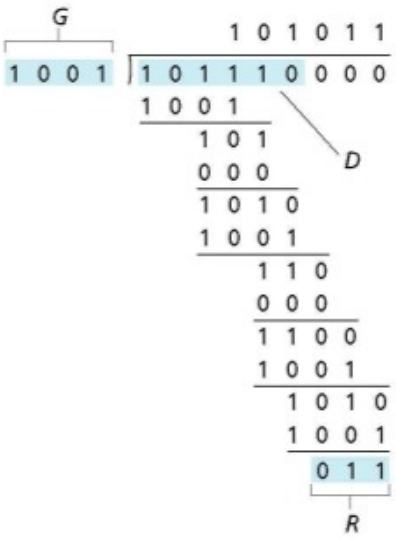
Tecnica di rilevazione degli errori molto più affidabile ed efficiente rispetto al controllo della parità. L'idea è quella di cercare una sequenza di operazioni matematiche per cui il risultato R viene univocamente e correttamente generato dai dati D della nostra sequenza.

Supponiamo di avere un byte pari a 100010111. Trasformiamo D in un polinomio dove la posizione del singolo bit viene usata come esponente del monomio se c'è 1. Se c'è zero non scriviamo nulla e saltiamo quell'esponente.

10010111 diventa x 7 +_+_+x 4 +_x 2 +x 1 +x 0.
Prendendo arbitrariamente un altro polinomio:
01011011 il quale diventa _ +x 6 +_+x 4 +x 3 +_+x 1 +x 0
Possiamo fare sia la somma che il prodotto:
Somma: x 7 +x 6 +x 3 +x 2
Prodotto: x 13 +x 11 +x 8 +x 7 +x 5 +x 4 +x 0
Possiamo farlo dal punto di vista matematico perché i coefficienti di questo polinomio
sono nel campo Z 2 , che è un campo dove esistono solo 0 e 1, la somma rispecchia la tabella di verità di XOR, e il prodotto è l'operazione AND.
Il fatto che sia un campo ci assicura che ci siano le proprietà commutativa, associativa.



Funzionamento
Sia D una sequenza costituita da d bit e sia G un generatore ossia una sequenza di r+1 bit nota sia al mittente che al destinatario. L'idea è quella di andare ad aggiungere a D, una sequenza R di r bit: quando il destinatario riceve questo pacchetto di m+r bit egli dovrà dividere quest'ultimo per G e verificare il resto. Se pari a 0 allora il pacchetto non è corrotto, altrimenti se diverso da 0, significa che il pacchetto presenta degli errori e quindi viene scartato. La sequenza di R bit viene calcolata andando ad aggiungere a D, r bit di ridondanza, che nella pratica consiste nell'andare ad aggiungere r zeri come bit meno significativi. Successivamente si va a dividere tale sequenza per G: il resto di questa divisione costituisce R e verrà aggiunto a D nel pacchetto. Il generatore G dipende dalla versione di CRC usata. Se usiamo CRC-32 ad esempio avremo un generatore di 32 bit.



Nota Bene: CRC permette solo di rilevare errori, ma non correggerli. In caso di errore ci si limita a scartare il pacchetto.

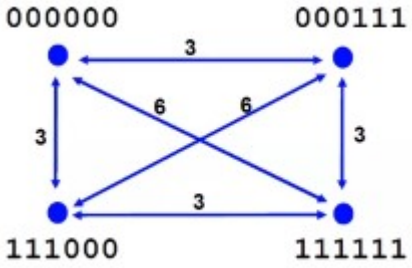
Distanza di Hamming e Correzione di errori

Parliamo di distanza tra due codeword andando a capire di quanti bit differiscono(si fa uno XOR: se i bit sono uguali da 0 altrimenti da 1). Ad esempio la distanza tra le due codeword 10001100 XOR 11000100 = 01001000 è pari a 2.

Questo significa che per passare da una word all'altra dobbiamo cambiare 2 bit.

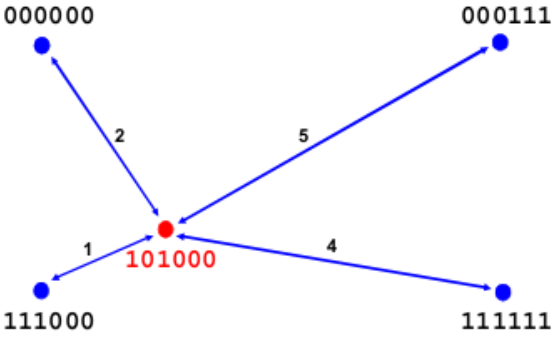
Quindi la distanza può anche essere espressa come il numero di variazioni per passare da una codeword ad un'altra.

Dato un insieme di codeword definite, è possibile parlare di vocabolario.



Considerando le codeword 000000 000111 111000 111111 esse formano il vocabolario in figura che può formare fino a 64 possibili combinazioni(codeword). Diremo che il vocabolario è composto da 4 codeword e la distanza è pari alla distanza minima tra le distanze calcolate sulle 4 codeword. Quindi nel nostro esempio il vocavolario ha distanza pari a 3.

Se il mittente può trasmettere solo queste 4 codeword, tutte le altre possibili combinazioni devono essere considerate come degli errori. Per ogni codeword errata ricevuta possiamo calcolare la distanza delle codeword 'legali'. Ad esempio



La codeword 101000 dista 1 dalla codeword 111000 mentre ha distanza maggiore rispetto alle altre.

Se una codeword ha un solo errore allora essa ha sicuramente una distanza pari ad 1 rispetto ad una delle codeword legali. Se ipotizziamo che la probabilità che si verifichi un errore è molto maggiore rispetto alla probabilità che se ne verifichino più di uno possiamo affermare che in un vocabolario di 4 codeword con distanza pari a 3, presa una codeword errata, nel caso peggiore sarà a distanza 1 da una codeword lecita e a distanza 2 da un'altra lecita. In sostanza se abbiamo un vocabolario a distanza=3 possiamo correggere gli errori su singoli bit.

Se vogliamo correggere un numero più grande di errori allora abbiamo bisogno di un vocabolario di distanza d=2e+1. Se invece vogliamo RILEVARE una quantità di errori è necessario un vocabolario con d=e+1.

Per procedere alla correzione di errori singoli è quindi necessario che:

1. La probabilità P(e=1) >>P(e=2) >> P(e=3).. dove P(x) è la probabilità che si verifichino x errori su quella codeword.
2. Definire un vocabolario con distanza minima d=3

Esempio: codewords da 10 bit(2 dati, 8 ridondanza)

0000000000 0000011111 1111100000 1111111111

Parole valide: 4

Distanza: 5

Correzione errori: 2 bit

Rilevazione errori: 4 bit

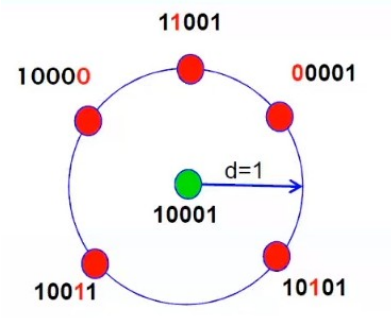
È possibile selezionare un altro vocabolario con parole da 10 bit ma con più parole, tale che la distanza rimane 5?

A 0000000000
B 0000011111
C 0011100011
D 0011111100
E 1100100101
F 1100111010
G 1111000110
H 1111011001

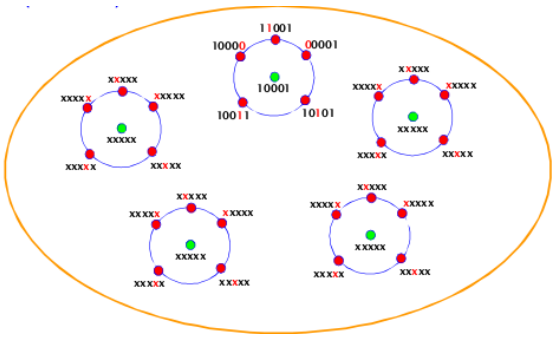
	A	B	C	D	E	F	G	H
A	0	5	5	6	5	6	6	7
B	5	0	6	5	6	5	7	6
C	5	6	0	5	6	7	5	6
D	6	5	5	0	7	6	6	5
E	5	6	6	7	0	5	5	6
F	6	5	7	6	5	0	6	5
G	6	7	5	6	5	6	0	5
H	7	6	6	5	6	5	5	0

Parole valide: 8
Distanza: 5
Bit di dati 3
Bit di controllo(ridondanza) 7
Combinazioni possibili 2^10
Sulla matrice sono riportate le distanze tra una codeword e l'altra. Possiamo fare di meglio?
Supponendo di avere 3 bit di dati, quanta ridondanza dobbiamo aggiungere per poter correggere su base probabilistica? Come costruire il vocabolario?

Supponiamo di avere un vocabolario con m bit dati e r bit controllo/ridondanza.
Quindi ci saranno n=m+r bit per ogni codeword.
Le combinazioni possibili con n bit sono 2^n di cui solo 2^m sono valide.

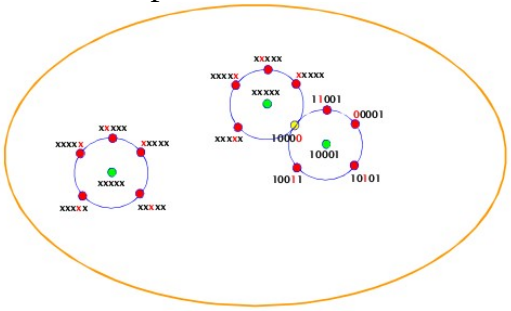


Consideriamo una codeword valida: 10001: cambiando un bit alla volta possiamo scrivere n codeword tutte sicuramente errate ed a distanza 1 da quella lecita, andando così a descrivere un cerchio di raggio d=1



Estendendo questo ragionamento a tutte le codeword valide avremo un totale di 2^m cerchi ognuno dei quali contiene n+1 codeword
Il totale dovrà essere minore o uguale a 2^n.

Affinchè avvenga questo, è necessario che non ci siano codeword che sono a distanza 1 da due altre codeword(legali) contemporaneamente.

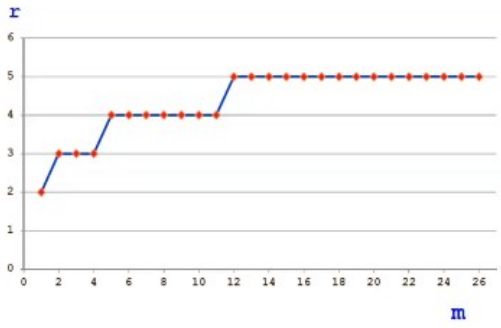


Se il vocabolario ha distanza 3 allora i cerchietti sono tutti separati e quindi la loro unione è pari alla somma.

Con d=3 possiamo scrivere con certezza che $(n+1)2^m \leq 2^n$
Da cui otteniamo che $(m+r+1)2^m \leq 2^{m+r} \rightarrow (m+r+1) \leq 2^r \rightarrow m+1 \leq 2^r - r$

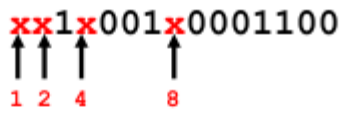
m	r	n
1	2	3
2	3	5
3	3	6
4	3	7
5	4	9
6	4	10
7	4	11
8	4	12
9	4	13
10	4	14
11	4	15
12	5	17
13	5	18

m	r	n
14	5	19
15	5	20
16	5	21
17	5	22
18	5	23
19	5	24
20	5	25
21	5	26
22	5	27
23	5	28
24	5	29
25	5	30
26	5	31
27	6	33



In particolare osservando i valori di m ed r per cui vale l'espressione, notiamo che quando arriviamo al valore in cui le due quantità sono uguali invece che minore stretto, allora viene incrementato r: ad esempio se m=11 ed r=4 allora (11+4+1)<= 2^4 ovvero 16<=16. Da questo passo sia m che r vengono incrementati passando ad m=12 ed r=5.
Questo fa si che n non assuma mai valori pari a potenze del 2.

Codici di Hamming
L'idea è quella di andare a numerare le posizioni dei bit di una sequenza in ordine crescente da sinistra verso destra, andando a porre nelle posizioni che sono potenze di 2 dei bit di controllo, mentre poniamo i bit di dati nelle posizioni che non sono potenze di 2.



$b_1 = 3 \otimes 5 \otimes 7 \otimes 9 \otimes 11 \otimes 13 \otimes 15$
$b_2 = 3 \otimes 6 \otimes 7 \otimes 10 \otimes 11 \otimes 14 \otimes 15$
$b_4 = 5 \otimes 6 \otimes 7 \otimes 12 \otimes 13 \otimes 14 \otimes 15$
$b_8 = 9 \otimes 10 \otimes 11 \otimes 12 \otimes 13 \otimes 14 \otimes 15$

$b_1 = 1 \otimes 0 \otimes 1 \otimes 0 \otimes 0 \otimes 1 \otimes 0 = 1$
$b_2 = 1 \otimes 0 \otimes 1 \otimes 0 \otimes 0 \otimes 0 \otimes 0 = 0$
$b_4 = 0 \otimes 0 \otimes 1 \otimes 1 \otimes 1 \otimes 0 \otimes 0 = 1$
$b_8 = 0 \otimes 0 \otimes 0 \otimes 1 \otimes 1 \otimes 0 \otimes 0 = 0$

1011001000011100

3 = 1+2
5 = 1 +4
6 = 2+4
7 = 1+2+4
9 = 1 +8
10 = 2 +8
11 = 1+2 +8
12 = 4+8
13 = 1 +4+8
14 = 2+4+8
15 = 1+2+4+8

Supponiamo di avere la stringa 10010001100 e quindi di dover calcolare 4 bit di controllo i quali verranno posizionati nelle prime posizioni con indice una potenza di 2.
In particolare il bit in posizione 1 è pari al bit in posizione 3 XOR il bit in posizione 5 XOR e così via. Come mostra la figura a destra ogni posizione viene ottenuta come somma di potenze del due. Se in quella somma non compare la posizione del bit di controllo calcolato allora, il bit con quella posizione non compare nello XOR.
Ad esempio nel calcolare il valore del bit di controllo in posizione 2 non compare il bit di posizione 9 perché non è presente il 2 nella sua somma.

La sequenza risultante è 101100100001100.
Inseriamo ora un errore in un bit qualsiasi tale che la sequenza che arriva al destinatario sia: 101100100101100
Il destinatario esegue lo stesso calcolo con gli XOR di prima al fine di controllare tutti i bit di controllo. In questo caso i bit in posizione 2 e 8 risultanti dagli XOR non coincidono con i bit nella sequenza arrivata: nella sequenza valgono 0 e 0 mentre tramite il calcolo essi valgono 1 e 1. Per trovare l'errore basta consultare la figura a destra andando a vedere per quale posizione vengono coinvolte SOLO le posizioni 2 e 8: la 10. Per cui il bit 10 è quello errato.

Se più errori fossero concentrati in una sequenza:

```
xx1x110xx0x011xx0x110xx1x000xx0x011xx1x100xx0x001xx0x11
0xx1x011xx1x110xx1x001xx0x011
```

```
xx1x110
xx0x011
xx0x110
xx1x000
xx0x011
xx1x100
xx0x001
xx0x110
xx1x011
xx1x110
xx1x001
xx0x011
```

Procediamo andando a separare le sequenze in gruppi
Trasmettiamo le codeword colonna per colonna.

```
xxxxxxxxxxxxxxxxxxxxxxxx100101001110xxxxxxxxxxxx101001
01010011101001110101001010101 1
```

Se si verificano una serie di errori

```
xxxxxxxxxxxxxxxxxxxxxxxx100101001110xxxxxxxxxxxx101001
01010000101111101010010101011
```

```
xx1x100
xx0x001
xx0x100
xx1x010
xx0x001
xx1x110
xx0x011
xx0x110
xx1x011
xx1x110
xx1x001
xx0x011
```

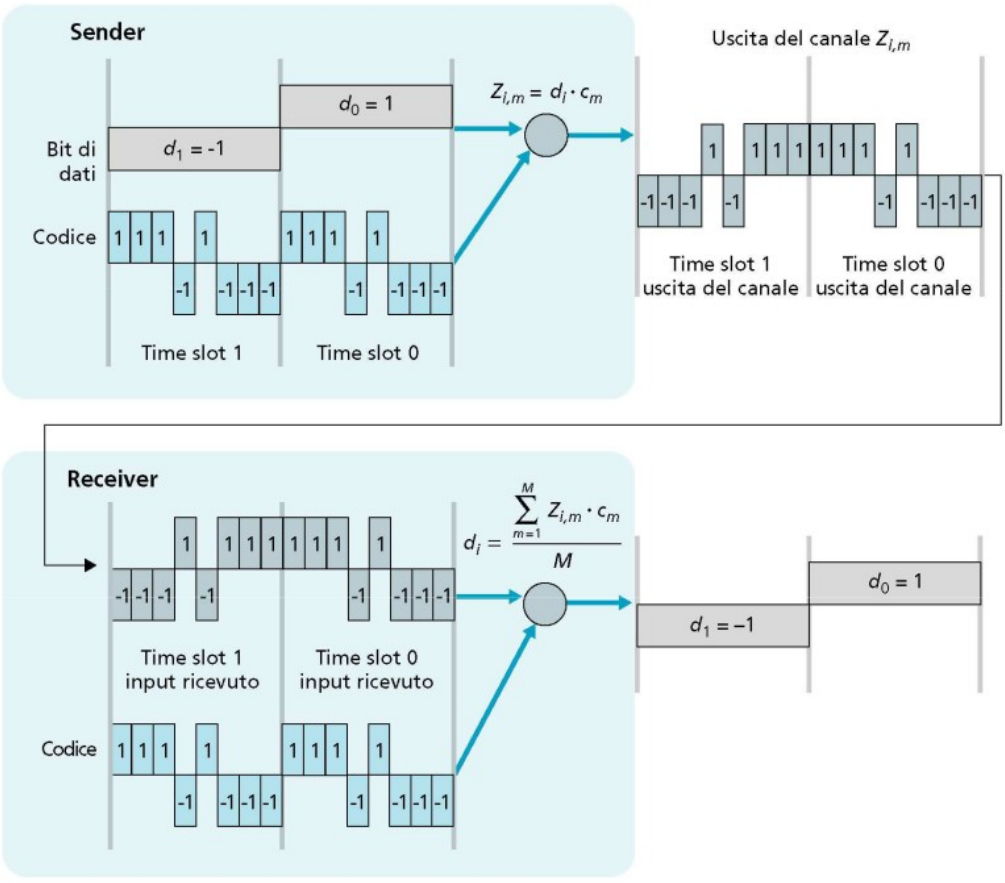
La raffica di errori si va a distribuire su codeword differenti: tali errori possono essere gestiti con i codici di hamming. Lo svantaggio di questo sistema è che per trasmettere in colonna e poi riconvertire andiamo ad introdurre un certo ritardo di trasmissione.

Accesso al link(MAC SubLayer)
Come già detto un altro compito del livello DLL è quello di gestire l'accesso al mezzo fisico. Nei canali punto a punto non è necessario questo tipo di protocollo.
Il problema risiede nei mezzi condivisi dove è più probabile che possano verificarsi collisioni tra segnali.
Il protocollo può essere centralizzato o distribuito: in generale preferiamo avere un protocollo distribuito perché avere una macchina unica comporta avere un unico punto di guasto.

TDMA
Prevede che un host chieda il permesso di parlare e se concesso, parla per uno slot di tempo determinato. L'arbitro stabilisce gli slot(una volta per tutte le comunicazioni a venire)→ in caso di guasto le macchine potranno continuare a comunicare

FDMA
Lo stesso schema può basarsi sulle frequenze. Un esempio banale è comunicare con dei laser colorati diversamente con un filtro ottico che fa passare solo una certa frequenza nei canali. Ogni host usa solo lo slot assegnatogli ma per tutto il tempo che vuole. (Come le radio) se troppo vicine causano interferenza.

CDMA



L'idea consiste nell'andare a marcare ogni bit del segnale con un valore -1 o 1. A tale segnale andiamo ad imprimere un codice standard(sempre costituito da -1 o 1 per ogni bit) tramite una semplice moltiplicazione.
Il ricevente riapplica lo stesso codice per ottenere la sequenza originale.
Se avessimo due mittenti, con due codifiche differenti ed una logica di separazione ben precisa, essi potrebbero trasmettere contemporaneamente sulle stesse frequenze, nello stesso tempo, sovrapponendo i segnali ma senza rovinale l'output finale. Da sottolineare che i codici devono essere tra loro linearmente indipendenti.

Accesso Casuale
Questi protocolli sono distribuiti.

ALOHA
Fu sviluppato in Hawaii per realizzare una rete di comunicazione wireless tra le varie isole dell'arcipelago.
L'idea di ALOHA è di avere una serie di nodi che non sono coordinati. La regola è che si parla quando vuole: i nodi non prestano attenzione al fatto che qualcuno sta usando il canale e non interrompono la loro trasmissione se un altro nodo sta cominciando ad interferire. Si usava un antenna per trasmettere e una per ricevere nelle stazioni periferiche ed un'antenna centrale che faceva da ripetitore. Se una stazione periferica voleva trasmettere verso un'altra, trasmetteva la sua frame verso l'antenna centrale la quale ripeteva la frame su un canale di ripetizione(broadcast). Se nessun altro disturbava il canale allora arrivava correttamente a destinazione, altrimenti in caso di collisione veniva trasmesso il segnale rovinato.
Se sul canale di ritorno riusciva a sentire la sua frame allora essa non era stata rovinata, altrimenti la ritrasmetteva.
Siano G i tentativi al secondo e S il throughput per tempo di frame, allora il throughput massimo si ha per G=0.5 e con S=0.184. Il canale viene usato al 18.4% delle sue potenzialità.

Slotted Aloha
Introduce degli intervalli di tempo chiamati slot in cui poter trasmettere: una stazione può trasmettere solo all'inizio di ogni slot. Tuttavia le collisioni possono ancora manifestarsi nel caso in cui due stazioni decidono di trasmettere all'inizio dello stesso slot oppure nel caso in cui una stazione comincia a trasmettere proprio quando lo slot corrente sta per finire.
Slotted ALOHA ha prestazioni all'incirca doppie rispetto ad ALOHA puro, riducendo anche il numero di collisioni. In generale entrambi i sistemi funzionano con basso traffico.

CSMA
Protocollo che si adegua alle regole della buona educazione.
Rilevamento della portante: se c'è qualcuno che sta già parlando allora aspetto finché non ha concluso(Carrier Sense).
Due macchine bloccate sulla stessa portante iniziano a trasmettere non appena rilevano la fine della trasmissione, disturbandosi tra loro(1-persistent).
Potrei pensare di aspettare un tempo random, dopo la fine della trasmissione, prima dii trasmettere io stesso(non-persistent).
Oppure trasmettere con una certa probabilità p: appena sento che la trasmissione precedente è terminata, se sono fortunato trasmetto, altrimenti sto ancora zitto e aspetto un altro po' prima di ritentare(p-persistent).

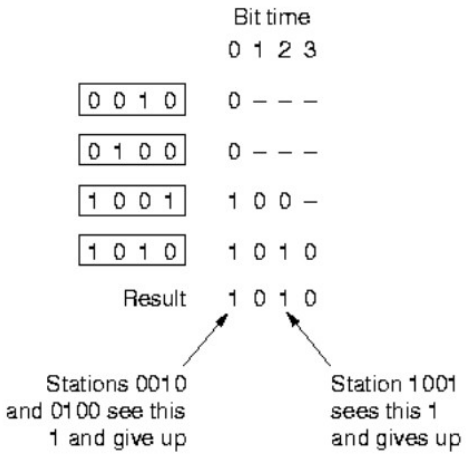
CSMA/CD
Rilevamento della collisione: se mentre trasmetto qualcun'altro comincia a parlare allora smetto di farlo. Trascorso un certo intervallo di tempo provo a trasmettere nuovamente.
A differenza di p-persistent, CSMA/CD non tenta in base ad una probabilità, ma in base ad un tempo casuale. L'intervallo di tempo è casuale proprio per evitare che entrambe le macchine provino a ritrasmettere contemporaneamente. Per calcolare questo tempo viene utilizzato l'algoritmo di binary exponential backoff.
Supponiamo che una scheda di rete tenti di trasmettere e che nel mentre rilevi una collisione. Allora sceglie casualmente un valore K dall'insieme 0 – 2^n – 1 dove n è l'indice della collisione. Essendo la prima, K può essere 0 o 1. Se sceglie K=0 allora inizia a rilevare la portante del canale(per verificare quando termina e cominciare a trasmettere), se sceglie K=1 allora aspetta il tempo necessario ad inviare K volte 512bit per poi ripetere il ciclo(rilevamento della portante e trasmissione).
Dopo una seconda collisione, K viene scelto nell'intervallo 0-3, dopo tre collisioni K è scelto con uguale probabilità nell'intervallo 0-7 . Dopo 10 collisioni tra 0-1023. Quindi la cardinalità dell'insieme da cui viene scelto K cresce esponenzialmente al crescere delle collisioni(In Ethernet n può arrivare a 10).
(?????)
Perchè si aspetta il tempo necessario ad inviare 512bit? Tale intervallo, detto tempo critico, è pari al tempo di propagazione massimo moltiplicato per due. Con Ethernet abbiamo una dimensione massima che è 2.5km. Alla velocità di 10Mbit, una macchina riesce a spedire proprio 512bit ovvero 64 byte, nel tempo di propagazione del segnale. Secondo questo schema, passati i 64byte ormai eventuali collisioni nella situazione peggiore, sono già tornati indietro. Pertanto se si continua a trasmettere, la frame sarà sicura dato che in precedenza non ci sono stati problemi.
Trasmettendo meno di 64byte invece non si può avere la certezza che la frame non sia stata rovinata. Per tale ragione Ethernet impone il campo dati abbia una dimensione minima di 46byte. Quindi il collision detect mi serve sia per sapere se devo fermarmi a causa della collisione ma anche che se è stata rovinata devo ritrasmetterla.(????)

Protocolli senza collisioni

Bitmap
Si ha un numero di slot pari al numero di stazioni che possono parlare contemporaneamente nel mezzo: se ho 50 macchine devo avere 50 slot separati.
Le macchine che vogliono trasmettere devono mettere un bit nel loro slot(Periodo di contesa). Conoscendo le macchine che vogliono parlare, sarà possibile per loro trasmettere in ordine rispettando il proprio turno. Il protocollo non è però esente da collisioni in quanto degli host potrebbero non rispettare il protocollo.

Problemi
Se ci sono tante macchine, il periodo di contesa è molto grande. Inoltre se abbiamo tante macchine che parlano poco, il canale rimane inutilizzato per gran parte del tempo(che viene sprecato nella contesa).
Ancora, dovrei sapere a priori quante macchine ci sono nel mio sistema e non posso aggiungere o togliere macchine a piacimento. Non essendoci sincronizzazione l'aggiunta di una macchina col suo slot personale, potrebbe non essere recepita da tutte le macchine e causare problemi nell'ordine di trasmissione.

CSMA/BA CanBus
Invece di distribuire gli slot con una bitmap, lo schema di comunicazione sul canale è tale che se voglio trasmettere un bit 0 sto zitto, altrimenti mando un segnale. Se due macchine dicono 1 contemporaneamente, il segnale non va in collisione ed il risultante è 1, mentre se una dice 0 e l'altra 1 allora vince il bit 1 e lo 0 scompare(senza collisione).
Ogni macchina mette il proprio bit più significativo sul canale e tutte ascoltano.



Al passaggio successivo, prima di trasmettere un altro bit, le macchine ascoltano: se sul canale sentono 1 ed invece loro avevano trasmesso 0 allora fermano la loro trasmissione, mentre quelle che avevano trasmesso 1 continuano a trasmettere.
Tale schema non rappresenta uno spreco di banda in quanto le macchine stanno già scrivendo sulla frame trasmettendo il proprio indirizzo.
Vantaggi: non ho slot e quindi non c'è nemmeno ridondanza in essi ed inoltre sto già cominciando a scrivere la frame.
Nota Bene: se ho un indirizzo alto(tanti 1) ho priorità sulle altre macchine.

Token Bus
L'idea è quella di realizzare un anello logico in cui le macchine scambiano tra loro un token(permesso) per parlare. Chi ha il token occupa il canale e quando termina la sua trasmissione lo passa alla macchina successiva.
Problema: il token è una frame e potrebbe perdersi o collidere con un altra frame.
Per tale ragione tutte le macchine sorvegliano il canale per individuare il passaggio del token, se dopo un certo lasso di tempo, tutte notano che il token non sta girando allora sono autorizzate a generarne uno nuovo.
Per regolare la presenza di più token nel canale è necessario attuare delle politiche per rimuovere quello superfluo.
Se l'uscita dall'anello non causa particolari problemi, non vale lo stesso per l'entrata. Dovrebbe essere previsto un momento di contesa dove tutte le macchine che vogliono entrare nel canale provano ad utilizzarlo, con la possibilità di collisione→ richiede del software molto complesso.

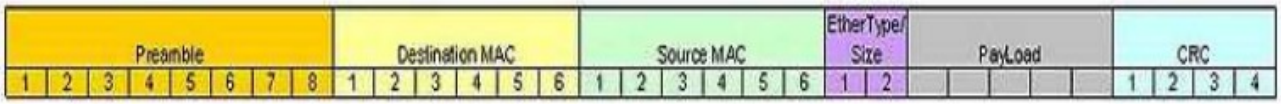
IEEE802
Tutte le reti LAN e metropolitane.

Cablaggi

Name	Cable	Max. seg.	Nodes/seg.	Advantages
10Base5	Thick coax	500 m	100	Original cable; now obsolete
10Base2	Thin coax	185 m	30	No hub needed
10Base-T	Twisted pair	100 m	1024	Cheapest system
10Base-F	Fiber optics	2000 m	1024	Best between buildings

Quello più comune è stato il 10Base[X], dove 10 sta per 10 Mbps, base si riferisce al fatto che il mezzo fisico trasporta solo traffico Ethernet ed infine 5,2,T o F indicano il tipo di cavo.
Il primo è il cavo coassiale grosso(thick), 10Base5, molto rigido in quanto costituito da un pezzo di rame unico non intrecciato, una guaina isolante, una calza metallica di protezione e una guaina esterna di protezione totale. La lunghezza massima di un segmento unico era di 500 metri e potevano essere collegati tra di loro con dei ripetitori/amplificatori di segnale fino ad un massimo di 5 segmenti consecutivi(2.5km). Per intercettare il cavo interno, si usava la presa a vampiro: veniva forata la parte isolante e andava a fare contatto con la parte interna. Per non fare cortocircuito si andava a tagliare la guaina in dei punti specifici segnati da dei puntini verdi(ogni 2.5metri) fino ad un massimo di 100 nodi.
10Base2 è invece molto più flessibile e per estendere il cavo semplicemente si tagliava e si inseriva il connettore cosa che rovinava il segnale. I segmenti avevano una lunghezza pari a 185-200metri ed era possibile unirne 5 per una lunghezza massima di 1km e 30 nodi per segmento.
10Base-T, costituito da due cavi intrecciati .I segmenti avevano grandezza pari a 100 metri cui potevano essere collegati fino a 1024 nodi. Lo schema di collegamento infatti prevedeva un hub centrale, un concentratore ed una connessione diretta tra scheda di rete e concentratore.
10Base-F è la fibra ottica. Può trasmettere fino a 2km ed avere al più 1024 nodi anche se in realtà la comunicazione è punto a punto come per quello intrecciato.

Ethernet



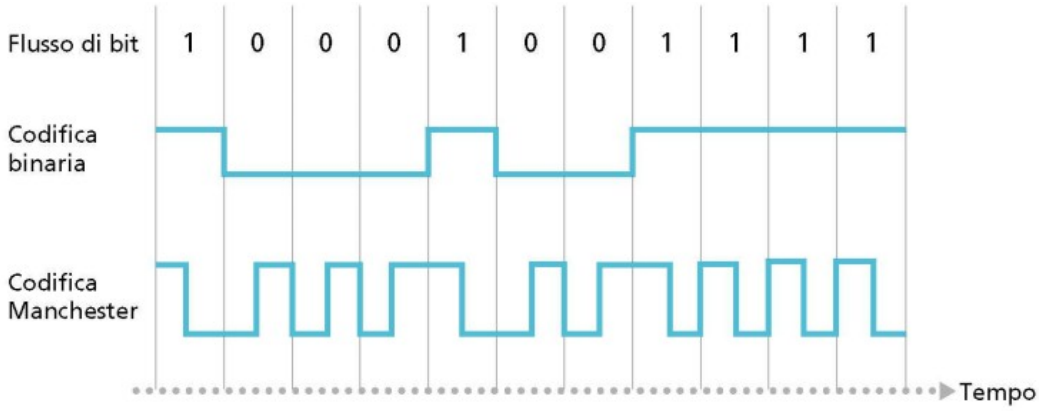
Un pacchetto Ethernet contiene 6 campi:

1. Campo di dati: contiene il datagramma IP. Ethernet ha un MTU pari a 1500byte: se il datagramma supera tale soglia allora deve essere frammentato. Se invece il datagramma è sotto la soglia minima del campo dati, corrispondente a 46byte, allora il campo viene riempito con dei dati fino a raggiungere quel valore(Padding).
2. Indirizzo di destinazione: campo che contiene l’indirizzo MAC dell’host ricevente.

3. Indirizzo sorgente: campo che contiene l'indirizzo MAC dell'host mittente.
4. Ether Type(2byte): consente ad Ethernet di supportare vari protocolli di rete. È il campo che permette di comunicare con il protocollo del livello di rete, come lo erano il campo protocollo nel datagramma IP ed i numeri di porta nel segmento di trasporto.
5. Controllo a ridondanza ciclica(CRC): campo che consente alla scheda di rete di rilevare degli errori nella frame.
6. Preambolo: ogni frame Ethernet inizia con un campo di 8 byte: sette hanno i bit 10101010 e l'ultimo ha bit 10101011. I primi sette servono per risvegliare le schede di rete e sincronizzare i loro clock, mentre l'ultimo avvisa la scheda di rete che la frame vera e propria sta arrivando.

Ethernet offre un servizio senza connessione(come UDP), privo di handshake e poco affidabile. In particolare il mittente che ha ricevuto la frame, effettua un controllo sulla sua correttezza tramite CRC, ma non invia alcun ACK né se la frame ha superato il controllo e né se invece è stato rilevato qualche errore.
Queste mancanze permettono però ad Ethernet di essere semplice ed economica.

Codifica Manchester
Tipo di codifica utilizzata nel passaggio dal livello fisico al livello di collegamento.



Ci sono due livelli, alto e basso: il bit 1 identifica il passaggio da alto a basso mentre il bit 0 il passaggio da basso ad alto.
Per scandire bene i passaggi tra un bit e l'altro dobbiamo sincronizzare mittente e destinatario: ciò avviene tramite il preambolo.
Esiste poi una codifica di manchester differenziale dove 1 continua a rappresentare una variazione, mentre lo zero rappresenta una ripetizione dello stato precedente.

Fast Ethernet

Name	Cable	Max. segment	Advantages
100Base-T4	Twisted pair	100 m	Uses category 3 UTP
100Base-TX	Twisted pair	100 m	Full duplex at 100 Mbps
100Base-FX	Fiber optics	2000 m	Full duplex at 100 Mbps; long runs

Aumento della velocità di Ethernet.
Per determinare la dimensione minima della frame devo guardare alla lunghezza massima del cavo.

Vengono usate tutte e 4 le coppie: due vengono poste una per l'andata ed una per il ritorno alla velocità di 33Mbps ciascuna mentre le altre due coppie le utilizzo alternativamente o in una direzione o nell'altra, in modo tale che all'occorrenza ho da un lato 33x3 e dall'altro 33. Quindi non ho 100Mbps Full-duplex ma solo quando mi serve, in una direzione o nell'altra.
Viene abbandonata la codifica Manchester: nel cavo TX si usa la 4B5B mentre per il T4 uso la 8B6T.

Gigabit Ethernet

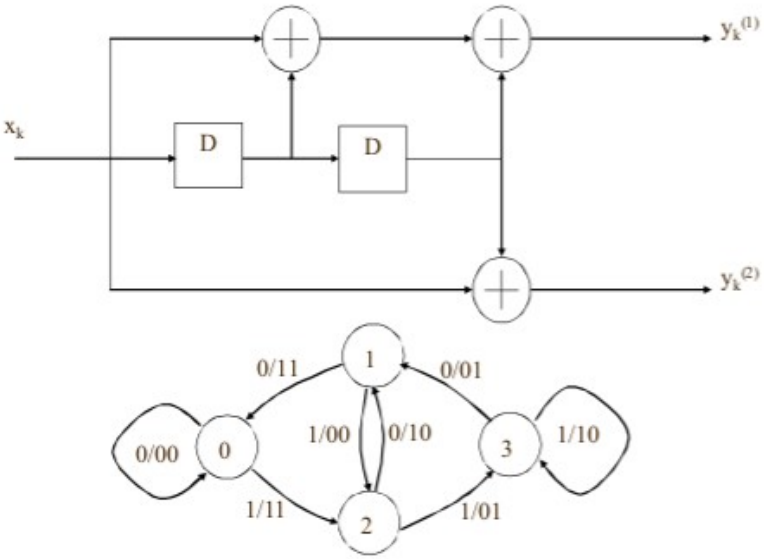
Name	Cable	Max. segment	Advantages
1000Base-SX	Fiber optics	550 m	Multimode fiber (50, 62.5 microns)
1000Base-LX	Fiber optics	5000 m	Single (10 μ) or multimode (50, 62.5 μ)
1000Base-CX	2 Pairs of STP	25 m	Shielded twisted pair
1000Base-T	4 Pairs of UTP	100 m	Standard category 5 UTP

Dato il successo di Fast Ethernet si è pensato di aumentare ancora di più la velocità. Il problema è però il cablaggio: col Fast si è già dovuto rinunciare al cavo coassiale a causa dei problemi di affidabilità ed alla dimensione minima delle frame(Fast Ethernet con cavo coassiale avrebbe portato la grandezza minima a 640byte).
L'innovazione è stata nel cercare di utilizzare i cavi già esistenti(CAT5e) utilizzando tutte e 4 le coppie ed evitando di ripassare i cavi dove già erano presenti.
Cinque passi verso 1000Base-T(CAT5):

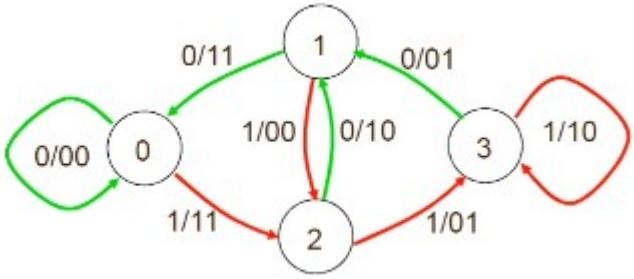
1. Rimozione codifica 4B5B: questo passaggio permette di aumentare il throughput da 100 a 125 Mbps ma ritorna il problema del framing.
2. Uso delle 4 coppie simultaneamente(125-500Mbps): a differenza di Fast che ne usa solo 2 una di andata ed una di ritorno qui vengono usate tutte e 4 simultaneamente.
3. Trasmissione full-duplex: conseguenza dell'uso di 4 coppie in simultanea.
4. Uso di 5 livelli per baud invece che 3(500-1Gbps full duplex): un livello viene usato per il framing, gli altri 4 permettono di trasportare 2 bit a livello → aumento numero di errori.
5. Usare Forward Correction: correzione errori a destinazione per sistemare il problema.

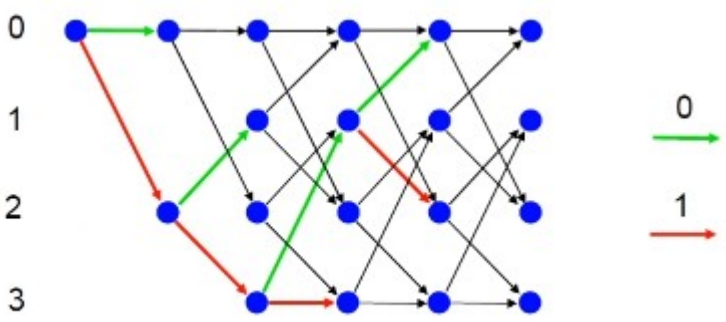
***
Un baud è la possibile variazione di segnale dallo stato precedente.
Il baud al secondo rappresenta quante variazioni ci possono essere nell'arco di tempo.
Ad ogni baud si può associare un solo bit oppure 2 bit nella codifica a 5 livelli.
***

Codifica Trellis Viterbi



Per decodificare il segnale si usa la codifica di Viterbi.
L'idea è quella di trasmettere 2 bit per ognuno che entra, introducendo così il 100% della ridondanza al fine di recuperare gli errori. Con un automa a stati finiti ci sono 4 stati {0,1,2,3} ognuno dei quali ha un comportamento e due frecce in uscita.
L'automa è presente sia alla sorgente che alla destinazione, le quali concordano lo stato iniziale. In base allo stato in cui mi trovo ed a quello che voglio trasmettere, mando la corrispettiva coppia di bit. Ad esempio lo stato 0 può rimanere nello stesso stato se in input riceve il bit 0 ed in uscita trasmetterà la coppia 00, oppure può passare allo stato 2 se riceve il bit 1 ed in output emetterà i bit 11.



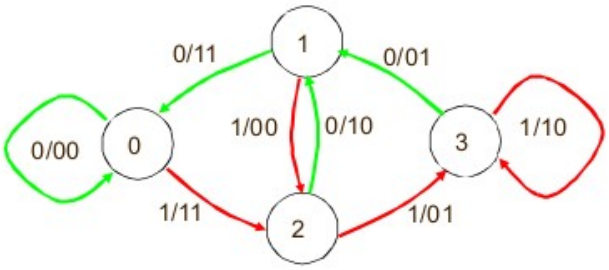


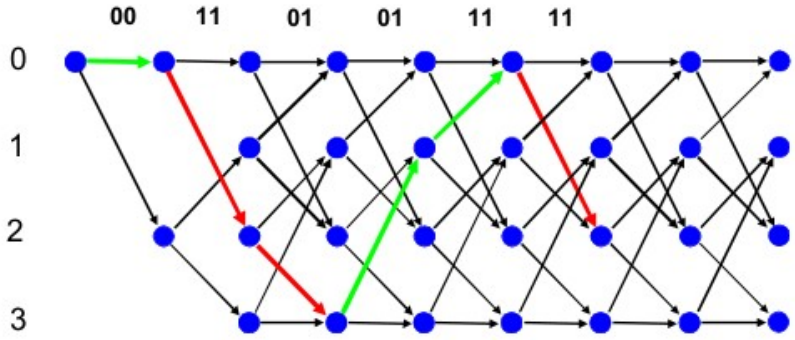
Evidenziando in verde i bit a 0 ed in rosso i bit ad 1 è possibile riprodurre l'automa con questo traliccio il quale ci dice appunto che: lo stato zero può rimanere a zero oppure passare allo stato 2 mentre ad esempio gli stati 1 e 2 non possono mai restare in loro stessi e variano sempre il loro stato.
Data una sequenza in input, genero un percorso all'interno di questo traliccio seguendo FSM.
Notiamo che:

1. Le uscite 0 e 3 sono antisimmetriche.
2. Le uscite 1 e 2 verso gli stati 0 e 3 sono antisimmetriche.
3. 0,3,1,2 sono simmetrici tra loro.
4. Le frecce entranti 0 e 1 hanno come input 0 mentre le frecce entranti in 2 e 3 hanno input 1.

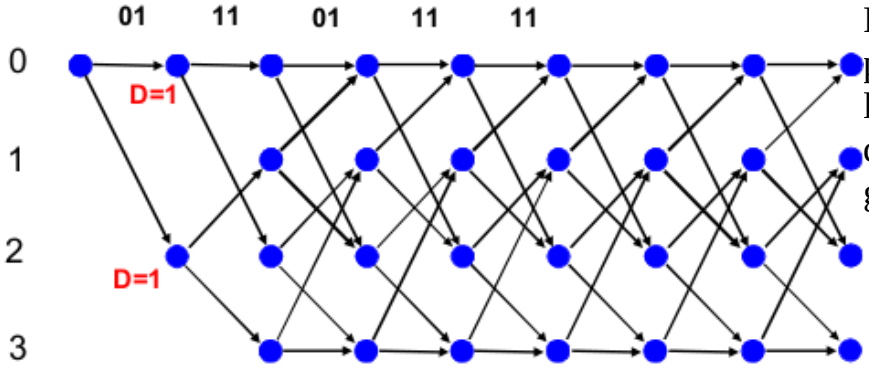
Considerando una sequenza abbastanza lunga a distanza di 2 salti posso arrivare a qualunque altro stato. Una volta passata la fase iniziale gli stati sono equiprobabili e questo ci permette di fare la correzione degli errori.
Esempio

0 1 1 0 0 1

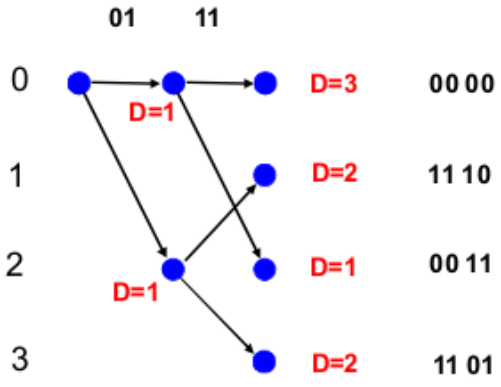




Supponiamo di ricevere in input la sequenza 011001. Lo stato iniziale viene settato a 0.La sequenza generata sarà 00 11 01 01 11 11. A distanza di 1 salto posso avere tutte le possibili coppie di 2 bit emesse. Questa regola è valida a partire da qualunque stano. In particolare avremo che 0 e 1 emettono coppie di bit uguali mentre 2 e 3 coppie di bit diversi. Se inserissimo degli errori nella sequenza ottenuta in modo da avere una sequenza del tipo 01 11 01 11 11 11, usando sempre lo stesso traliccio e sempre lo stesso stato iniziale cioè zero riusciremmo facilmente ad individuare gli errori. Ad esempio già dal fatto che lo stato iniziale è zero possiamo dedurre che la prima sequenza è errata in quanto coppia di bit diversi. Non sappiamo però quale bit è quello sbagliato.



Per tale motivo prendiamo i possibili cammini e manteniamo le distanze di Hamming rispetto a quello che poteva essere generato(ovvero 00 oppure 11).

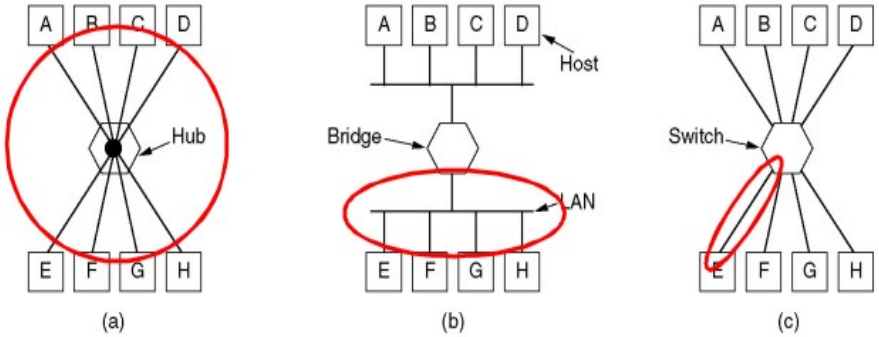


Facendo la stessa cosa per la coppia successiva 11:
Se tutte le destinazioni sono equiprobabili, la probabilità che ci sia un solo errore è maggiore a quella di averne due o tre, quindi la strada da seguire è probabilmente quella a distanza D=1. La considerazione continua andando a vedere cosa succede nella coppia successiva, cioè 01 e così via tutte le altre coppie.
In generale il percorso corretto è quello che mantiene al minimo possibile la distanza.
Questo sistema ci permette di effettuare la correzione a destinazione inserendo un elevato numero di bit errati(1 ogni 6) a patto che essi non

siano troppo vicini tra loro. In caso di fallimento di questo sistema c'è sempre il CRC che potrebbe individuare gli errori che sono sfuggiti.

Collegamenti tra LAN differenti

La definizione originale di LAN riguardava tutto il sistema di bus condiviso dove potevano verificarsi le collisioni(ovvero il dominio di collisione).



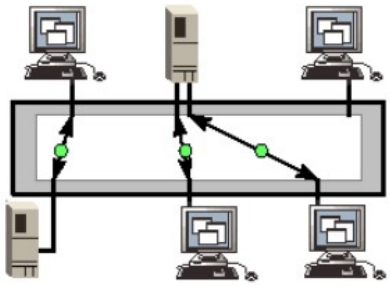
In Ethernet 10Base-2 o -5 c'era un unico cavo di connessioni, tante derivazioni più o meno lunghe ma di fatto c'era un unico mezzo fisico ben identificabile che costituiva appunto la LAN.
L'evoluzione è stata quella di passare da 10Base-2 a 10Base-T cioè il cavo Twisted Pair. A questo punto si parla di Hub ovvero un concentratore che non faceva altro che riprodurre elettricamente i segnali provenienti da una coppia senza fare alcuna interpretazione del segnale: in tal modo il dominio di collisione comprende tutti i cavi che arrivano all'hub.
Passando allo switch le cose si complicano: esso è un dispositivo intelligente che ha una scheda di rete per ogni interfaccia di ingresso. Ciò significa che legge le frame, le trasforma in bit e poi le ripropone sull'uscita corretta. Il dominio di collisione è quindi ristretto fino alla singola tratta di collisione tra i due dispositivi(switch e mittente) in quanto essendo una connessione punto-punto, eventuali collisioni non possono propagarsi sul resto della rete. In termini di LAN, lo switch presenta 8 LAN separate.

C'è inoltre un altro dispositivo chiamato Bridge il quale è un dispositivo intelligente(provvisto di DLL a differenza dell'Hub) con due interfacce distinte. In tal senso le frame della LAN E-H vengono interpretate e se necessario fatte girare verso l'altra uscita ovvero la LAN A-D. Dato che sono due interfacce separate queste LAN potrebbero anche essere considerate differenti. Se il bridge è connesso a due LAN Ethernet allora ha un compito facilissimo: prendere la frame da un lato e se necessario riproporla all'altro.

Lo switch può quindi essere considerato come un bridge specifico per Ethernet: in tal senso possiamo estendere il concetto di LAN a tutta la struttura dato che essa è uniforme e soprattutto le operazioni fatte dallo switch non tengono conto di cosa c'è sopra il DLL(guarda solo il MAC Addr). All'interno dello switch sono presenti un processore, una memoria ed un firmware operativo che permettono di far lavorare il dispositivo interpretando il contenuto della frame Ethernet.

Tali considerazioni ci portano a pensare alla LAN non più come dominio di collisione ma come rete uniforme in cui si parla lo stesso linguaggio: Ethernet.

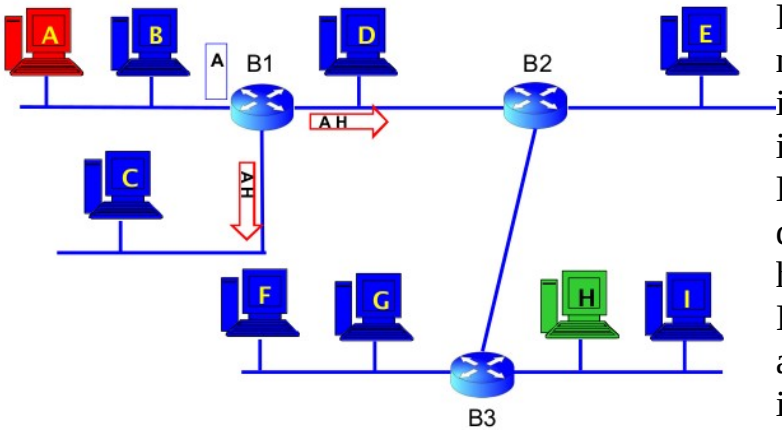
Switch



Indirizzo	Interfaccia	Tempo
62-FE-F7-11-89-A3	1	9:32
7C-BA-B2-B4-91-10	3	9:36
.....	.....	.....

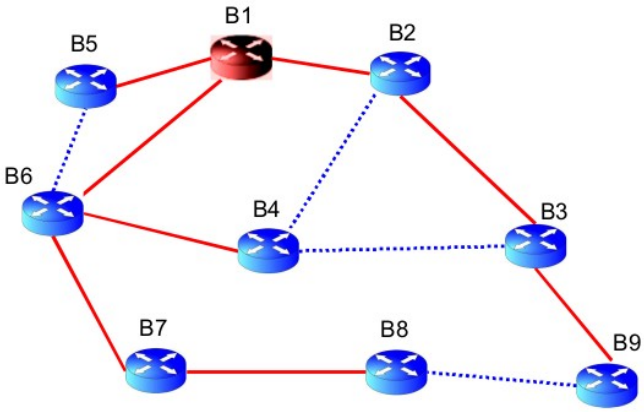
Il compito dello switch è quello di analizzare le intestazioni delle frame e trovare la strada migliore per raggiungere la destinazione preparando al suo interno un percorso per collegare le due macchine senza che le altre vengano influenzate. Questo significa che se per ogni comunicazione abbiamo un massimo di 10Mbps allora la comunicazione tra le macchine centrali andrà a 5mbps mentre quella a sinistra viaggerà a 10mbps. In totale si avrà quindi una banda aggregata utilizzata di 20Mbps.
Il processo di inoltro viene favorito grazie alla presenza di una tabella di inoltro che ha tre voci per ogni record: indirizzo destinazione, interfaccia e un parametro che indica quando è stato inserito il record.

Funzionamento(B1, B2, B3 sono switch)



La macchina A vuole parlare con la macchina H e prepara una frame dove inserisce gli indirizzi corretti. Essa viene inserita sul bus dove tutti ascoltano mentre B la ignora(pezzente). Gli switch sono dispositivi plug and play ovvero che non hanno bisogno di alcuna configurazione. Inizialmente le loro tabelle sono vuote, ma ad ogni frame che riceve acquisisce informazioni sugli host: infatti quando B1 vede la frame, non sapendo dove si trova la macchina H,la rimanda su tutte le uscite ed aggiorna la propria tabella di inoltro con la posizione di A. Anche C e D ignorano la frame, che arriva a B2. Questo fa la stessa cosa di B1 e dirige il pacchetto verso lo switch B3 tramite cui la frame arriva ad H. H risponde con una frame indirizzata ad A, ma stavolta gli switch sanno dove A si trova e quindi il pacchetto di risposta va direttamente sulla linea corretta.
Una volta arrivato tutti gli switch sanno dove si trovano A ed H. Da ora in poi nelle successive comunicazioni non verranno più coinvolti tutti i collegamenti ma solo quelli necessari a trasportare la frame da A ad H.
Quando dopo un dato periodo di tempo detto aging time lo switch non riceve frame da un determinato indirizzo lo cancella dalla tabella.

Problemi



Uno dei problemi degli switch riguarda la loro topologia. Al fine di prevenire cicli non è possibile realizzare dei grafi e viene utilizzato il metodo degli spanning tree: da un grafo con cicli si genera un albero di copertura con alcuni percorsi tagliati.
A tal fine si cerca di eleggere un nodo radice, magari quello con l'identificativo più basso. Supponiamo di avere una serie di switch B1...B10 allora per esempio B1 sfida B5 a chi ha l'identificativo più basso e vince. Tale sfida si ripercuote per tutto l'albero fino a decretare B1 come radice. In particolare gli switch collegati direttamente alla radice saranno nodi di primo livello, quelli a distanza 2 dalla radice di 2 livello e così per il resto dei nodi. I collegamenti tagliati sono quelli tra i nodi dello stesso livello che non possono comunicare direttamente ma solo grazie all'ausilio di altri nodi.

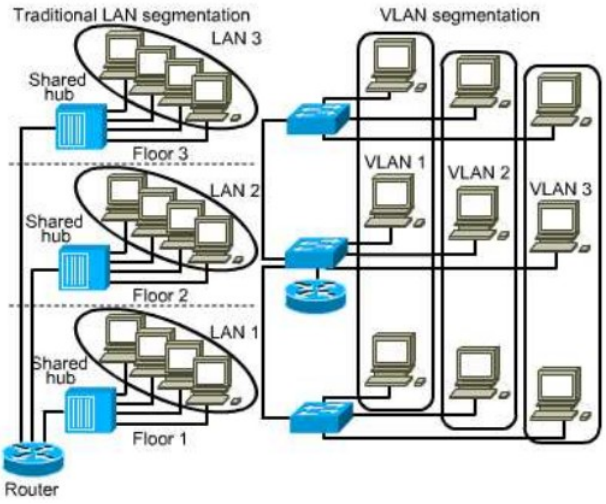
Differenze tra Repeater, Hub , Bridge, Switch
Repeater: è un amplificatore che serviva per connettere più segmenti tra loro. Dispone di due porte, una di entrata ed una di uscita, che gli permettono di ripetere il segnale tra un segmento ed un altro, con la possibilità di introdurre errori e collisioni.
Hub: dispositivo a livello fisico con una struttura a stella cui erano collegati più nodi. Come il repeater aveva il compito di ascoltare da una porta e ripeterlo sulle altre collegate con la possibilità di introdurre errori e collisioni nel caso ad esempio dell'arrivo di due frame contemporaneamente.
Bridge: dispositivo che permette di trasformare le frame da un formato LAN ad un altro. Considerando l'espansione di Ethernet è ormai poco utilizzato.
Switch:multiporta con 5-8 porte o 12-48. Dispositivo intelligente con una CPU e delle memorie in ingresso ed in uscita dove accodare pacchetti durante la ricezione/trasmissione: a differenza dell'hub in presenza di due frame invece di trasmetterle entrambe, ne tiene una in coda finché non ha finito di trasmettere l'altra.
Router: dispositivi a livello di rete.

Gerarchia
Possiamo distinguere tra dispositivi di livello 1(fisico) come repeater e hub, di livello 2(DLL) come Bridge e Switch(servono per connettere LAN differenti), mentre i router sono dispositivi di livello 3(livello di rete).
La grossa differenza tra switch e router riguarda il fatto che il primo non ha bisogno di configurazione mentre il secondo si.

Indirizzamento flat vs gerarchico(Ampliamento discorso MAC vs IP fatto anni fa)
Un'altra differenza da sottolineare è lo schema di indirizzamento di base che si usa. Abbiamo instradamento a livello DLL e instradamento a livello di Rete.
A livello DLL gli indirizzi hanno un formato flat, cioè senza nessuna logica di distribuzione di indirizzi nella rete: gli indirizzi MAC infatti dipendono dalla scheda

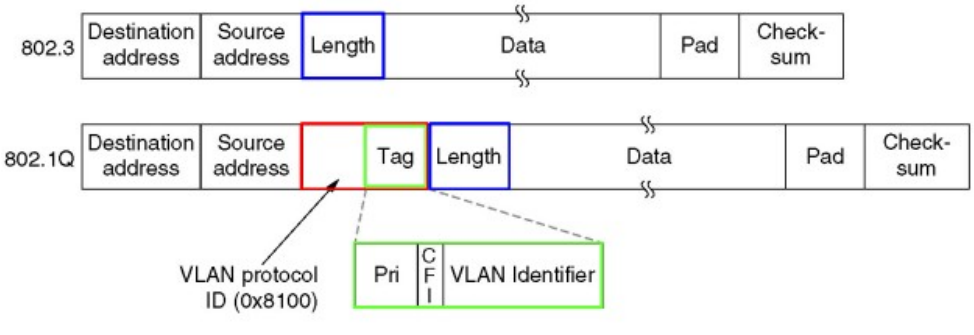
di rete e non contengono info di instradamento. Per tale motivo essi possono essere collegati in qualunque punto della rete. Gli indirizzi a livello di rete invece si dicono gerarchici proprio perché ci danno informazioni sulla collocazione dell'host in rete.
In particolare gli indirizzi IP sono composti da tre campi principalmente: rete, sottorete e host. La rete ci permette di effettuare la prima fase di routing che consiste nell'arrivare alla struttura di destinazione, dalla quale si userà poi la parte di sottorete ed infine la parte relativa al singolo host.
Nonostante gli indirizzi flat siano più comodi in quanto fissi(a meno che non sia l'utente stesso a cambiarlo) essi hanno un grosso limite: una LAN infatti non può essere composta da migliori di host perché Bridge e Switch nelle fasi iniziali ottengono informazioni inviando frame in broadcast→ con troppi host si perderebbero i svantaggi della struttura a switch.
Normalmente essi hanno una tabella di inoltro di 1000-4000 indirizzi non di più.

VLAN(LAN virtuali)



Supponiamo di avere una struttura a tre piani, dove ad ognuno di essi corrisponde una LAN. Generalmente in ogni LAN associata ad un piano ci sono macchine che svolgono un solo tipo di lavoro.
Le VLAN permettono invece di separare il posizionamento fisico(nel piano) della macchina dal posizionamento della LAN. Tramite le VLAN infatti possiamo avere macchine che appartengono alla stessa VLAN ma che stanno su piani,quindi LAN fisiche, diverse. Il traffico è quindi principalmente verticale e coinvolge piani separati.
Per farlo è necessario avere degli switch(3 livello) appositi.
In particolare le porte di uno switch per VLAN sono suddivise in gruppi, ognuno dei quali costituisce una VLAN: il traffico broadcast che proviene da una porta appartenente alla VLAN1 può raggiungere solo altre porte della stessa VLAN1.
Se non c'è un collegamento diretto tramite un cavo od un meccanismo software, le VLAN non possono comunicare tra di loro su switch diversi e nemmeno col mondo esterno.
Questo tipo di VLAN è detta **untagged** cioè sprovvista di un tag identificativo.

Col protocollo 802.1Q è possibile introdurre una variazione al formato delle frame Ethernet per creare delle VLAN **tagged**.



La nuova frame è uguale a quella standard con l'aggiunta

di una etichetta VLAN(il tag appunto) di 4 byte che trasporta l'identità della VLAN cui la frame appartiene.
La prima coppia di byte contiene un valore esadecimale che identifica il protocollo della VLAN(tag protocol identifier) mentre la seconda coppia di byte contiene tre campi: un campo Priority che identifica il livello di priorità, un campo CFI che identifica se si tratta di un indirizzo MAC canonico(bit a 0) o non canonico(bit a 1) ed infine un campo VLANID che costituisce un identificatore univoco per la VLAN.
Tramite questo protocollo è possibile andare ad unire VLAN differenti in quanto gli switch possono capire facilmente a quale VLAN appartiene un determinato frame.

Vantaggi VLAN
Le VLAN permettono maggiori prestazioni, in quanto il traffico broadcast viene confinato agli utenti della stessa VLAN. Un'altro importante aspetto riguarda la sicurezza: gli utenti di VLAN differenti non vedono i reciproci frame dati.
Infine l'uso delle VLAN permette di spostarsi su 'piani' differenti senza doversi connettere ad una LAN diversa.

Livello Fisico
Il livello fisico si occupa di trasportare i segnali: per farlo è necessario avere:

1. Un canale trasmissivo in grado di garantire il trasporto di segnali.
2. Un generatore di segnali.
3. Un rilevatore di segnali.

Il canale è un dispositivo che riesce a trasmettere segnali andando a deformare una sua caratteristica fisica(come una molla). Se a destinazione abbiamo un dispositivo che riesce a percepire tale deformazione allora è possibile generare una trasmissione di segnali.

I canali trasmissivi si suddividono in:
● Canali Perfetti: non causano distorsioni o ritardi nella propagazione dei segnali.
● Canali Ideali: causano solo un ritardo costante nella propagazione
● Canali reali: causano attenuazioni e ritardi, in funzione della frequenza dei segnali. ("le schifezze più immonde").

Un dispositivo semplice può essere un mezzo elettrico tramite cui è possibile passare da bit a segnale e viceversa. Ancora, è possibile usare le onde elettromagnetiche per i mezzi wireless o i mezzi ottici per i canali con larghezza di banda molto elevata. Questi tre approcci usano tutti onde elettromagnetiche, rispettivamente a bassa, alta ed altissima frequenza.

Analisi di Fourier

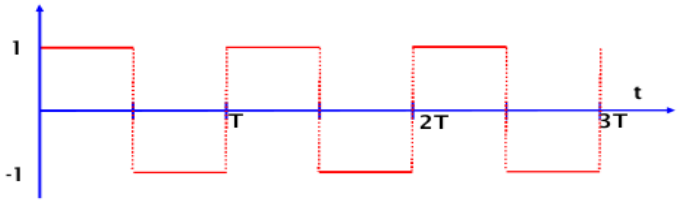
$$g(t) = \frac{1}{2}c + \sum_{n=1}^{\infty} a_n \sin(2\pi\ nft) + \sum_{n=1}^{\infty} b_n \cos(2\pi\ nft)$$

Un qualunque segnale di periodo finito può essere rappresentato per mezzo di una

sommatoria infinita di onde sinusoidali(dette armoniche date da somme di seni e coseni)di frequenza ed ampiezza opportune. Avendo un segnale periodico che si ripete all'infinito è possibile rappresentarlo in maniera matematica dove la frequenza è pari all'inverso del periodo del segnale stesso. In particolare il segnale risultante viene ottenuto andando a pesare la sinusoide con un peso a e la cosinusoide con un peso b tali che i quadrati di sinusoide e cosinusoide diano come risultato 1. I coefficienti vengono ricavati risolvendo tali integrali

$$a_n = \frac{2}{T}\int_0^T g(t)\sin(2\pi\, n\, f\, t)\,dt$$

$$b_n = \frac{2}{T}\int_0^T g(t)\cos(2\pi\, n\, f\, t)\,dt$$

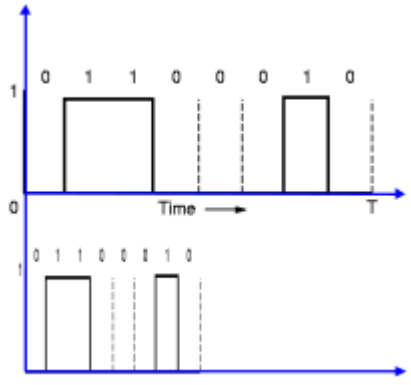


Esempio: Onde quadre
In questo caso applicando le formule dei coefficienti si scoprie che b e c sono pari a 0 mentre a diventa 0 per n pari, mentre diventa a=4/nπ per ogni n dispari. Quindi l'onda quadra può essere anche scritta come:

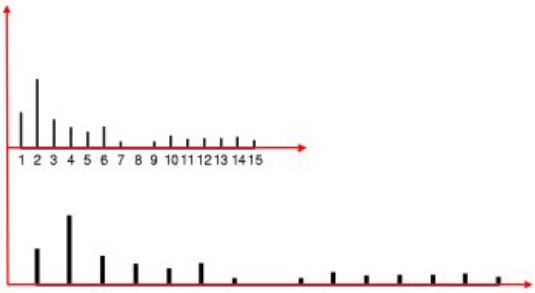
$$g(t) = \begin{cases} 1 & se \quad 0 + nT < t < \dfrac{1+2n}{2}T \\ -1 & se \quad \dfrac{1+2n}{2}T < t < (1+n)T \end{cases}$$

$$g(t) = \frac{4}{\pi}\sin(2\pi t) + \frac{4}{3\pi}\sin(6\pi t) + \frac{4}{5\pi}\sin(10\pi t)...$$

I contributi di ogni armonica(onda fondamentale, terza armonica , quinta…) diventano sempre più piccoli perché il denominatore cresce.
Man mano che vado a sommare le armoniche, ottengo un'approssimazione più o meno precisa dell'onda quadra.
L'analisi di Fourier mi permette di vedere l'onda come somma di onde sinusoidali a frequenze differenti. Invece di analizzare l'onda nel dominio del tempo(ovvero ad ogni istante) vado a scomporla in frequenze differenti, analizzandola nel dominio delle frequenze.
Questo non è altro che un modo diverso di vedere la stessa informazione(l'onda) che ci permette però di trarre qualche conclusione in più.
Supponiamo di voler trasmettere un segnale che sia assimilabile ad un'onda quadra e che rispetti le ipotesi di Fourier(onda periodica ed infinita).
Applicando le formule per ottenere i coefficienti, si può ottenere lo spettro del segnale, ovvero l'influenza che le varie armoniche hanno in esso.
Ipotizzando di trasmettere il segnale a 8Mbps, avremo che 1 bit ha una durata temporale di 1/(8*10^6)secondi. L'armonica fondamentale(la prima) è della frequenza di 1 MHz mentre la 256esima armonica è 256 volte la frequenza base ovvero 256MHz.

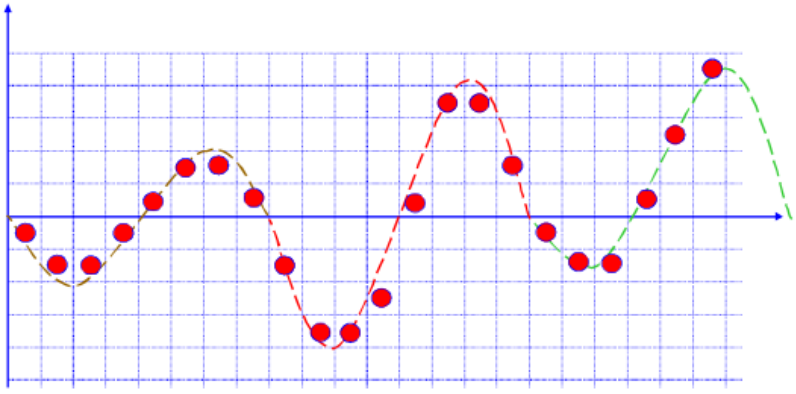


Larghezza di banda di un canale
Sulla base dell'esempio precedente, nel dominio del tempo il nostro segnale si presenta in questo modo:
Mentre nel dominio delle frequenze:



Se raddoppiassimo il bitrate passando da 8 a 16Mbps, il tempo della trasmissione sarebbe la metà e come conseguenza le frequenze raddoppiano. Tali sequenze devono però passare attraverso un canale reale(magari a bassa frequenza) : alcune di queste fuoriescono dal canale il quale le azzera.
Andando dunque ad aumentare il bitrate, il canale applica delle modifiche al segnale, generandone uno difficilmente interpretabile.
È importante sottolineare che le alterazioni del segnale non dipendono solo dal canale ma anche dal trasmettitore/ricevitore.

Quantizzazione(Quanto mi manchi Stanco)



La quantizzazione consiste nel limitare il numero di valori che la grandezza(il segnale) può assumere → il numero di livelli.
Quando devo trasmettere un segnale(informazione) quello che fa il trasmettitore è decidere a quale livello esso corrisponde.
Il trasmettitore(ed anche il ricevitore) non possono generare livelli di segnale infiniti ma procedono per quanti discreti. Questo tipo di quantizzazione è detto verticale: ne esiste poi un altro tipo detto orizzontale che corrisponde al tempo impiegato per passare da una frequenza alta ad una bassa(ovvero il tempo impiegato per cambiare stato).
Nella figura sopra, la differenza tra un elemento e quello successivo(pallino) descrive la larghezza di banda massima del segnale.
Teorema di Nyquist
Nyquist ha elaborato un’espressione che lega la larghezza di banda di un segnale con la quantità di informazioni trasportabili:
Max Bitrate= 2*H*log2(V) bit/s dove H è la larghezza di banda del segnale(data dalla quantizzazione orizzontale) mentre V è il numero di livelli nel segnale(dato dalla quantizzazione verticale).
Per aumentare il Bitrate dunque, se siamo limitati dalla larghrzza di banda, basta aumentare il numero di livelli(cioè fare la griglia più fitta) → è quello che è stato fatto con Ethernet passando da 3 livelli a 5 per portare 2 bit.
Non posso però aumentare i livelli all’infinito in quanto esiste il rumore.

Importante!!!!!!
Nyquist non pone limiti al bitrate trasportabile da un segnale.

Rumore
Il rumore è un segnale che si sovrappone alla comunicazione disturbandola.
Esistono vari tipi di rumore.

Rumore Termico
Scaturisce dall'eccitazione degli atomi causata dal calore .
A qualunque temperatura superiore a quella corretta, gli atomi oscillano all'interno della struttura in cui si trovano e generano disturbi elettromagnetici che seppur minimali possono causare interferenze in un segnale.
Per superare l'interferenza dovuta al rumore termico dobbiamo superare le interferenze prodotte dalle oscillazioni totali. La formula:
P=4KTBw ci dice che la potenza del segnale dipende dalla temperatura T, da una costante K ma soprattutto dalla frequenza che vogliamo andare a considerare.
Generalmente il rumore è a tutte le frequenze, ma ce ne sono alcune in cui non ci disturba più di tanto.
Ancora il rumore dipende anche dalla lunghezza del cavo: più lungo è il cavo maggiore sarà la resistenza che mette nella comunicazione(e quindi maggiore sarà il rumore). Sarebbe possibile ridurre la resistenza usando dei cavi più spessi, ma sono molto costosi.

Interferenze elettromagnetiche
Cavo Coassiale
Un cavo libero senza protezioni assorbe le radiazioni e le trasforma in energia elettrica andando così ad alterare il segnale. Per evitare questo assorbimento si potrebbe utilizzare un cavo coassiale(intrecciato a formare una treccia)che protegge il conduttore interno: in tal modo si impedisce all'energia di passare ed arrivare al conduttore. Partendo dall'interno un cavo coassiale è formato dal conduttore interno, dal dielettrico, da un nastro, una treccia di rame ed una guaina protettiva.
Questo sistema permette di creare un ottimo conduttore di segnali perché subisce poche interferenze elettromagnetiche dall'esterno. Tale sistema è però stato eliminato in quanto questi conduttori interni creavano delle interruzioni nel cavo che li rendevano inaffidabili.

Doppino Intrecciato
È costituito da una coppia di cavi elettrici intrecciati fra loro, lasciando un po' di spazio, a formare una spira. La corrente percorre un cavo da sinistra verso destra e l'altro da destra verso sinitra: in pratica si annullano tra loro(anche se il valore non è mai nulla data la possibile imperfezione nell'intrecciamento). Essendo intrecciata anche la spira è una pessima antenna, ma partendo da questo modello si può andare a

creare cavi migliori come CAT5, CAT5E, CAT6 e così via, ottenendo meno interferenza e meno rumore.
In CAT 6A ad esempio, abbiamo quattro coppie di cavi. Tra una coppia ed un altra si manifesta un fenomeno di passaggio dell'energia il quale può essere diminuito andando ad aumentare la separazione fra le coppie. Andando poi a ricoprire ogni coppia con un rivestimento è possibile ridurre ancor di più le interferenze.
Quindi, con piccoli accorgimenti possiamo realizzare cavi sempre più sofisticati che permettono di avere interferenze elettromagnetiche sempre più limitate e quindi andare a velocità maggiore.

Teorema di Shannon
Secondo Shannon, due sono i fenomeni che limitano la trasmissione:

1. Il numero di livelli discriminabili(al contrario di quanto afferma Nyquist)
2. La presenza di rumore(termico o interferenze).

Il Teorema di Shannon ci dice che: Massimo Bitrate= H*log2(1+S/N) bit/s
dove H è la larghezza di banda del segnale , S è la potenza del segnale sul canale ed N è la potenza del rumore sul canale. Da questa espressione si deduce chiaramente che per ogni canale esiste un ben preciso limite fisico. Per aumentare il bitrate è necessario aumentare la potenza del segnale o diminuire il rumore.
Se tutte le variabili sono però fisse non è possibile andare oltre il bitrate calcolato con la formula.
Per aumentare il birate, nel caso in cui S/N sia costante allora dobbiamo necessariamente aumentare la banda del segnale.
Se invece la banda è costante dobbiamo migliorare il rapporto segnale/rumore.
Più lungo è il canale trasmissivo maggiore è la potenza del rumore.
Shannon dice esattamente l'opposto di Nyquist: avere meno livelli rendi i dati meno suscettibili al rumore in quanto se ci sono meno livelli il rumore non riesce a rovinare i dati in quanto non può modificare i valori quantizzati.
Quindi per Shannon il numero di livelli non conta, ma va ad evidenziare invece il rapporto segnale-rumore e la larghezza di banda.

Esempio Canale Telefonico
Un normale canale telefonico consente di utilizzare frequenze nel range 30-3000Hz mentre tutto il resto viene tagliato. Tipicamente il rapporto segnale, rumore S/N è di 30 dB. Applicando la formula di Shannn il bitrate massimo è di 30000bps. Questo valore è un valore medio, perché dipende da molti fattori come ad esempio la distanza dalla centrale telefonica. I modem migliori potevano arrivare anche a 52kbps.

Modulazione
La modulazione indica l'insieme delle tecniche di trasmissioni finalizzate ad imprimere un segnale elettrico o elettromagnetico detto modulante(contenente informazioni) su un altro segnale elettrico o elettromagnetico ad alta frequenza, detto portante.
La Modulazione serve a far sì che il segnale che vogliamo trasmettere possa rientrare

all’interno del range in cui il canale può trasmettere.
Indichiamo con una funzione generica s(t) il segnale da dover trasmettere. In particolare il segnale varia nel tempo e avrà il suo spettro di frequenze.
Consideriamo poi un’onda sinusoidale rappresentata dalla funzione p(t) detta onda portante. Sia omega t la frequenza del nostro segnale e gamma la fase iniziale; quest’ultima viene utilizzata per far si che questa onda sinusoidale sia:

1. Per gamma=0 un seno.
2. Per gamma=pigreco/2 un coseno.
3. Infine per gamma compreso nei valori di p(t), un’onda traslata orizzontalmente di un certo periodo(come Fourier).

Quindi p(t) si chiama portante proprio perché deve trasmettere il segnale che ci interessa. Combinando s(t) con p(t) si ottiene una funzione g(t) che sarà quella inserita sul canale. Ci sono tre possibilità di combinazione in quanto abbiamo nella portante tre elementi che possono cambiare: A, omega e gamma e quindi avremo tre tipi di modulazioni a seconda di ove inseriamo il segnale nella formula(moltiplicato per una costante k). In altre parole il segnale fa variare l’ampiezza, la frequenza o la fase dell’onda.

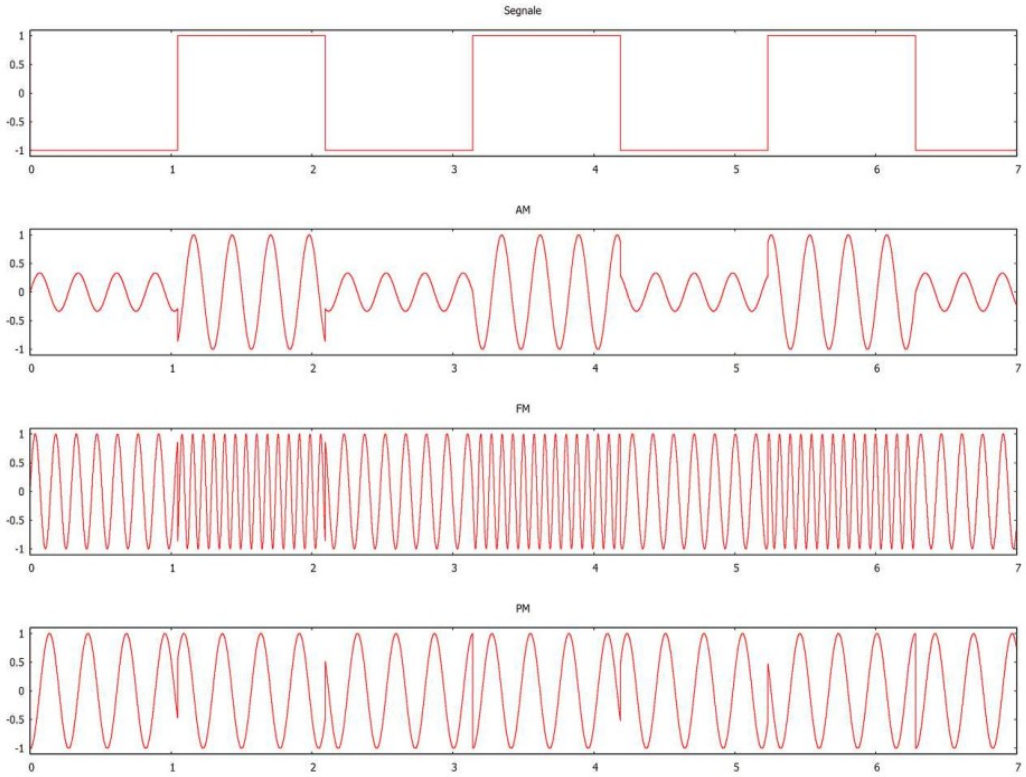
Modulazione di Ampiezza
Ha l’effetto di far avvicinare elementi lotani tra loro.
Ponendo l’ampiezza A fissa ad 1 otteniamo che s(t)=sin(omega1 t);p(t)=sin(omega2 t) g(t)=s(t)*p(t)→ otteniamo la formula di Werner:
$g(t) = 1/2 * [\cos((\omega1-\omega2)*t) – \cos((\omega1+\omega2)*t)]$
Per ottenere nuovamente l’originale, basta applicare l’operazione inversa.

Modulazione di Frequenza
L’effetto è quello di una compressione nelle zone positive del segnale originale e di una dilatazione nelle zone negative.

Modulazione di Fase
Determina un risultato praticamente identico alla modulazione di frequenza: le loro onde infatti sono legate matematicamente da un’operazione di derivazione/integrazione.

Spesso si usano due sistemi di modulazione in contemporanea: ad esempio la televisione analogica usava la modulazione di ampiezza e quella di frequenza.

Esempio di modulazione con onda quadra



Da notare come nella modulazione di fase, si abbia tuttosommato un segnale uniforme, ma in corrispondenza di un onda quadra si nota chiaramente un cambiamento di fase. Questo cambiamento permette facilmente di notare cambiamenti nel segnale e quidni porre adeguatamente i bit a 0/1.

Analogico o Digitale

Una grandezza è detta analogica se può variare in maniera continua tra un minimo ed un massimo mentre è detta digitale se può assumero solo valori precisi in un numero finito di livelli.

Un segnale analogico utilizza una larghezza di banda limitata ma subisce distorsioni ad ogni processo rigenerativo. Per esempio, una musicassetta copiata 10 volte, la decima avrà il rumore di dieci duplicazioni consecutive.

Un segnale digitale ha una larghezza di banda più elevata, ma può essere rigenerato con estrema precisione (quindi senza introdurre errori). Per questo motivo le comunicazioni digitali stanno soppiantando le analogiche.

Analogico e digitale sono modi di vedere una trasmissione.

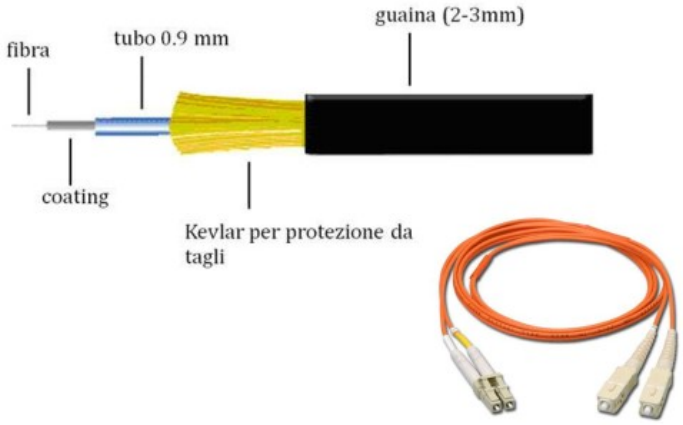
Dal punto di vista della trasmissione digitale dipende tutto da quanta precisione si riesce ad avere nella quantizzazione del segnale.

Essenzialmente, nel sistema analogico abbiamo un numero enorme di stati discreti, talmente enorme che il ricevitore/trasmettitore difficilmente riescono a discriminare tra un livello e l’altro.

Invece il sistema lo chiameremo digitale se il numero di stati che noi consideriamo

validi per la comunicazione è un numero molto limitato (4, 16, 32…), in maniera tale che siano sufficientemente distanti tra loro per far sì che il rumore che si aggiunge alla trasmissione non vada a fare interpretare un livello rispetto a un altro.
Dunque, il segnale analogico, non esiste, come non esiste il digitale puro.
Sono tutte e due approssimazioni che facciamo per comunicare.

Fibra Ottica
Un raggio luminoso a contatto con una superficie si scompone in due componenti, una riflessa ed una rifratta. Con un certo angolo di incidenza però la componente rifratta è nulla.
La somma energetica fra queste due componenti è pari al raggio iniziale.

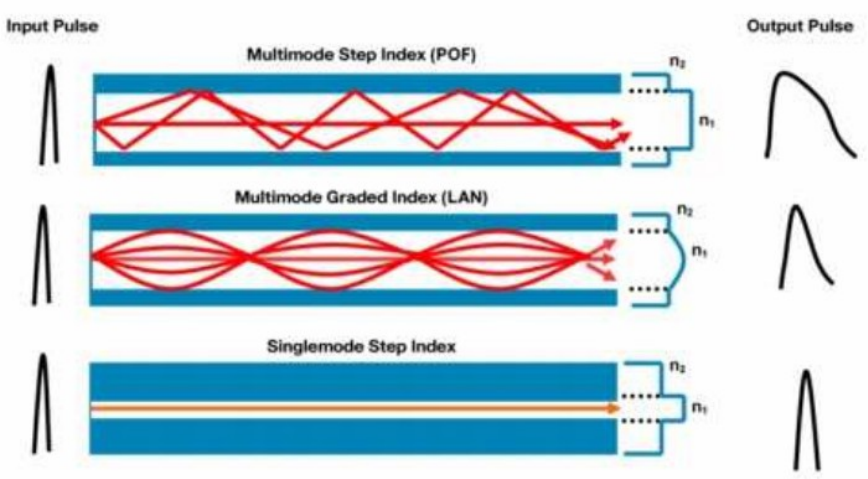


L’idea della fibra ottica è quella di realizzare un tubo in maniera tale che il raggio luminoso che entra dentro la parte interna, vi rimanga intrappolato.
La fibra ottica viene realizzata esternamente da una guaina in plastica, poi c’è una struttura in Kevlar che permette di dare resistenza alla trazione, poi un tubo di plastica detto cavidotto, un’altra protezione detta coating ed infine abbiamo la fibra vera e propria la quale è realizzata in vetro. Quest’ultimo viene ridotto a diametri molto piccoli da poter essere flessibili.
La fibra si comporta bene con tutte le frequenze del visibile sepppur con qualche differenza.
Alcuni cavi in fibra ottica sono realizzati da decine di cavi in fibra, tutti separati tra di loro: in generale non si hanno particolari disturbi elettromagnetici fra i cavi e nemmeno disturbi ottici eccessivi.
Per giuntare due fibre ottiche dobbiamo assicurarci che esse siano allineate: se non sono perfettamente in asse potrebbero esserci dei malfunzionamenti.
Al fine di effettuare l’operazione in maniera precisa si usa un’apparecchiatura dotata di microscopio. Per prima cosa, si tagliano con una piccola ghigliottina i due cavi; successivamente si può procedere con una lisciatura al fine di eliminare qualunque increspatura che potrebbe deviare il segnale; una volta posizione si fa scoccare un piccolo arco elettrico che fonde istantaneamente i due cavi nel punto prestabilito.
Lo strumento infine permette anche di verificare quanta energia si perde a causa della giunzione.
Essendo un canale anche la fibra ha i problemi dei canali in rame però ha anche alcuni aspetti positivi.
Il fatto di essere un mezzo fisico dielettrico(non è propenso ad essere attraversato da corrente elettrica) fa si che non ci siano interferenze elettromagnetiche: quindi possiamo mettere due fibre vicine senza che generino interferenze.
Una delle cause di interferenza è data della natura corpuscolare della luce,: i fotoni si scontrano con gli atomi delle strutture molecolari del vetro e generano delle piccole

riflessioni. Per cui parte dell'energia che passa nella fibra viene dissipata, e si disperde in vari modi come il calore. Nonostante questo la fibra permette di avere un rapporto S/N abbastanza elevato, sicuramente migliore di altri mezzi trasmissivi.
D'altra parte l'uso della luce permette di avere un bitrate elevato grazie ad una maggiore frequenza del segnale(H): questo ci permette di migliorare il risultato nella formula di shannon e quindi di aumentare il tasso trasmissivo, arrivando ad ottenere in laboratorio fino a valori nell'ordine dei 50000-100000Tbps.
Nell'uso pratico si possono avere collegamenti nell'ordine dei 10Gbps du distante di chilometri senza necessità di ripetitori.

Quale mezzo luminoso viene usato?
Le lampadine a incandescienza hanno una velocità di commutazione(tempo di accensione/spegnimento) elevatissima rispetto alle potenzialità della fibra ottica.
LED: molto economici ma risentono del problema che la luce viene emessa in tante direzioni e non è centrata. Hanno una velocità di commutazione discreta per cui potrebbero essere utilizzati per basse velocità.
Laser: la luce emessa dai laser invece è centrata(coerente), cosa che di fatto ne amplifica la potenza(tanto da poter bruciare una retina umana).

Fibre Multimodali



La prima è una fibra tradizionale con una netta separazione tra il core interno ed il cladding esterno. Questo fa si che ci siano delle riflessioni marcate ed il raggio emesso non risulta perfettamente in asse con la fibra ottica: questo può causare il loro reciproco annullamento o in generale dei disturbi.

Nel secondo tipo si ha un andamento più continuo, ciò permette di raggiungere velocità superiori perché migliora il rapporto S/N data la diminuzione delle interferenze.
Entrambi i tipi presentano di fatto più raggi riflessi contemporaneamente e questo aumenta le riflessioni.
Hanno un diametro di 50micron.
Fibre Monomodali
L'ultimo tipo invece è una fibra monomodale: ciò si traduce in un raggio luminoso singolo, senza riflessioni e centrato, che consente di diminuire il rumore, coprire distanze maggiori ed aumentare il bitrate. Sono però più costose.
Il core ha un diametro di 8 micron.

Il costo nell’utilizzo della fibra ottica non deriva tanto dal cavo in sé, ma è dovuto all’installazione, che richiede dei mezzi e competenze specifiche.

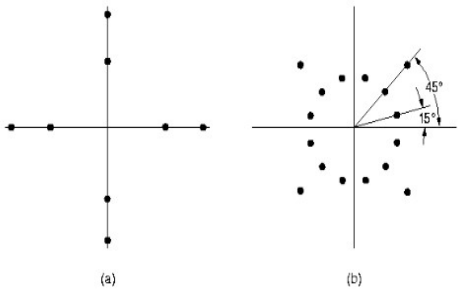
P.S: La fibra è unidirezionale.

Collegamenti Broadcast con Fibra Ottica
Il modo più semplice è quello di stabilire dei collegamenti punto-punto tra una macchina e la successiva, con un’interfaccia ottica che permetta di rilevare la luce, convertirla in segnale elettrico e poi nuovamente in un segnale ottico verso un’altra macchina. L’idea sarebbe quella di realizzare un anello in cui ogni macchina legge, ricopia e trasmette alla successiva.

Reti Wireless
Sono delle reti di comunicazione senza cavi di collegamento fissi.
La sifda maggiore che le reti wireless devono affrontare sono i disturbi cui la comunicazione è soggetta. Le reti wireless sono per definizione reti broadcast e soffrono parecchio il confronto con le reti cablate punto a punto.
Le onde utilizzate per le reti wireless sono quelle radio: AM, FM, trasmissioni LTE.
Al diminuire della frequenza, le strutture fisiche esercitano un ostacolo minore nei confronti delle onde elettromagnetiche.

Reti telefoniche
Le reti telefoniche sono state progettate per le comunicazioni realizzate tramite la voce umana per tale motivo non si prestano alle comunicazioni fra computer. Il canale veniva suddiviso in slot separati ed ogni slot assegnato ad un utente. In tal modo nello stesso filo potevano viaggiare telefonate differenti. La voce però veniva deformata a causa del taglio delle altre frequenze(cosa che avveniva mediante un filtro passa-basso) cosa che però non danneggiava più di tanto la comunicazione.

Modem
Modulatore-Demodulatore
L’utilizzo delle linee telefoniche(analogiche) avviene utilizzando i modem, che trasformano il segnale da digitale ad analogica e viceversa. Solitamente veniva usata una portante a 2100 Hz modulata entro ciò che il filtro fa passare. Dato che la frequenza era abbastanza bassa si rendeva necessario aumentare il numero di livelli per aumentare il bitrate(Nyquist).
Venivano utilizzate due differenti modulazioni: una portante in fase ed una portante in ampiezza. L’ampiezza è la distanza dall’origine mentre la fase è lo sfasamento di gradi.



Tali modulazioni possono essere così rappresentate.
Nello schema a sinistra abbiamo due livelli di ampiezza per un totale di 8 punti: 3 bit/baud.
Lo schema a destra invece ha sempre due livelli di ampiezza, ma tante fasi, una ogni 30°: 4 bit/baud.

Con 2400 baud(variazioni) è quindi possibile arrivare a 9600 bps. Questi schemi sono definiti costellazioni.
Per poter comunicare tra loro due modem devono utilizzare la stessa costellazione.
Le costellazioni sono poi andate a evolversi con numeri di livelli di ampiezza e fasi sempre maggiori.
Quando inviamo un FAX, spesso viene emanato un fischio o rumore di fondo: questo rumore permette ai modem di indentificarsi tra loro e rappresenta la portante.
Si parte da una velocità più bassa che piano piano va ad aumentare fino alla massima possibile.

DSL
Uno dei problemi dei modem era il fatto che utilizzavano il canale telefonico e quindi il telefono era inutilizzabile. Con le linee DSL si è deciso di lasciare una parte del canale riservata alle comunicazioni telefoniche ed un piccolo spazio vuoto per evitare disturbi: il resto del canale è stato poi suddiviso in blocchi da 4K utilizzabili da più modem in parallelo tra di loro. Quindi il modem DSL permette agli altri dispositvi di comunicare, utilizzando tutti i canali ad eccezione di quelli bassi.
I vari blocchi furono poi suddivisi in blocchi di downstream e di upstream, con l'idea che il primo dovesse essere più veloce.
Lato telefonico, un filtro passa basso eliminava tutte le alte frequenze, lato modem invece un filtro passa-alto eliminava tutte le basse frequenze: in tal modo i due reparti non interferivano fra loro. Per accoppiarsi con un modem DSL, la centrale telefonica usava un dispositivo determinato DSLAM.
ADSL è chiamata così proprio per questa asimmetria tra downstream/upstream mentre nella XDSL sono uguali.
Ci sono poi anche le linee VDSL che propongono velocità superiori: sono collegati ad armadi di strada(Cabinet) e dopo proseguono in rame.