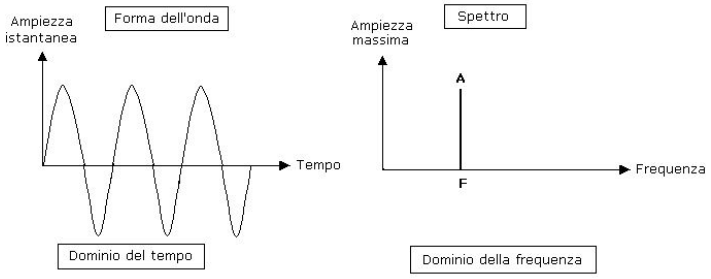
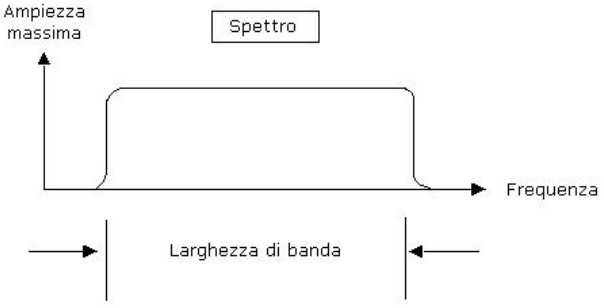
Reti di calcolatori, parte 1
Lezione 1
Segnali
Nell'ambito dei sistemi di comunicazione è necessario usare i *segnali*.
È possibile creare segnali anche analogicamente, non traducendoli in bit (si pensi alla voce). Per effettuare la trasmissione si effettua quindi una trasduzione del segnale (trasmissione senza digitalizzazione).
Il segnale analogico può essere

- periodico (ha la proprietà che il valore che assume all'istante "t" è uguale al valore che assume all'istante t + T, dove T sarà il periodo)
- non periodico.

La frequenza è l'inverso del periodo (1/T) e si misura in hertz
→ un segnale che si ripete ogni secondo ha 1hz, una onda che oscilla 1000 volte al secondo ha una frequenza di 1Khz
Le onde possono essere curve, quadre, triangolari, a dente di sega.
La fase è il momento in cui si inizia la misurazione rispetto all'inizio del segnale.
Teorema di Fourier: ogni onda periodica può essere rappresentata come somma (potenzialmente con infiniti termini) di onde sinusoidali di frequenze diverse.
Armoniche: Sinusoidi di frequenze diverse che comporranno il segnale, e avranno frequenze multiple rispetto al segnale originale.
Spettro discreto di un segnale: rappresentazione del segnale sul dominio delle frequenze. Ogni componente del segnale viene espressa come una linea verticale con altezza pari alla sua ampiezza nella posizione relativa alla frequenza



Larghezza di banda: identifica l'intervallo di frequenze dove il 90% del segnale viene coperto (l'intervallo tra la prima "linea verticale" e l'ultima sul grafico con la frequenza sull'asse X



I materiali dove viene fatto passare il segnale hanno diverse reazioni, che possono filtrare e/o bloccare il segnale.
→ In tutti i casi esiste un segnale periodico (***sequenza di benchmark***) la cui larghezza di banda sovrastima quella di qualunque segnale digitale non periodico, e considerando quella larghezza di banda si può stare tranquilli che passeranno tutti i segnali non periodici più piccoli del periodico

Codifica dei dati
Modem: Modulatore/demodulatore. Prende il segnale digitale di un PC (già tradotto da una dimensione statica variante nello spazio a una dimensione variante nel tempo (un bit alla volta) ed esegue un'operazione di moltiplicazione dell'onda quadra per un segnale sinusoidale (detto *portante*).

Ciò trasla lo spettro in avanti della frequenza della portante. La demodulazione sarà l'operazione opposta. Ogni modem (o codec, in caso i modem abbiano la stessa portante, così da cambiare il segnale prima di entrare sul trunk) potrà moltiplicare il segnale per una portante differente (***FDM***, Frequency Division Multiplexing), così che non si incroci con i segnali degli altri computer, così da poter andare insieme su un ***trunk*** (cavo multisegnale).
Altre tecniche di trasmissione sul trunk ***telefonico*** sono:

- ***WDM*** (Wave Division Mul. - Fibra)
- ***TDM*** (Time Division M.)
- ***PCM*** (modulazione a codifica di impulsi)
- ***modulazione delta***

Esistono diversi tipi di modulazione:

- Modulazione d'ampiezza: moltiplicazione per la portante
- Modulazione di frequenza: vengono adottate diverse frequenze in base al dato da trasmettere
- Modulazione di fase: modulazione che fa crescere o decrescere in base ai cambiamenti di fase

Bitrate: numero di bit trasmessi al secondo (b/sec)
Baud Rate: numero di variazioni al secondo trasmesse. (segnali/sec)
È il baud rate che governa lo spettro, e nell'onda benchmark bitrate = baud rate

La codifica di un segnale digitale nella trasmissione è diversa dalla codifica nello storage.
Le principali tecniche di codifica sono:

- NRZ
  - Bit 1 → segnale alto
  - Bit 0 → segnale basso
- NRZ-I (baud rate < bitrate)
  - Bit 1 → Transazione all'interno della cella di bit (si passa da 0 a 1 o da 1 a 0)
  - Bit 0 → Nessuna transazione all'inizio della cella di bit
- Manchester (baud rate > bitrate, al peggio doppio)
  - Bit 1 → basso-alto
  - Bit 0 → alto-basso

La ***basilare (fondamentale)*** ha baud rate e bitrate uguali.

COMUNICAZIONI TELEFONICHE
Nelle comunicazioni telefoniche, il multiplexing (trasmissione contemporanea di segnali diversi separabili alla destinazione) permette, tramite la moltiplicazione per segnali portanti diversi, di immettere tutte le comunicazioni nello stesso trunk.
Nelle reti di calcolatori non è così. La portante per cui il segnale viene moltiplicato è uguale per tutti, e per funzionare si impone che si possa utilizzare il cavo di comunicazione in comune ***a turno*** (SDM, Statistical division multiplexing, alternativamente, dare ad ogni coppia di comunicanti un cavo dedicato, ossia lo switching).
Le portanti dati usate dai telefonini sono comunque diverse tra loro.

Nel caso di segnale luminoso non si può moltiplicare per una portante, ma si potranno usare dei laser differenti (tramite l'utilizzo di fibre multicromatiche, dove potranno andare più colori di laser). Utilizzando un segnale elettrico sarà possibile modulare il segnale laser.
Nel caso si voglia utilizzare delle fibre monocromatiche, un modo per eseguire il multiplexing è l'uso del "*Time-division multiplexing*". Qui si opterà per trasmettere a turno dalle varie sorgenti (che sono "lente") un bit. (quindi se ci sono 3 sorgenti, si avrà un bit della determinata sorgente ogni 3, quidni ogni segnale manda 1 unità ogni 3T secondi). Se una sorgente è spenta, sarà necessario trasmettere dei bit di "idle" ai relativi turni.

In un TDM si ha una *tramatura*: le varie sorgenti sono aggregate, e si va ad assegnare uno spazio di bit alla sorgente che è la centralina di aggregazione, che inserisce dei bit di controllo (bit di signaling). Tra questi bit di tramatura ci dovranno essere anche le informazioni che riguardano il destinatario (sarà il numero digitato a dire cosa scrivere nei bit di tramatura). Quindi il bitrate di un cavo TDM è meno di N volte il bitrate delle sue sorgenti, in quanto la centrale si riserva alcuni bit per delle informazioni di servizio, utili per comunicare con la centrale di destinazione (es. standard SONET).

Un'altra operazione eseguita dalle centrali di zona è l'operazione di "toll", ossia di scatto, che fa partire la tariffazione in base a dove si trova il destinatario.
Storicamente, esiste un modo per impostare i bit di tramatura in modo da non far partire l'addebito (storicamente il numero era rappresentato come una sequenza di toni di frequenza in hertz diverse). Questo era possibile frapponendo dei toni che rappresentassero le impostazioni di centralina. Questo concetto si chiama "in-band signaling".

COMUNICAZIONI MOBILI

Il funzionamento della portante per i segnali da cellulari si basa su due risultati matematici:

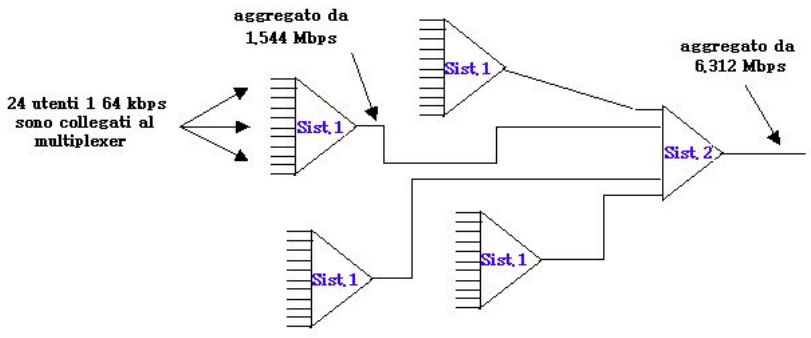
- il teorema di Viterbi (vattelo a guardare su wikipedia, cit.)
- il teorema dei quattro colori

Quest'ultimo dice che una mappa per essere colorata in modo che due stati adiacenti non abbiano lo stesso colore, sono sufficienti 4 colori.
Questo teorema permette il riutilizzo delle portanti, e cioè ogni cella gestisce un pettine (insieme) di portanti che è diverso dai pettini gestiti dalle celle adiacenti, ma che sarà possibile riutilizzare nelle celle non adiacenti.
Nella pratica, con l'aumento della densità di telefonini per una singola cella, si cerca comunque di riutilizzare la stessa portante (ad esempio, affidando la stessa portante a due telefonini posizioni diverse all'interno dell'area della cella). Inoltre esiste una tecnica chiamata CDM che permette di utilizzare un codice identificativo del cellulare, così da usare tecniche a livello di cella per recuperare segnali parzialmente sovrapposti.
Il procedimento di "*handover*" è il procedimento secondo cui, a seguito dello spostamento, il telefonino affida alla cella più potente attualmente la comunicazione, comunicando alla rete per quale cella deve passare la comunicazione. Ad avvenuto handover, la cella lo comunica al MTSO (stazione multicella) che provvederà a switchare le comunicazioni verso la nuova cella

MULTIPLEXING DIGITALE

Ogni offerta digitale da parte dei fornitori pubblici è multiplo di 64kbit (più piccola dimensione possibile), e che ogni aggregatore TDM ha un certo numero di flussi di 64kbit per costruire un flusso più veloce.
Esistono diverse categorie di segnali:

- DS-0: 64kbps
- DS-1 (o T1): 1.544 Mbps (24 DS-0 + 8 Kbps di tramatura/overhead)
- DS-2 (o T2): 6.312 Mbps (96 DS-0 + 168 Kbps di tramatura/overhead)
- DS-3: 44.736 Mbps (672 DS-0 + 1.728 Kbps di tramatura/overhead)
- DS-4: 274.176 Mbps (4032 DS-0 + 16.128 Kbps di tramatura/overhead)



Quindi la larghezza di banda dipenderà dal numero di canali DS-0 che la centralina mette a disposizione (minimo 1, e comunque soggetta ai canali che vengono occupati dagli altri, dove il caso migliore si avrà quando ci sono tutti gli utenti scollegati).

SONET

Standard ottico (in fibra, viene utilizzato un translator invece che modem) che prevede l'utilizzo di un fascio TDM su fibra, dove i bit vengono ottenuti tramite la modulazione della luce.
Chiaramente viene inserito un overhead di tramatura, per identificare ed estrarre i diversi contenuti in ingresso man mano che arrivano, anche per fare in modo che i vari convertitori da segnale ottico a elettrico.

Lezione 2

INTRODUZIONE AL LIVELLO DATA LINK

La soluzione adottata per permettere la comunicazione tra elaboratori è SDM (ossia l'imposizione dell'utilizzo del canale di comunicazione a turno).
Si ha quindi una condivisione del canale di tipo statistico, ossia tramite una randomizzazione di cerca di fare in modo che si utilizzi un po' tutti il canale di comunicazione.
Questo ha permesso anche di contenere i costi di produzione (schede di rete che facessero da modem sarebbero costate moltissimo).

Protocollo ALOHA e Slotted ALOHA

L'idea è nata dalla necessità di trasmettere informazioni tra le università nelle isole hawaii usando un trasmettitore a frequenza unica. Si decise che quando due campus si dovevano trasmettere info gli altri dovevano stare zitti. Si optò per l'utilizzo di un prefisso che identificasse per gli ascoltatori l'inizio di una trasmissione ("ALOHA")
→ non funziona perché la velocità di propagazione è limitata (la condizione ideale sarebbe un tempo di propagazione nullo), e quindi è possibile che al momento in cui si inizi la ricezione qualcun altro abbia già iniziato la sua e si sia ricevuto il messaggi.
Inoltre può provocare starvation (per questo si è proposto di dividere il tempo in intervalli - slotting -, dove ogni trasmissione non può durare più dell'intervallo).
Uno slotted Aloha con round-robin permette comunque un funzionamento, seppur inefficiente.

Protocollo CSMA/CD

Il protocollo Ethernet (chiamato Carrier Sensing Multiple Access / Collision Detection) è un protocollo SDB basato sul collision detection (e non avoidance, come Aloha), così da sopravvivere alla collisione (collisione: trasmissione sulla stessa portante nello stesso momento).
Si sta quindi ad ascoltare il canale in anticipo trasmettendo solo quando non si sente niente, la portante è libera (Carrier Sensing).
Se si sente il canale libero, ognuno può iniziare a trasmettere (Multiple Access) - da qui le collisioni, dovute sempre alla velocità di propagazione non nulla, che non mi porta ad avere un ascolto del canale real time.
Viene quindi continuamente paragonato il proprio bitstream al segnale in rete dopo la trasmissione: se sta avvenendo una collisione si ha un segnale sommato, non uguale a quello che sto scrivendo. Ciò permette di rilevare la collisione (Collision Detection).
Se si rileva la collisione, entrambi gli host smetteranno di trasmettere, e attenderanno un tempo random (quindi molto probabilmente diverso tra i due), e si rieffettua la trasmissione. Il frame trasmesso dovrà comunque avere una lunghezza massima.
Comunque l'estrazione casuale del numero (che poi è un insieme limitato di possibilità) comunque non garantisce ottime prestazioni con più di due host in collisione.

Le schede di rete moderne sono in grado di gestire il meccanismo delle collisioni, tuttavia oggi le reti locali sono fortemente switchate, e quindi si ha un cammino indipendente per ogni host collegato (come se ci fosse un cavo che collega fisicamente ogni calcolatore).

Reti ad accesso condiviso
In questo scenario si hanno nodi multipli sul medesimo collegamento fisico (coassiale)
Si hanno strutture fisiche a bus (con terminatori, ossia degli specchi che facevano tornare indietro il segnale) e ad anello (ring). Tutte le schede erano collegate con il bus o con l'anello.
TDM ed FDM non erano possibili su questo cavo (non aveva portanti multiple, come il trunk telefonico).
Si cercò di utilizzare uno slotted ALOHA con un meccanismo di assegnazione degli slot: invece che assegnare lo slot per stazione, si effettuava una assegnazione in due fasi (*token ring*):

- un giro dove ci si poteva prenotare tramite il token, chi aveva appena finito di trasmettere emetteva un segnale di libero, mentre chi voleva prenotarsi metteva un segnale con il suo numero
- al secondo giro, se chi voleva prenotarsi rivede il suo numero, allora capisce che è il suo turno

La complessità della circuiteria fece decadere questa tecnologia (col passar del tempo, le schede token ring rimasero molto costose, mentre quelle ethernet no).

Ethernet

IEEE 802.3 (CSMA/CD - Ethernet): Standard storico ancora utilizzato. Scomposto in due parti:

- una parte di definizione della lunghezza dei frame e la sua struttura
- una parte di definizione il protocollo di accesso al mezzo (Medium Access Control, MAC, che utilizza a tal proposito il MAC Address)

Utilizza la codifica manchester.
802.3 è uno standard storico per via del bitrate, in quanto questo è stato bloccato a 10Mbps per molto tempo. Questo valore proviene dal concetto di "integrità del frame" (il bitrate è legato alla lunghezza del frame e alla lunghezza massima del cavo).
Se il cavo è molto lungo, può capitare che non ci si accorga che un altro ha cominciato a trasmettere (non si rileva la collisione), finendo la trasmissione prima, così che il destinatario possa ricevere dati errati senza ritrasmissione. Il passare dallo standard classico a quelli moderni significa quindi ridurre la lunghezza dei cavi.
802.3u e 802.3z hanno rispettivamente bitrate 100Mbps e 1Gbps (multipli di 10).
Per avere reti molto veloci senza curarsi della lunghezza, bisognerà eliminare ogni condivisione, ricorrendo allo switching.

Hub
Soluzione che permette di sostituire il cavo di connessione. Incrementò notevolmente il successo di ethernet, perché permise di sostituire il bus fisico con una stella fisica. Non risolve le collisioni (funziona esattamente come i cavi) ma consente di avere un ambiente di collegamento molto più comodo. Mantiene comunque il concetto di "bus logico".
Quando un hub riceve un segnale su una porta lo copia su tutte le altre (funzionando anche da ripetitore di segnale).
La regola 5-4-3 è una regola pratica dice che qualunque coppia di host connessi via Ethernet possono esserci al più cinque segmenti connessi da 4 repeater, di cui al più 3 popolati (due macchine sullo stesso hub sono sullo stesso segmento, due PC collegati a due hub collegati tra loro sono due segmenti diversi).
Questa regola è utile per costruire una rete "storica" senza violare le condizioni di integrità del frame.
Quando si deve considerare delle reti collegate in hub, il numero di metri per cui si può estendere il cavo non fa più fede, in quando bisognerà valutare le costanti di ritardo introdotte dall'utilizzo degli hub.
L'errore di perdita dei frame (quindi il mancato rilevamento delle collisioni) non è facile da individuare, sembra un errore di software.

Problema della rilevazione
Il fulcro sta nell'impedire ad A di terminare di scrivere prima che si accorga che il segnale è sovrapposto.
Il tempo T di propagazione da un host a A un host B equivale a L / V se c'è un cavo, altrimenti è uguale al tempo di ritardo del Hub. Quindi il tempo di ritorno del segnale ad A corrisponde a 2T.

Lo standard IEEE 802.3 specifica che il valore massimo di 2T deve essere 51.2 microsecondi, che corrisponde ad una distanza massima di 2500m.
A 10Mbps richiede 0.1 microsecondi per trasmettere un bit, quindi 512 bit (64 byte) richiedono 51.2 microsecondi
→ i frame Ethernet **devono** essere più lunghi di almeno 64 byte. Questo valore permette quindi difar sì che il segnale possa andare e tornare dal cavo.

Struttura del frame
802.3 impone che un frame abbia:

- 14 byte di intestazione (mac address destinatario, mac address mittente, preambolo - sequenza bit preordinata che serve al destinatario per capire che sta cominciando un frame)
- 46 o più di byte di dati (padding di 0 se non ci sono dati)
- 4 byte di CRC

Quando mittente e destinatario si accorgono della collisione non possono smettere di trasmettere (potrebbe lasciare spazio ad altri), e quindi per questo viene scritto un numero convenzionale ("segnale di jam", o segnale di disturbo, di 48 byte). Solo dopo la trasmissione del segnale di jam si sorteggerà il numero casuale per l'attesa e proseguirà la comunicazione.

Estrazione valore casuale
Si usa la tecnica di exponential backoff,e cioè ad ogni collisione ripetuta viene raddoppiato il numero di elementi nell'insieme dei numeri estraibili (1ª volta K = { 0 , 1 }, e l'attesa sarà K * 51.2 microsecondi, la 2ª volta si sceglierà K tra { 0, 1, 2, 3 }, e così via). Dopo 16 tentativi (K = 1023) ci si ferma: le schede sono fallate.

ETHERNET SWITCHATA

L'operazione di switching consiste nel determinare la porta d'uscita per ciascun frame ethernet sulla base dell'indirizzo MAC del destinatario.
Lo switch prevede che ogni porta sia collegata a tutte le altre. Introduce comunque dei delay (ad esempio, a causa dell'accodamento di uscita su una porta, dove si utilizzeranno dei buffer).
All'arrivo di un frame nello switch, viene esaminato l'indirizzo mittente nel messaggio e lo associa alla posta da cui il frame proviene (adesso questa attività viene svolta al momento del collegamento della scheda allo switch, in quanto la scheda fornisce già in via preventiva il suo mac - "***autopresentazione***" o "***negoziazione***").
Ogni switch si tiene le varie coppie <MAC address, porta> in una memoria apposita detta "CAM", content addressable memory. Ciò significa che l'indirizzo del destinatario del frame in arrivo viene posizionato in buffer che dispone di un comparatore, un circuito complesso che in un colpo di clock riesce a comparare l'indirizzo con le righe presenti in memoria. Questo circuito avrà un N (porte) * 48 bit di ingresso, ed è questa la ragione per cui gli switch costano molto.
Lo switch controllerà poi controllerà se l'indirizzo destinazione nel frame corrisponde ad una delle righe della tabella, e in caso copia il frame sulla porta specificata da quella riga, altrimenti lo copia su tutte le porte.

Esistono degli switch chiamati "knock-out", dove ci sono delle parti dei percorsi in comune. Questo può portare a delle occasioni di collisioni, ma poche, solo in occorrenza di precisi cammini attivati contemporaneamente.
Le porte possono essere divise in gruppi in modo che gli host collegati a un gruppo di porte comunichino solo tra loro, formando delle Ethernet virtuali (VLAN).
Uno switch può essere raggiunto da un PC, e collegandosi si accede ad una interfaccia di amministrazione dove è possibile anche sapere a quale porta dello switch sono collegato.
Off-topic: sniffing = mettere in modalità promiscua (tramite una invocazione ad una funzione del driver della scheda) la scheda ethernet, ossia legge anche i messaggi con MAC di destinazione che non è il suo. In un'ethernet switchata non può esistere: i cammini sono indipendenti, e inoltre aumentando le VLAN si attuano delle separazioni fisiche vere e proprie.

FRAME DI ETHERNET

Il frame ethernet è composto da almeno 64 byte.

- 7 byte di preambolo, ogni byte impostato a “10101010”
- 6 byte di Mac Address di destinazione*
- 6 byte di Mac Address sorgente
- 2 byte di tipo:
    - 0x0800 indica che il campo dati contiene un datagramma IPv4
    - 0x0806 contiene un frame ARP
    - 0x8100 indica un frame IEEE 802.1Q
    - 0x86DD indica un datagramma IPv6
- 45-1500 byte Corpo (payload)
- 4 byte CRC (resto della divisione del messaggio per un polinomio convenzionale, che si riesegue all’arrivo per sapere se il frame ha subito alterazione)

Preambolo (7B)	MAC DEST (6B)	MAC MITT (6B)	TIPO (2B)	CORPO (46-1500 B)	CRC (4B)

Il mac è assegnato al momento della fabbricazione, e serve per la comunicazione nell’ambito della rete locale. Se si è su una rete basata su hub, serve per la scheda di destinazione per sapere il frame destinato a lei, mentre nella ethernet switchata serve allo switch per sapere su quale collegamento interno instradare il frame.
Il Mac address è un indirizzo di 48 bit (6 byte) che si scrive in esadecimale.
Esiste un indirizzo speciale, chiamato “indirizzo di broadcast” di valore FF-FF-FF-FF-FF-FF (tutti 1) che si usa per raggiungere tutti i computer della rete.
Questo nelle ethernet switchate farà in modo che sia copiato il frame su tutte le porte di uscita.

Il campo tipo è una chiave di demultiplazione, che serve per stabilire a quale protocollo di livello più alto (es. ip) il frame è diretto, così che in base al particolare tipo del pacchetto contenuto nel corpo si avrà una valorizzazione del campo tipo.
Il campo corpo può contenere fino a 1500 byte.

TIPI DI SWITCH

- Switch a configurazione fissa, che hanno un numero fisso di porte Ethernet (fino a 46)
- Switch modulari, che hanno architetture interne che gli consentono l’aggiunta successiva di porte.
    - Per questi è possibile impilare e collegare tra loro certi modelli di switch a configurazione fissa, la il collegamento tra i due introduce un collo di bottiglia

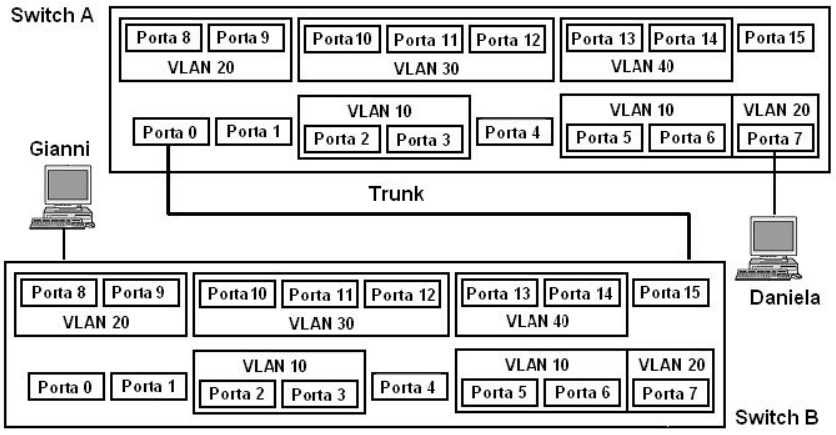
VLAN

Permette di realizzare multiethernet avendo un unico switch (o anche una ethernet cross-switch).
È possibile far comunicare due VLAN usando dei dispositivi collegati ad entrambe le VLAN (se il dispositivo collegato ad entrambe le VLAN, e che ha due interfacce di rete, esegue il ROUTING, cioè copia il traffico che arriva a lui (al suo mac) in uscita sull’altra interfaccia, collegata all’altra VLAN.
Questo implica che il contenuto del frame sia cambiato (estraendo il payload dal frame in arrivo e innestandolo in un nuovo frame indirizzato ad un altro calcolatore.
Questo ha senso nel momento in cui si vuole ottenere un “***confinamento del traffico***”. Il router, in questo caso, eseguite la comunicazione selettivamente in base al contenuto del frame, creando delle condizioni di instradamento, mentre nello switch non ci sarebbero possibilità di limitazione.
Oggi queste funzioni di routing possono essere già implementate, così da abilitare il routing e far comunicare le VLAN, inoltrando i frame in base a condizione decise su misura.

Trunking

È possibile configurare le VLAN perché siano cross-switch, così che la VLAN abbia una o più porte su più switch. Questo permette di far corrispondere la VLAN alle unità organizzative (contabilità, sviluppo…), però

prescindendo dalla locazione fisica (ad esempio, una VLAN “aule” può dover avere porte su switch su più piani).
A tal proposito si usano le porte di trunking, ossia porte speciali che fanno in modo che il traffico generato in determinate porte dello switch venga copiato sulle porte dell’altro switch che hanno VLAN con lo stesso numero. Il collegamento tra switch avviene tramite il cavo di trunking, che non osserva un protocollo ethernet, in quando occorre che il traffico generato venga copiato sull’altro switch.



(In questa figura Gianni può parlare con Daniela)

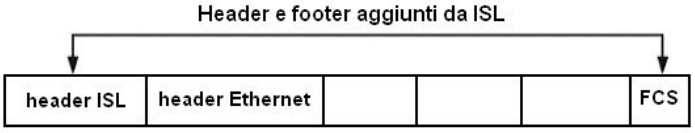
Quello che correrà sul cavo di trunking sarà il traffico generato sullo switch A che riguarda VLAN che sono definite anche sullo switch B, e per far questo è necessario inserire l’identificativo della VLAN

- Esercizio che c’è sempre all’esame -

Per sapere le porte su cui copiare i frame sarà necessario impacchettare i frame e inserire l’identificativo della VLAN (da qua, le differenze con ethernet, e il perché è stato definito un protocollo specifico per trunking).

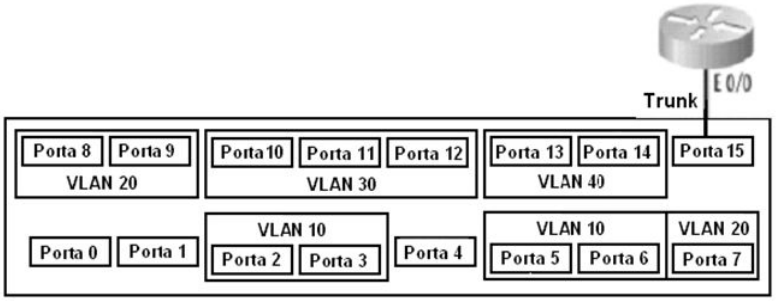
Protocollo di trunking

Protocollo di livello 2, che incapsula il traffico generato su uno switch per un certo VLAN number e diretto ad un altro switch dove è definito uno stesso VLAN number.
Esistono due tipi di protocolli di trunking:
1) Incapsulamento: viene aggiunto uno header al frame ethernet per indicare la VLAN di destinazione



Lo switch di destinazione utilizzerà l’header di incapsulamento per sapere su quale porta specifica (tra quelle della VLAN con lo stesso numero) copiare il frame.
Questo protocollo ha il nome di ***ISL***, e i pacchetti che girano su queste porte ISL sono frame SL (e non ethernet!).
*NB: L’incapsulamento che avviene su uno stesso livello prende il nome di “tunneling”
2) Piggyback (IEEE 802.1Q) → modifica al frame ethernet per inserire tra indirizzo sorgente e tipo (aggiungendo 4 byte prima del campo tipo) per scrivere il numero della VLAN. (2 byte → 0x8100, 3 bit per la priorità, 1 bit per dire se il frame è scartabile in caso di congestione e 12 bit per l’ID della VLAN)

Vantaggi e svantaggi: piggyback non perde nulla di banda (ISL aggiunge dei bit che mangiano banda), ma fa perdere tempo in termini computazionali, in quanto bisognerà ricalcolare i CRC per via delle nuove info inserite.
=> Lo stesso risultato che si ottiene facendo routing tra due o più VLAN diverse si può ottenere collegando il router alle porte di trunking (“router-on-a-stick”), in quanto questo vedrà tutto il traffico diretto tra le VLAN e decidere quale instradare.



Lezione 3

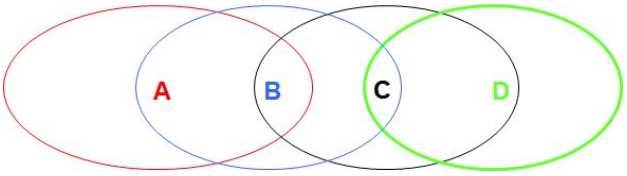
SOTTOLIVELLO MAC

Sono esistite molte proposte di soluzioni che usano tecniche MAC diverse da CSMA/CD.
Ci sono pervenute quindi delle strategie miste, ad esempio la divisione delle stazioni in gruppi, di cui ogni ognuno può contendersi uno slot diverso (mix tra mac 802.3 e ALOHA slotted o 802.5 - token con prenotazione-):

- nel caso limite in cui ogni stazione è membro di un gruppo formato da solo sé stessa, ogni gruppo avrà uno slot, e quindi si potrà usare ALOHA slotted o token con prenotazione
- nel caso limite in cui tutte le stazioni fanno parte di un gruppo → CSMA/CD
- nel caso intermedio si sceglie la dimensione del gruppo a seconda del carico (il protocollo sarà adattivo). All'interno del gruppo si fa CSMA/CD. Si mira ad ottenere un MAC intelligente che gestisca in maniera adattiva i gruppo, massimizzando il throughput, e abilitando lo slot adattivamente, permettendo che si operi CSMA/CD solo quando il gruppo è abilitato.

Questi tipi di proposte sono ancora attuali per quanto riguarda il discorso dell'onda libera (Wi-fi).
Il Medium Access Control è l'alternativa statistica al FDM per le onde libere, in quanto i sistemi che utilizzano FDM devono avere una riserva di banda per garantire la diversa portante per tutti. Un access point intelligente cercherà di capire qual'è la dimensione giusta dei gruppi, in base alle collisioni che si verificano (dividendo i dispositivi progressivamente in più gruppi, che saranno più gestibili in termini di collisioni).

Il MAC a contesa funziona nel caso tutte le stazioni siano collegate ad un cavo di lunghezza consona, in modo da non violare l'integrità del frame. Tuttavia bisogna imporre l'ipotesi che ogni stazione riceva le emissioni di ogni altra. *Questo nel cavo è valido* in quanto il cavo è disposto come anello o con gli specchi di ritorno, ma nell'onda libera ciò potrebbe non esserci. Per garantire la situazione dove il protocollo di contesa funziona, il range di ogni dispositivo dovrebbe coprire tutti i dispositivi connessi, diversamente potrebbe non funzionare.



Ciò può portare a diversi problemi:

- Problema della stazione nascosta: quando A trasmette a B, C percepisce il mezzo libero
- Problema della stazione esposta: quando B trasmette ad A, C percepisce il mezzo occupato e non trasmette a D

A tal proposito 802.11 ha dovuto investire banda per due opportuni segnali che facessero da riserva esplicita (non essendoci specchi o strutture ad anello):

- RTS (Request to send): Richiesta del permesso per inviare che parte dalla stazione che vuole parlare, che tutte le stazioni dovranno propagare, dove tutte le altre stazioni dovranno stare in silenzio fino all'arrivo di un CTS (come fosse un token)
- CTS (Clear to send): alla ricezione, la stazione che ha emesso il RTS avrà il permesso di parlare perché tutti gli altri staranno in silenzio (la stazione che voleva parlare avrà ottenuto uno slot per comunicare)

Questo *non è quindi definibile* un ethernet senza cavo, né uno slotted, ma piuttosto un MAC con riserva esplicita.

DATA LINK E ISO/OSI

Il livello fisico definisce l'interfaccia fisica tra il dispositivo di trasmissione dei dati (es. PC) e il mezzo di trasmissione o la rete. Definirà quindi la modulazione, la frequenza (o le) portante/i, il tipo di guida d'onda (o se è onda libera), il bit rate, caratteristiche del mezzo di trasmissione.
È una definizione a 4 livelli:

- meccanica
- elettrica
- funzionale
- procedurale (le procedure per il testing e la stesura, ad es. quanto piegare i cavi e cose così)

Attualmente il livello fisico delle schede è stabilito in fase di produzione, tuttavia si va verso una definizione di questo tramite livello applicativo (software defined networking).

Il livello data link è composto da MAC (protocollo di accesso al mezzo) e LLC (definizione del frame).
Il Data link può andare su più tipi di livelli fisici (cavo ethernet, doppino, coassiale…). Quindi la definizione degli standard 802.x definiscono il livello di data link. È un livello "LOCALE".

Internet nasce come sistema software per far parlare tra loro macchine che sono su ethernet diverse, ma collegate tra loro (un'alternativa concorrente alla rete telefonica).
Il modo per far uscire un frame dalla ethernet consiste nell'utilizzo di nodi intelligenti che prelevino il contenuto del frame e lo mettano dentro un frame di altro tipo, da instradare verso la destinazione esterna. Questo sistema è detto ***incapsulamento*** ed è il sistema alla base della comunicazione tra un livello e l'altro.
Inizialmente ISO/OSI prevedeva 7 livelli:

- Fisico
- Data link
- Rete
- Trasporto
- Sessione
- Presentazione
- Applicazione

Tuttavia, ad oggi si utilizza un modello semplificato (i livelli di sessione, presentazione e applicazione sono realizzati dal software applicativo senza fare formali incapsulamenti).
Dunque si tratterà il modello TCP/IP in 5 livelli:

- Fisico
- Accesso di rete (che il livello di rete locale 802.x o un livello sincrono di rete pubblica)
- Standard di rete IP (Liv. Internet)
- Standard di trasporto TCP (Liv. Trasporto)
- Livello applicativo con i relativi protocolli applicativi

Il sistema operativo implementa gli standard di incapsulamenti. Il messaggio, prima di uscire dal livello fisico, subisce diversi incapsulamenti dal SO, e questi sono critici in termini di throughput punto-punto.

STRUTTURAZIONE GERARCHICA

Importante per esame: spesso potrebbe essere richiesto di progettare VLAN in situazioni dove le unità fisiche e le unità organizzative (ambiti di traffico) non coincidono, e quindi fare esempio di frame in transito sul cavo di trunking.
Es. progettare VLAN per campus con due capannoni che ha come ambiti di traffico in uno il magazzino, e nell'altro amministrazione e magazzino, e mostrare esempio di frame su cavo di trunk.

- Si dovranno creare 2 VLAN (per due ambiti organizzativi di traffico) → fare una tabellina

- Pensando di avere 2 switch si 32 porte, si dedicheranno 16 porte (G1/32 - G16/32) per l'amministrazione e 16 porte (G17/32 - G32/32) per il magazzino nello switch 1, mentre nello switch 2 si dedicheranno tutte e 32 le porte (G1/32 - G32-32) alla VLAN del magazzino, e i due switch saranno collegati da un cavo di trunking

Es. di assegnazione VLAN

VLAN	NOME
10	IT
20	Personale
30	Contabilità
40	MagazzinoMilano
50	MagazzinoRoma
60	Spedizioni
70	SedeCentrale
80	Ricezione
90	Laboratorio
100	Produzione

- Per mostrare l'esempio di frame, bisognerà "effettuare un'ipotesi" dove si dovrà dire quale protocollo applicare, e quindi fare il disegnino del frame
    - CISCO ISL: Disegnino frame con Header con scritto VLAN "2" + trailer
    - Piggyback: Inserimento dei 4 byte nel campo tipo del frame

→ Proseguendo con la teoria
Difficilmente la situazione reale sarà così semplicistica ("rete piatta"). Se la situazione si complicasse non si potrebbe usare il trunking, ma si opterebbe per una scelta di livello 2.
In questo caso, si useranno degli switch unicamente per il traffico di trunking (operando una gerarchizzazione). Quindi vi saranno degli switch di accesso a cui saranno connessi generatori di traffico, mentre ci saranno degli switch di distribuzione a cui sono connessi gli switch di accesso.
Gli switch vengono quindi partizionati in gruppi, o domini (VTP domain), e gli switch di distribuzione saranno i capi del gruppo di switch accesso.
Ci può essere l'evenienza di implementare un terzo livello ("core switch" ad altissima velocità).
Tip: se si hanno da progettare più di 10 VLAN (unità organizzative), conviene comunque non rimanere in una rete piatta, in quanto le porte rimangono comunque limitate.

Il protocollo VTP è un protocollo inter-switch permetterà di cambiare la configurazione di associazione tra porte e relative VLAN sugli switch di accesso partendo dagli switch di distribuzione e propagandone le modifiche (si possono definire degli switch trasparenti che non accetteranno questa propagazione, e bisognerà impostarli manualmente)
> bello e utile! <

TOPOLOGIE RIDONDANTI

L'interruzione del funzionamento di un collegamento può provocare la mancanza di raggiungbilità tra stazioni, e causare un partizionamento di rete.
Si può pensare a percorsi ridondanti (che rimangono spenti, e vengono accesi quando un percorso viene a mancare, e che quindi funzionano alternatamente).
A livello 2 si hanno 4 tipi di traffico:

- Unicast conosciuto: l'indirizzo di destinazione si trova nelle tabelle degli switch
- Unicast sconosciuto: l'indirizzo di destinazione non si trova nelle tabelle degli switch → in questo caso il pacchetto viene duplicato ed inviato su tutte le porte dove sono presenti switch che non si sono "presentati", generando così un multicast

- Multicast: il traffico viene inviato a un gruppo di indirizzi (quindi un gruppo di porte dello switch) → viene eseguito quando ci sono delle porte a cui sono collegate delle macchine sconosciute, che non si sono presentate (se la macchina destinataria risponde, lo switch aggiornerà le sue tabelle di associazione)
- Broadcast: il traffico viene inoltrato a tutte le interfacce tranne quella di ingresso

I multicast possono creare dei problemi: è possibile che il multicast generi un ritorno del messaggio (se c'è un anello tra le connessioni degli switch) (*broadcast (o multicast) storm*), generando nella rete un traffico continuo di frame che vanno a rubare banda al traffico effettivo, ed è per questo che non ci devono essere cicli nel grafo (ogni switch deve essere raggiungibile tramite un unico cammino).
→ NB: A differenza di IP, nell'intestazione del Livello 2 non c'è TTL.

Tuttavia, si vuole avere ridondanza, per avere dei canali di riserva in caso un canale smetta di funzionare. Per questo si farà in modo di predisporre i cammini alternativi.
Per far questo si dovrà avere sì una topologia fisica con cicli, ma la topologia logica (ossia i percorsi attivi) dovranno essere uno "spanning tree" del grafo, ossia un albero che ha gli stessi nodi del grafo dove tra ogni coppia di nodi c'è uno e un solo cammino.
Per applicare ciò, alla topologia fisica ridondante, si applicherà un algoritmo a livello di switch così da spegnere delle porte per avere attivo un solo spanning tree sul grafo.
Al momento del guasto, si esegue un ricalcolo di uno spanning tree diverso, e attivare le relative porte degli switch.
Su un ambito distribuito, l'algoritmo non calcolerà lo spanning tree sulla matrice del grafo che dice come sono collegati tra loro gli switch della rete (di cui gli switch non sono a conoscenza), ma dovrà operare degli scambi di messaggi tra switch, ognuno dei quali conosce una riga della matrice di adiacenza del grafo.
Questi algoritmi prevedono i calcoli sulla sola parte dei dati reperibili da specifici agenti.
Ogni switch dovrà decidere quindi quali percorsi spegnere: viene scelta una radice, e quindi il mantenimento da parte di tutti della porta che ha la distanza minima verso la radice.
La radice viene decisa tramite il protocollo distribuito.

Lezione 4

SONET

L'ambito delle reti private locali è stato allargato dall'utilizzo di emulazioni di protocolli locali sulle reti pubbliche.
Il concetto sta nell'emulare reti ethernet fully switched per fare in modo che non ci sia il problema dell'integrità del frame.
Un percorso SONET va ad effettuare questa emulazione utilizzando dei cavi ottici, che simuleranno i cavi di trunking tra i vari switch. Quando si richiede una hyperlan per collegare del traffico di trunking tra gli switch, verranno dati degli scatolotti da collegare agli switch fisicamente distanti, che permetteranno l'ingresso alla fibra. Nel percorso tra gli statolotti di collegamento, vi saranno diversi pozzetti (terminazioni di sezioni) che permetteranno di creare tappe intermedie per il traffico (normalmente ogni 10km). Un cliente si potrà collegare solamente se sono presenti delle terminazioni di sezioni nella sua zona.
L'unità di informazione trasmessa sul cavo ottico prende il nome di "tramatura", in quanto contiene tutte le informazioni per quanto riguarda il terminatore di sezione nel quale deve uscire.
In genere, sulla rete pubblica il frame è più piccolo (810 byte), in quanto serviranno anche per trasmettere contenuti multimediali.

- Domanda esame -
È possibile utilizzare la tecnica del piggybacking per risolvere il problema di trasporto dello standard ethernet su rete pubblica?
→ No, il frame da mettere dentro è più grosso. La tecnica di inserimento di un frame ethernet in una trama di livello 2 pubblica è più grande di un frame ethernet. Si può però applicare la "frammentazione", ossia spezzare il frame in due frame con lo stesso header

Esistono dei casi in cui non si può effettuare la frammentazione, nel caso in cui la cella del livello 2 ospite sia così piccola che non ci stia nemmeno lo header. Al minimo, la lunghezza della trama di rete pubblica per trasportare un frame ethernet deve essere 64 byte (la dimensione minima di un frame). Operando la frammentazione è possibile (ma non conveniente) frammentare e trasportare minuscole parti di un frame ethernet per ogni trama di rete di pubblica (non conviene in quanto per ogni trama viene trasmessa lo stesso header e solo un ridotto frammento di informazione).

Esiste la possibilità di frammentare direttamente un pacchetto IP su trama pubblica, ma è estremamente inefficiente (questa è la differenza tra stack IP/Ethernet/SDH → questo frammenta il pacchetto in frame, e il frame su SDH; e IP/SDH → questo con frammentazione diretta su SDH)

LIVELLO DI RETE

I protocolli di rete locale, come ethernet, offrono un servizio di recapito di frame (solo) tra le schede collegate alla stessa rete locale. Il SO fornisce il servizio di trasformazione di dati memorizzati in uno stream di bit seriale da trasmettere sul filo (tramite il "supporto di rete").
Il software di rete è un servizio che viene avviato al boot del SO e che si occuperà di eseguire le chiamate di sistema necessarie per invocare il driver del chipset per generare il frame. Il programmatore potrà invocare i servizi del software di rete tramite delle librerie.
Quando un'applicazione deve trasmettere un dato lo passa al sw di rete che aggiunge un'intestazione (header) contenente delle info di servizio, ottenendo un pacchetto, e inoltra il pacchetto alla scheda di rete (la trasmissione dei dati avviene solo a livello fisico, tutti gli altri livelli si occupano di generazione degli header incapsulamento (che poi l'incapsulamento è concettuale, non eseguito fisicamente)).

Il livello 3 permette di avere due tipologie di connessione:

- ***reliable flow***: il mittene controlla se il destinatario è in linea tramite l'invio di pacchetti speciali, e il destinatario dovrà confermare al mittente la ricezione di ogni pacchetto tramite un acknowledgment.
- ***best effort***: si mandano i pacchetti di dati senza controlli o confermi

La velocità delle due fasi di generazione dei pacchetti, ed è molto diversa: in una chat la velocità di generazione dei pacchetti dipende dall'utente e può essere più bassa della velocità della scheda, mentre un server web può generare più pacchetti di quanti la scheda riesca a smaltirne.

Il software di rete è realizzato con una comunicazione bufferizzata tra processi consumatori e produtti (un livello genera un dato e lo mette in coda nel livello successivo. Questo genererà header e trailer, e metterà quanto ottenuto in un'altra coda, e così via fino alla scheda, dove viene convertito il dato in uno stream seriale).

Vi sono protocolli a ***header fisso*** (cioè tutti gli header hanno la stessa struttura): il software deve solo decidere i valori da inserire nello header per ciascun pacchetto. Ciò è semplice ma rigido.
Esistono poi protocolli ***header chain***, che definiscono un repertorio di header ammessi, che si possono concatenare: il software sceglierà tra gli header possibili quelli giusti per il pacchetto, permettendo di aggiungere informazioni in base alla singola tipologia di informazioni che si desidera trasmettere.
A liv. 2 ci saranno solo frame a header fisso, a livello 3 ci potrebbero essere dei casi di header chain.

*Nota di importanza su frammentazione: è possibile frammentazione di pacchetto IPv4 su frame ethernet perché lo header IPv4 è più piccolo della dimensione del frame!

Il protocollo non definisce chi deve ricomporre i frame frammentati, lo potrebbe fare anche il destinatario ma questo comporterebbe un notevole costo computazione, e per questo il compito viene affidato al dispositivo di ingresso alla rete del destinatario (bisogna avere però un router che sappia farlo).

All'interno dell'host mittente sono necessarie delle operazioni di copia per portare il dato da trasmettere (dallo spazio di memoria dell'applicazione a quello del kernel, e dallo spazio del kernel alla scheda di rete).

Queste richiedono l'utilizzo del bus di sistema, e questo potrebbe creare un ***collo di bottiglia interno***.

- Esercizio d'esame -
Il bit rate del collo di bottiglia interno è b / k dove
b → bit rate del bus di sistema
k (solitamente 2) è il numero di operazioni di copia richieste per inviare un dato.

Problema:
Se **B (bitrate nominale - visto dal mittente, ad es. 1 Gbps) > b / k** non stiamo sfruttando appieno la rete.

Es. un host con un bus interno PCI con bitrate di picco a 1056 Mbyte/s e k = 2 ha un collo di bottiglia a 4.24 Gbit/sec, troppo basso per una connessione Ethernet a 5 Gbit/sec

TECNICHE GATHER-WRITE

Si tratta di tecniche permettono di diminuire k (il numero delle copie). L'idea di base di questa tecnica (in genere implementata su macchine server e non client) è quella di non generare mai dei pacchetti, ma semplicemente di generare gli header e tenere nota dei puntatori alle loro zone di memoria, così da fare in modo che scheda e driver leggano questi puntatori e trasferiscano in un'unica operazione di copia tutto sull'interfaccia.
Questo risparmia la prima copia, e la seconda avviene sempre al peak rate del bus interno (PCI = 1056 Mbit/s). Lo stesso si può fare in lettura (scatter-read) - questo però potrebbe far in modo che dei dati non validi entrino nel sistema, in quanto il CRC viene verificato solo in seguito alla sua copiatura all'interno.

INDIRIZZI SOFTWARE

È possibile avere un livello 3 che abbia lo stesso ambito di recapito del livello 2 (le macchine raggiungibili sono solo quelle sulla stessa ethernet). Questo permetterebbe di non scrivere nello header di livello 3 l'indirizzo, o di copiare l'indirizzo a 48bit del livello 2. Ancora, con lo stesso ambito di recapito, si potrebbe inventare uno schema di indirizzamento di livello 3 che ha un mapping 1 a 1 con quello ethernet.
A tal proposito, esistono tanti protocolli di rete locale a livello 3. L'utilità sta nel fatto che ethernet a livello 2 garantisce solo il best effort, mentre il livello 3 permette anche il reliable flow, con le conferme di ricezione.

Se però si cambia l'ambito di recapito (volendo raggiungere macchine che non sono nella stessa ethernet, sia pure attraverso macchine router) bisogna cambiare indirizzo, in quanto gli indirizzi ethernet non hanno la ***proprietà prefisso*** (l'indirizzo ethernet è legato al produttore, quindi possono essere totalmente diversi anche nella stessa network → le macchine nella stessa ethernet non condividono nessuna parte dell'indirizzo → il protocollo ethernet non è instradabile).
Dunque la soluzione è stato adottare un indirizzo di instradamento di livello 3 di 32 bit. Questo indirizzo identificherà un'interfaccia software che viene usata dai programmi su di un host per comunicare con altre applicazioni su altri host.

Recapito

L'indirizzo software dispone della proprietà di prefisso.
Come fa il software di rete a generare l'indirizzo di ethernet corretto a partire dall'indirizzo software?
→ Se il destinatario è sulla stessa rete locale (questo si potrà determinare in base al prefisso, che corrisponderà a quello della rete locale), risponderà al broadcasting ethernet con un frame
→ Se il destinatario non è sulla stessa rete locale, si utilizzerà l'indirizzo MAC della passerella (router, o meglio "default gateway") che tutti gli elaboratori della rete sanno a priori. Questo è necessario per uscire dalla rete locale. Il router genererà quindi un nuovo frame da inoltrare:

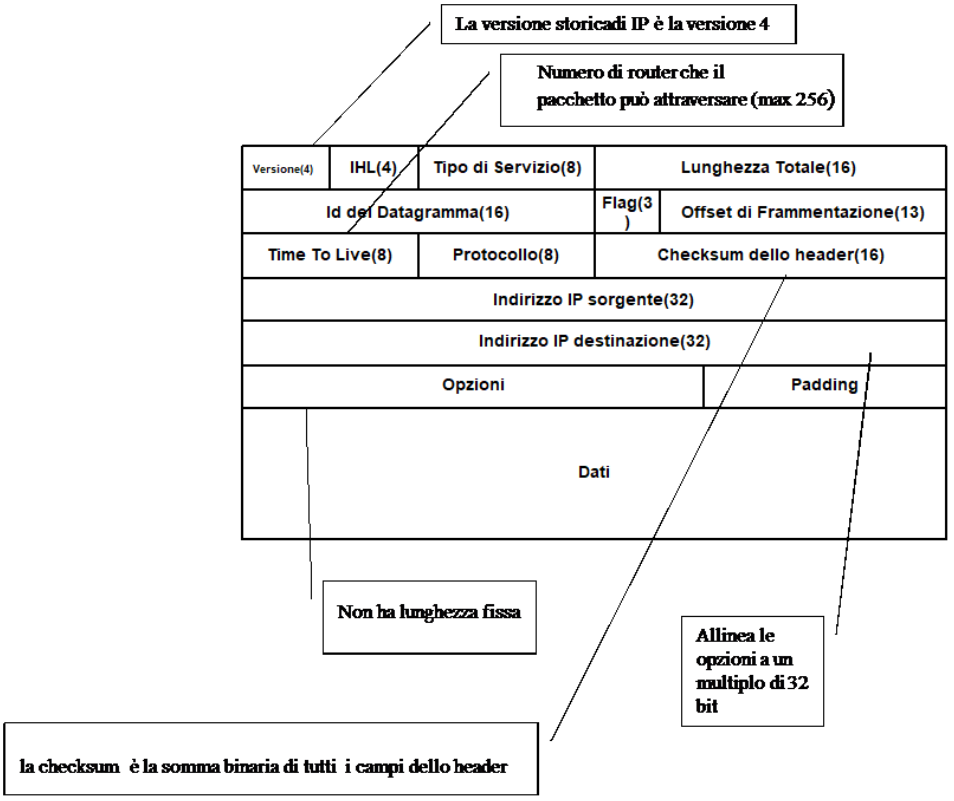
- al destinatario se il destinatario risponde al broadcast
- altrimenti ad un "collega" router collegato nella rete che farà la stessa cosa

Il router si baserà sul prefisso per decidere verso quale rete a cui è collegato instradare il frame.

Tra i protocolli non instradabili, si ricorda NetBEUI, originale delle reti Microsoft.
Tra i protocolli instradabili invece ci sono IP/SPX (Novell) e TCP/IP

TCP/IP

Struttura del pacchetto IP

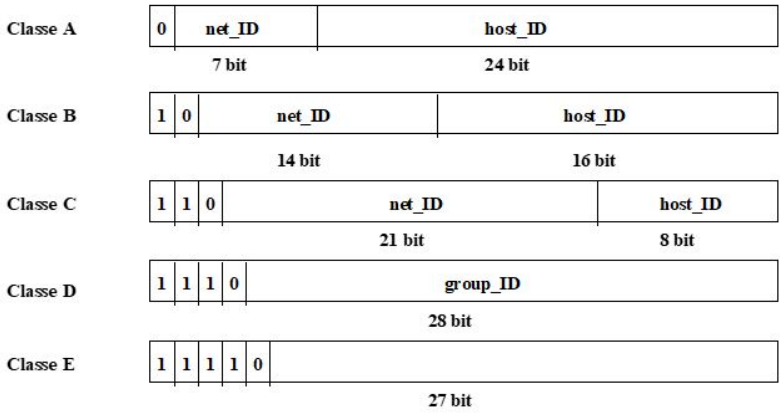


Descrizione campi non ovvi:

- ***TTL*** → Nessun pacchetto può oltrepassare una passerella più più del numero di volte indicato nel TTL (al max 256). Al raggiungimento di 0, il pacchetto viene droppato.
- ***Checksum*** → è una somma binaria di controllo dell'header
- ***Protocollo (SAP)*** (8 bit) → indica quale header di livello 4 il livello 3 sta trasportando (campo di demultiplazione interna). Ve ne sono 4 possibili (6 = TCP, 17 = UDP, 1 = ICMP e 2 = IGMP)
- ***Offset di frammentazione*** → contatore del punto in cui sono di un pacchetto più grande che ho frammentato
- ***Flag*** (3 bit) → A "1" se si tratta di un frammento o meno (2 dei 3 bit di flag che ci sono). L'altro bit specifica se i router possono frammentare o meno il pacchetto.
- ***Id del datagramma*** (16 bit) → Identifica l'identificativo del pacchetto, e la sua dimensione identifica la dimensione massima di un pacchetto IP (64Kb, che è >1.5Kb del frame ethernet)

Off-topic interessante: esistono diversi tipi di attacchi per le implementazioni del protocollo non curate: far corrispondere indirizzo destinatario a quello ricevente, inserire un valore anomalo nel campo "protocollo", indicare una lunghezza non corretta per il pacchetto (kiss of death: lungh. pacchetto < lungh. scritta: il ciclo che andrà a incrementare il puntatore farà troppi cicli, andando in segmentation fault, facendo crashare il processo di rete.
In generale, la malformazione degli header dei vari livelli applicativi può creare notevoli disagi (attacco recente a header RPC).

INDIRIZZAMENTO DI CLASSFUL



Il concetto è che ci sono:

- poche organizzazioni grandi che avranno necessità di avere tantissimi host
- abbastanza organizzazioni medie che avranno di bisogno di un buon numero di host
- tante piccole organizzazioni che avranno bisogno di un numero di host di limitato

Si decise di utilizzare i primissimi bit per identificare la classe, e i bit a seguire per identificare il prefisso. (Es. 0 → Classe A, 7 bit → Prefisso)
> Classe A → 124 reti in tutto il mondo → ciascuna può avere 16kk di host
> Classe B → 16k reti in tutto il mondo → 65k host
> Classe C

Complementari (non alternativi a A, B, C)
> Classe D → Complementare. Destinato al multicast. Prevede di assegnare al router di ingresso un gruppo in aggiunta ai vari indirizzi A, B, C (per limitare, ad esempio, il numero di indirizzi verso il quale spedire un pacchetto... tot host saranno .
> Classe E → Scopi sperimentali

SUBNET MASK

Ad oggi non si ha una separazione così rigida, ma si ha una lunghezza del prefisso dinamica. Per far ciò si usa una convenzione, e cioè una sequenza di bit che ha tutti 1 per i bit che indicano il prefisso e tutti 0 per la seconda parte (maschera di sottorete). Es. se ho l'indirizzo 190.50.30.20 e la subnet mask 255.255.0.0, il prefisso è 190.50 (16 bit, anche se l'indirizzo sembra di classe C). La notazione può essere anche {Ip}/{num bit di maschera}.
La dicitura "maschera di classe" identifica il numero di bit standard per la classe a cui si affida l'IP.
Classful: Si usano le maschere predefinite delle classi:

- Classe A: 255.0.0.0
- Classe B: 255.255.0.0
- Classe C: 255.255.255.0

Classless: l'ip viene presentato con la relativa maschera che permette di estrarne il prefisso

- Esercizio d'esame -
Calcolare in decimale il più piccolo e il più grande prefisso in classe A, classe B e classe C
> Classe A → Min 1 (0 | 0000001), Max 126 (0 | 1111110)
→ Non si considerano i "tutti 0" e i "tutti 1" (broadcast di sottorete) in quanto sono riservati.
…

Lezione 5

ARP

Address Resolution Protocol. Si tratta della tecnica per trovare l'indirizzo MAC a cui spedire informazioni all'interno della stessa sottorete.
In ogni macchina è presente una cache ARP con corrispondenza IP/MAC, e viene riempita tramite operazioni di broadcast di pacchetti che hanno l'ip di cui si desidera conoscere il MAC address come payload.
La cache ARP è modificabile, e questo rende la macchina vulnerabile ad attacchi dove modificando la cache ARP con un MAC proprio, e tramite questo monitorare il traffico creato o mostrare del materiale ingannevole.

- Esercizio d'esame -

Creare il flow chart di ARP

1. Il mittente controlla la sua address cache ARP per vedere se contiene la corrispondenza tra indirizzo fisico e indirizzo IP.

2. Se l'indirizzo IP non è nella address cache ARP, il mittente invia una richiesta a tutte le macchine della rete locale. L'interrogazione è del tipo: "Se stai usando questo indirizzo IP per favore fammi sapere qual è il tuo indirizzo hardware (il mio indirizzo IP è ... e il mio indirizzo hardware è...)".

3. Ogni host sulla rete locale controlla se l'indirizzo IP richiesto corrisponde al suo e risponde solo in caso affermativo. La risposta ARP è inviata direttamente al mittente e contiene l'indirizzo ethernet richiesto. La risposta sarà del tipo:"A: 20-4C-4F-4F-50-20 (200.20.1.50), che significa: "Io sono 200.20.1.50 e il mio indirizzo ethernet è 20-4C-4F-4F-50-20".

4. I due interlocutori aggiornano le proprie cache ARP con le corrispondenze tra indirizzi.

SUBNETTING

Utilizzando una maschera di sottorete non di default come 255.255.255.128 si ripastisce un net_id di classe C (ad es. 196.20.70) in due subnet distinte, una con gli host con host_id > 128 e una con gli host con host_id < 128, ciascuna di 125 host l'una (127 - 2). Le due reti avranno prefisso 196.20.70.0 e 196.20.70.128.
Senza conoscere la maschera di sottorete (in ambito classless) è impossibile stabilire se due IP si trovano o meno sulla stessa sottorete.
Una volta fatto il piano delle VLAN di livello 2, per ogni VLAN si dovrà applicare una sottorete partendo dal prefisso.
Quindi si dovrà stabilire il prefisso di livello 3 partendo da quello base, decidere quali VLAN dovranno comunicare tra di loro e collegare tramite gateway, ossia computer con due o più schede di rete, una per ciascuna delle sottoreti a cui è collegato (modo tradizionale: un router per ogni coppia di vlan che devono comunicare; oppure router-on-a-stick). Infine, bisognerà configurare i gateways, e quindi configurare il default gateway per tutte le sottoreti (che desiderano comunicare con altre VLAN, altrimenti no).
> Num VLAN = Num Sottoreti

L'attività di progetto di un'inter-rete si articola nelle seguenti fasi:

- Determinazione del numero di subnet_id necessari per le future esigenze di crescita.
- Determinazione del numero massimo di indirizzi IP su ogni subnet, considerando la crescita futura.
- Determinazione delle maschere di sottorete
- Calcolo dei subnet_id da utilizzare
- Calcolo degli host_id e assegnazione degli indirizzi IP agli host

Es. fare 4 sottoreti partendo da 192.168.0.0
> La maschera sarà 255.255.255.192 (11111111.11111111.11111111.11000000)

> I quattro prefissi sarebbero (ammettendo tutti zeri e tutti uni, quindi 0 e 192, cosa possibile dal 2002)

- 192.168.0.0
- 192.168.0.64
- 192.168.0.128
- 192.168.0.192

Per ogni host i potranno avere 62 hosts (1 - 62, 65 - 126, 129 - 190, 192 - 254)

La tecnica che utilizza una maschera di rete a lunghezza fissa per tutte le sottoreti prende il nome di ***fixed length subnet mask (FLSM)***. Quando si hanno dei collegamenti punto-punto ciò è sconveniente. In quel caso si ricorre al ***Variable LSM (VLSM)***:
Se ad esempio si vogliono creare 4 sottoreti A, B, C, D c'è un fabbisogno di indirizzi per cui A = 100, B = 100, C = 24 e D = 24 e avessimo un net_id di classe C 192.68.20.x.
Con la tecnica FLSM ciò non sarebbe possibile (ogni sottorete avrebbe 62 host al massimo).
VLSM permette di usare varie maschere che possono avere lunghezze differenti per sottoreti ottenute dallo stesso net_it:

- Rete A e B: 255.255.255.128 (126 host ciscuna) (maschera di 25 bit - 1 bit di subnetting). Prefissi:
  - 192.68.20.0
  - 192.68.20.128
- Rete C e D: 255.255.255.224 (30 host) (maschera di 27 bit - 3 bit di subnetting (2 di subsub)) (che poi non è altro che una subnet a 2 bit della prima subnet). Prefissi:
  - 192.68.20.0
  - 192.68.20.32
  - 192.68.20.64
  - 192.68.20.96

>> Se ci sono dei collegamenti punto-punto (collegamento tra insieme di reti e internet o collegamenti a lunga distanza) si utilizzerà una subnet a parte per loro.

- Domande/**Esercizi** d'esame -

Data una rete 192.168.10.0 e una maschera "/25" (tutti 255 e 128 finale)

- Quanti sottoreti ci possono essere? → $2^{(\text{Num bit a 1 della maschera - lunghezza in bit del prefisso di classe})}$ → $2^{25-24} = 2$
- Quanti host per subnet ci possono essere? → $2^{(32 - \text{Num bit a 1 della maschera})} - 2$ → $2^{32-25} - 2 = 126$
- Quali sono le sottoreti valide? → Le varie combinazioni di bit della subnet mask
  → x.x.x.00000000 e x.x.x.10000000 = 0 e 128
- Quali sono gli indirizzi di broadcast per ciascuna subnet? → Si mettono i bit dell'host tutti a 1
  → x.x.x.127 e x.x.x.255
- Siccome si sta ricorrendo al subnetting, qual è l'indirizzo delle due interfacce del router che collega le due subnet? → Due indirizzi a scelta tra il pool degli indirizzi liberi per gli host
- Quali sono gli host validi? → Intervallo tra indirizzo "0" e indirizzo di broadcast per le reti → 1 - 126 / 129 - 254

Il formato della risposta dovrà essere

Subnet Address	0	32	.....	192	224
First Host	1	33		193	225
Last Host	30	62		222	254
Broadcast Address	31	63		223	255

- Se fosse richiesto di completare a livello 2, esistono due possibilità in questo caso:
  - Si utilizza uno switch dando 126 porte a ciascuna rete e quindi il router che collega le due VLAN
  - Si utilizza uno switch dando 126 porte a ciascuna rete e si collega il router alla porta di trunking (router-on-a-stick), mappando entrambe le interfacce sulla stessa scheda di rete

- Se viene dato il numero di host che si vuole per ogni sottorete:
    - Si decide la potenza di 2 immediatamente maggiore al numero di host (che chiamo A)
    - Si ottengono i bit da riservare alla sottorete facendo (32 - A - num) bit prefisso di classe
- Se viene dato il numero di sottoreti che si vuole ottenere:
    - Si decide la potenza di 2 immediatamente successiva al numero di sottoreti chieste (che chiamo A)
    - A è il numero di bit da riservare alla sottorete

Per quanto riguarda **VLSM**, questo torna utile quando ci sono dei collegamenti punto-punto (ad esempio, due router da collegare tra loro per le far comunicare le reti). Con FLSM si andrebbe a concedere un prefisso con tutti i suoi host ad ad una sottorete che comunque non conterebbe mai più di due host (e cioè i due router collegati tra loro).
Esempio: Si vuole avere 5 sottoreti tali che netA = 14 hosts, netB = 28 hosts, netC = 2 hosts, netD = 7 hosts e netE = 28 hosts. Il prefisso è 204.15.5.0
VLSM prevede che:

- Si definisca la maschera che sarebbe giusta per il numero di host per ogni sottorete (considerano 2 indirizzi in più per tutti zeri e tutti uni):
    - netA → /28
    - netB → /27
    - netC → /30
    - netD → /28
    - netE → /27
- Si ***ordinano*** le maschere in ordine crescente, e si comincia a considerare le sottoreti:
    - netB → x.x.x.0/27 → host da 1 a 30
    - netE → x.x.x.32/27 → host da 33 a 62
    - netA → x.x.x.64/28 → host da 65 a 78
    - netD → x.x.x.80/28 → host da 81 a 94
    - netC → x.x.x.96/30 → host da 97 a 98

NOTA:
Attualmente, FLSM e VLSM vengono utilizzati in CIDR (classless interdomain routing). Questo permette, disponendo ad esempio di un IP di classe B che può avere tanti host, di subnettare e subappaltare dei range di IP a dei clienti, i quali potranno subnettare a loro volta.

CONFIGURAZIONE IP
Per configurare un host bisogna dare:

- un indirizzo IP all'host
- una maschera di sottorete che permette l'estrazione del prefisso
- un gateway predefinito a cui saranno mandati i pacchetti IP i cui destinatari non si trovano sulla stessa subnet del mittente

Anche il gateway predefinito può essere trovato tramite una variante del protocollo ARP.

ATTRIBUZIONE DEGLI INDIRIZZI IP
In linea di principio si pensò di effettuare l'attribuzione degli IP utilizzando un db centrale, in modo che ogni ip concesso fosse univoco.
Questo però era limitativo, in quanto il numero di IP è limitato a circa 4kkk. Per questo si ebbe l'idea di assegnare ad ogni host un IP pubblico univoco solo quando serve, e conservare un IP privato non univoco. Questo è possibile grazie al tunneling (IP è autoincapsulante). In secondo luogo, l'indirizzo pubblico non viene concesso per sempre, ma solo temporaneamente.
Gli indirizzi privati possono essere di tre classi;

- Classe A pvt: 10.x.x.x

- Classe B pvt: 172.16.x.x
- Classe C pvt: 192.168.0.x

Un indirizzo importante è la rete 127.0.0.0 di classe A, usato per gli indirizzi di loopback.

- Domanda d'esame: differenza tra inviarsi il pacchetto e inviarlo a loopback -
I pacchetti inviati a "se stessi" attivano tutta la catena di header fino al frame ethernet e viene fatto rientrare, mentre un pacchetto inviato a 127.0.0.1 non genera gli header, e copia il pacchetto dalla coda di uscita a quella di ingresso, a livello software.

ASSEGNAZIONE DINAMICA IP
Gli indirizzi IP possono essere assegnati da un server nella sottorete su richiesta, invece di essere impostati manualmente e staticamente.
Esistono tre protocolli:

- RARP (Reverse ARP): funziona in modo simile ad ARP, in quanto invia una richiesta dell'indirizzo IP associato a un indirizzo MAC dato. Il server RARP risponde con un indirizzo IP. RARP assegna solo l'indirizzo IP
- BOOTP: consente di ricevere l'info minima di rete che consente il boot di un SO di rete (IP e maschera), concepito sempre per scaricare immagini di memoria per stazioni diskless.
  L'host fa una richiesta BOOTP con il suo mac in broadcast. Questo arriva a tutte le macchine della sottorete, compresa la macchina che può fare da server BOOTP. Il server manda l'indirizzo IP della macchina richiedente, l'ip server che distribuirà l'immagine di boot per il SO e il nome del file. Al che interpella la macchina server per ottenere l'immagine da schiaffare in RAM.
- DHCP (Dynamic Host Communication Protocol): protocollo per l'assegnazione dinamica di Ip (sia pubblici che privati), che concede gli IP solo a macchine che lo richiedono, permettendo di avere più macchine rispetto al numero di Ip disponibili (a patto che non stiano tutte accese contemporaneamente). DHCP, a differenza di BOOTP da Ip non solo al boot del PC ma ogni tot di tempo (cambio al termine della periodo di tempo definito nel server, detto periodo di "lease").

Formato dei messaggi BOOTP/DHCP:

Protocolli test-and-set: tutti i valori sono fissi, anche il payload, e richiesta e risposta hanno lo stesso formato (ad es. indirizzo IP del gateway sarà inviato vuoto e restituito compilato). Si tratta di payload ethernet diretti, non pacchetti IP.
Il campo TTL (conteggio hop) dice il numero specifico di router che questo DHCP può attraversare, e in genere viene impostato a 0 per fare in modo che il pacchetto non attraversi reti VLAN o internet diverse.

Il tipo di messaggio DHCP viene inviato come un'opzione:

- 1: DHCPDISCOVER (inviato in broadcast per scoprire il server DHCP nella sottorete - TTL a 0)
- 2. DHCPOFFER (risposta dai vari server possibili DHCP, viene presa la prima che arriva……..)
- 3: DHCPREQUEST
- 4: DHCPDECLINE
- 5: DHCPACK
- 6: DHCPNAK
- 7: DHCPRELEASE
- 8: DHCPINFORM

Altre info inviate come opzioni DHCP (info che può dare il server DHCP, su richiesta):

- maschera di sottorete nome server, nome host, nome dominio
- TTL IP di default, indirizzo broadcast, instradamento statico, incapsulamento ethernet
- tipi di messaggio DHC, tempo di rinnovo DHCP, time STP server, SMTP server, nome stampante...

Si può richiedere il rinnovamento di un lease inviato quando il 50% del lease è scaduto. È importante settare un tempo di lease giusto (né troppo lungo, che potrebbe tenere occupati degli IP inutilmente, né troppo corto, che potrebbe creare problemi di stabilità della comunicazione).

INSTRADAMENTO IP

Se un router ha due sole interfacce si dovrà limitare a spostare il traffico dall'interfaccia di entrata all'altra interfaccia.
Se però ha più di due interfacce esistono modalità di scelta dell'interfaccia.
Un router statico è un router che decide l'interfaccia nel quale instradare il traffico sulla base di informazioni locali che non cambiano nel tempo.
L'instradamento può essere:

- Diretto: non c'è bisogno di un router, in quanto due macchine della stessa sottorete comunicano tra loro
- Indiretto: instradamento verso un'altra ethernet tramite il routing (se ha una sola ethernet possibile, si parla di "default routing"). Tramite l'effettuazione del test sul prefisso è possibile sapere se il pacchetto è destinato ad un ip su una rete esterna, e in caso si utilizzerà il gateway (creando un frame destinato a questo che ha al suo interno però l'Ip del destinatario).
  Il gateway che riceve un frame il cui ip di destinatario non è il suo si occuperà di estrarlo e iniettandolo su un'altra rete.

Una tecnica "ignorante" è il flooding (generazione di tanti frame con lo stesso pacchetto e inoltro a tutte le reti a cui si è collegati) o "hot potato", cioè copa del pacchetto sull'interfaccia con la coda più libera (con la non-garanzia di arrivo del pacchetto)
Tuttavia in genere si ricorre a tecniche basate su delle tabelle routing (table-driven) che possono essere compilate manualmente (routing statico) o dinamicamente tramite comunicazione tra router con un protocollo di livello 4.

Le tabelle di instradamento hanno *4 colonne*:

- il ***net_id*** (confrontato col prefisso dell'ip del pacchetto in arrivo) per ogni sottorete conosciuta (sarà presente una riga con un net_id di default 0.0.0.0, che viene utilizzata per instradare i pacchetti il cui net_id di destinazione non compare esplicitamente in nessuna riga della tabella stessa)
- La ***maschera di sottorete*** associata l net_id
- L'***IP del gateway di inoltro***, a cui mandare il pacchetto se il suo net_id coincide con la coppia net_id/maschera
- ***Interfaccia di livello 2*** da utilizzare per l'inoltro

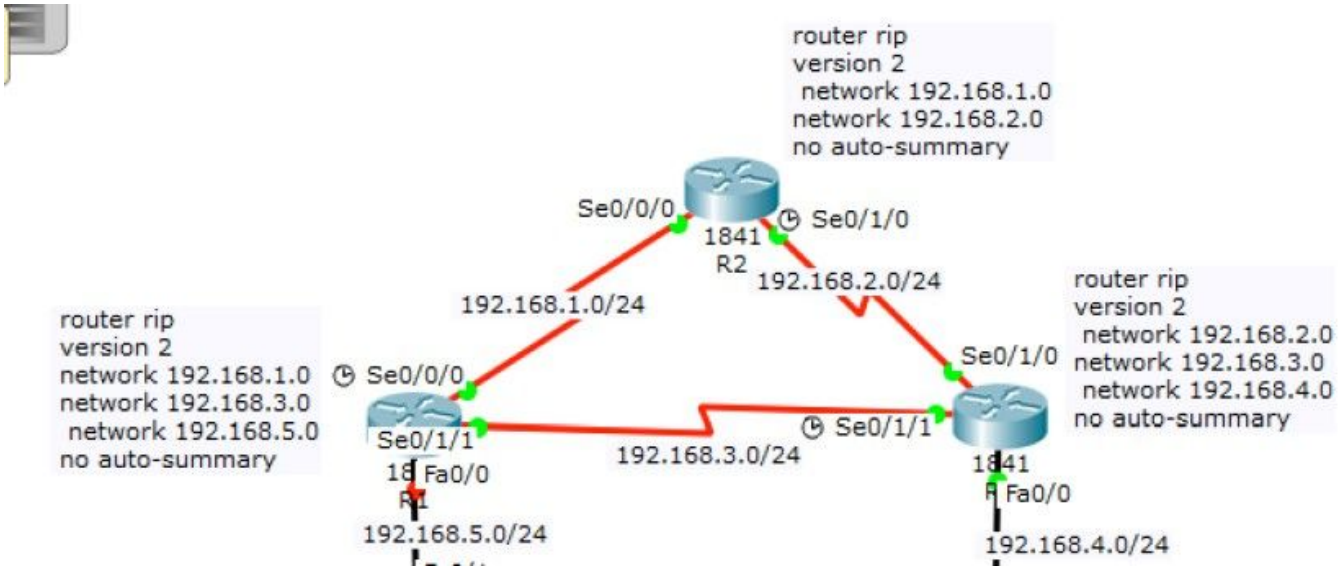
Le tabelle di routing possono essere configurate anche a livello di host, permettendo di scegliere verso quale gateway inoltrare il traffico (ad esempio, se si è collegati ad un gateway che da su internet e uno che da su

una VLAN della rete, se ci serve inviare il traffico alla VLAN sarà meglio passare per il secondo gateway - sempre basandosi sul prefisso -).
Tramite il comando "`route -n`" o "`netstat -rn`" permette di vedere la tabella di routing locale Es.

```
[root@pluto root]# netstat -rn          (oppure route -n)
Kernel IP routing table
Destination     Gateway         Genmask         Flag    MSS Window  irtt Iface
130.22.37.0     0.0.0.0         255.255.255.0   U        40 0          0 eth0
127.0.0.0       0.0.0.0         255.0.0.0       U        40 0          0 lo
0.0.0.0         130.22.37.1     0.0.0.0         UG       40 0          0 eth0
```

U: la rotta è UP
G: la rotta è verso un gateway, altrimenti è diretta
H: la rotta è verso un host, altrimenti è verso una rete

Altro esempio di tabella di routing



netstat eseguito su R2 genera la seguente uscita:
show ip route

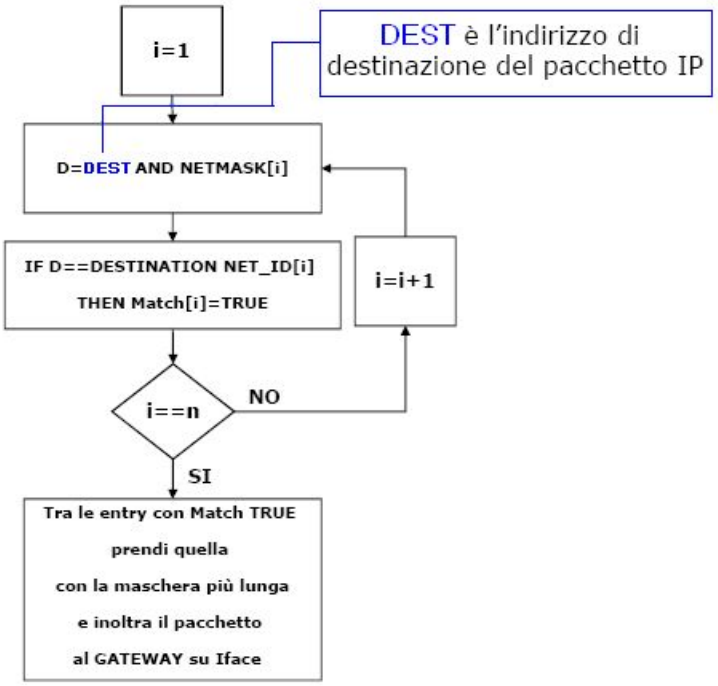
```
Codes:

C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
* - candidate default, U - per-user static route, o - ODR   P - periodic
downloaded static route Gateway of last resort is not set.

C 192.168.1.0/24 is directly connected, Serial0/0/0
C 192.168.2.0/24 is directly connected, Serial0/1/0
R 192.168.3.0/24 [120/1] via 192.168.1.1, 00:00:18, Serial0/0/0
R 192.168.4.0/24 [120/2] via 192.168.1.1, 00:00:18, Serial0/0/0
```

- Esercizio esame -
"Questa macchina ha cache ARP vuota e deve inviare un pacchetto all'indirizzo 130.22.37.0. Cosa succede?

1. La macchina esegue un broadcast ARP per rintracciare l'indirizzo di destinazione e poi gli manda il frame con il pacchetto
2. La macchina inoltra il pacchetto in un frame al default gateway

Risp 1. in quanto i matching sono di lunghezza diversa: con il prefisso 130.22.31 ha matching di lunghezza 3, mentre con 0.0.0.0 ha un matching di lunghezza zero (non è matching nel caso della seconda riga).
Quando viene richiesto il "calcolo dell'instradamento" bisogna quantificare i match di tutte le righe della tabella, per far vedere che ci possono essere più match e che si utilizza quello con quantificativo di match maggiore.

Algoritmo di instradamento "ingenuo" (Ingenuo perché esegue un ciclo for sulle righe della tabella invece che un'operazione di costo costante)



Questo algoritmo permette di contare i match sulla tabella (che dovranno rispettare la maschera, quindi se è presente una route con "192.168.5.0/24" la destinazione 192.168.6.5 NON MATCHA, è sbagliato dire che matcha per i primi due byte).

Per configurare il routing su un router è necessario accedere alle loro interfacce di amministrazione:
- Su alcuni router cisco, dotati di SO, è possibile accedere tramite un'interfaccia terminale seriale
- Altrimenti, bisogna accedere a delle interfacce di rete che permettono la configurazione dei router e che i router non collegano direttamente nelle reti che gestiscono (stanno in delle reti speciali chiame "control plain", diverse dalle "data plain"). Si utilizzerà il protocollo SNMP (Simple Network management Protocol) che è un protocollo sul quale i router rispondono sul "control plain" alle richieste di amministrazione (previa autenticazione)
  Fatto sta che bisogna tenere queste interfacce separate dal data plain, presumibilmente in una VLAN apposita per l'amministrazione.

Comando TRACERT
Utilizza il TTL di IP. Se il router droppa un pacchetto con TTL troppo basso, manda una notifica per segnalare ciò.
Questo permette di ricreare una traccia dei router attraversati per l'arrivo a destinazione (in base al numero di hop). I router che non inviano la notifica vengono stampati a video con un "*". Il cambiamento di un percorso in vari momenti della giornata può avvenire a causa del cambio della situazione delle tabelle di routing (cambio percorsi, percorsi che vanno in "down"...).

ICMP

Questo protocollo è utilizzato per la comunicazione Router - Host, per comunicare al mittente errori o eventi avvenuti nell'inoltro di un pacchetto IP. Si tratta di un protocollo di livello 4.
C'è l'eccezione del pacchetto di sondaggio ("***ping***") che può essere inviato anche ad altri host della rete.
Piuttosto, la notifica di drop a causa del TTL a zero è un messaggio ICMP.
Esistono tre tipi di messaggio:

- Router to router: per scambio di info di servizio
- Router to host: per tenere sotto controllo le modalità con cui gli host generano pacchetti (per farli rallentarli o dirottare altrove il traffico)
- Host to host: scambio di info per verificare funzionamento e topologia di rete

ICMP e IP vanno insieme, e necessitano l'uno e l'altro.

I pacchetti ICMP vengono instradati dai router prima dei pacchetti IP ordinari, e questo terrà una coda diversa per pacchetti IP e pacchetti ICMP (che saranno appunto prioritari).
Al drop di un frammento, viene generato un messaggio ICMP solo se è stato droppato il frammento 0.
I messaggi ICMP non sono mai inviati in risposta a pacchetti IP multicast (0.0.0.0, 127.0.0.1, Ip broadcast).
Inoltre, i msg ICMP non sono mai inviati in risposta a pacchetti ICMP.

Tabella messaggi ICMP

No.	Mittente-destinatario	significato
0 Echo reply	Da host a host	Risposta a una richiesta di eco
3 Destination unreachable	Da router a host	La destinazione è irraggiungibile
4 Source quench	Da router a host	Il mittente deve rallentare l'invio di pacchetti
5 Redirect	Da router a host	il pacchetto viene inviato a un altro gateway d'inoltro
8 Echo	Da host a host	Richiesta d'eco
9 Router advertisement	Da router a router/host	Annuncio di rotte
10 Router solicitation	Da router a router/host	Richiesta di rotte
11 Time exceeded	Da router a host	Drop pacchetto per TTL=0
12 Parameter problem	Da router a router/host	Un parametro non è valido
13 Timestamp request	Da host a router	Richiesta rilevazione tempo di attraversamento
14 Timestamp reply	Da router a host	Rilevazione tempo di attraversamento
17 Address mask request	Da router a router/host	Richiesta maschera di sottorete
18 Address mask reply	Da router a router/host	Risposta maschera di sottorete

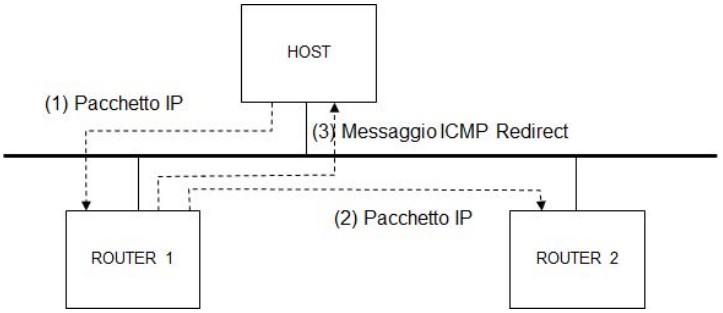
- Domande esame -

Spiegare come il controllo ICMP supporta il ***congestion control****:*
Il protocollo ICMP supporta il congestion control nella comunicazione router to host che prevede il msg di source quench indirizzato alle sorgenti dei pacchetti quando la coda di attesa supera una certa percentuale di occupazione.

Se il router di default è configurato in redirect, spiegare cosa succede quando l'host genera un pacchetto che deve andare in una sottorete interna, mentre il default gateway inoltra i pacchetti in internet
Il default gateway, quando vede arrivare un pacchetto che ha come ip di destinazione il prefisso di una delle sottoreti aziendali interne lo droppa o lo inoltra alla sottorete, e quindi invia all'host un messaggio ICMP che conterrà l'ip dell'altro router da utilizzare. Così facendo, il default gateway farà da direttore del traffico.
Qual è l'alternativa?

L'alternativa è configurare le tabelle di routing di ogni host in modo che queste permettano di generare l'indirizzo del gateway giusto e non solo quello di default.

ICMP Redirect

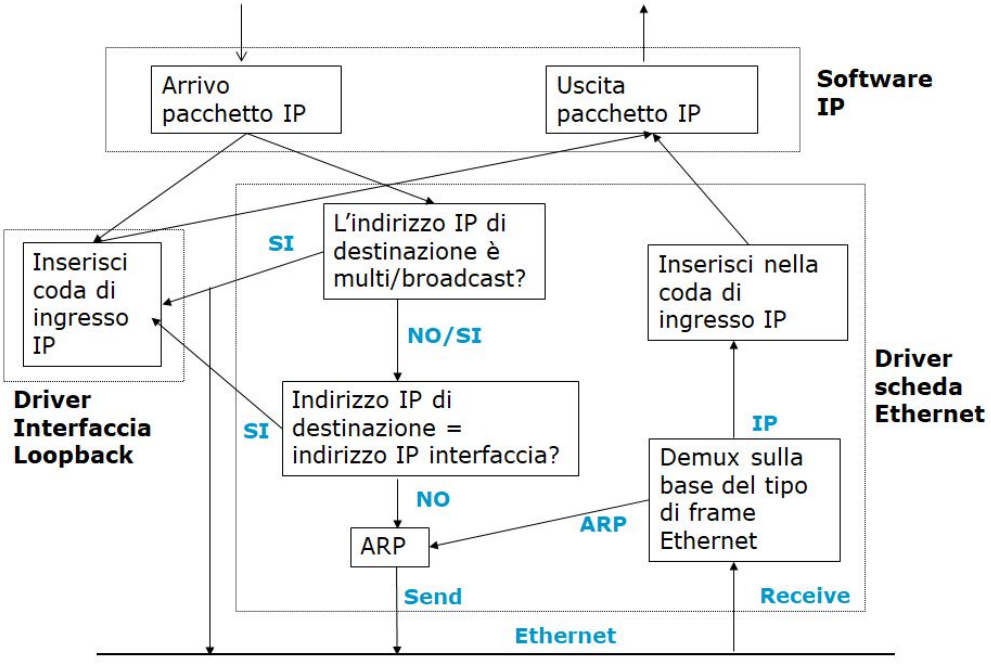


Un router utilizza ICMP per comunicare al mittente di un pacchetto che il pacchetto IP viene inoltrato ad un altro router sulla stessa rete. Alternativamente, dropperà il pacchetto, e sarà compito dell'host ritrasmetterlo. L'host terrà conto di questa indicazione e userà il router "giusto" per i pacchetti successivi diretti alla stessa destinazione.

- A questo punto la **progettazione del livello 3** diventa come segue -

- Piano di indirizzamento con subnetting
- Attribuzione indirizzi agli host e ai router
- Configurazione delle tabelle di instradamento dei router
- Configurare sul router che si collega a internet il redirect

Interfaccia di Loopback



Risoluzione degli indirizzi IP: problemi

Possono sorgere dei problemi dalla risoluzione degli indirizzi IP:

- Uso di maschere di sottorete errate: ciò rende il test del prefisso non affidabile, e ci si può trovare con degli host irraggiungibili. Inoltre ciò non crea problemi solo per l'host a cui si modifica la maschera, ma alla rete in generale, in quanto i pacchetti ARP in broadcast creano traffico e occupano banda.
- Indirizzi IP duplicati sulla stessa sottorete: questo può essere verificato in diverse situazioni, tipo un broadcast ARP da parte della macchina con lo stesso ip, e la reazione dipende dai SO.

INDIRIZZI PUBBLICI E PRIVATI

Una rete privata IP è una rete privata che utilizza gli indirizzi IP che non è è direttamente connessa a internet. In una rete privata gli indirizzi IP possono essere assegnati in modo arbitrario (senza registrarli e senza univocità).
In genere le reti private usano indirizzi che fanno parte della serie di indirizzi riservati che segue (*non instradabili*) (***esempio tabella NAT***)

- 10.0.0.0 / 10.255.255.255
- 172.16.0.0 / 172.31.255.255
- 192.168.0.0 / 192.168.255.255

Questi prefissi vengono affidati ad interfacce che non si interfacciano direttamente sull'internet pubblico (e infatti non sono univoci nel mondo).

NAT (Network Address Translation)
Si tratta di una funzione dei router che collegano reti private a Internet pubblica.
Questa prevede che gli indirizzi IP (e a volte i numeri di porta dei pacchetti IP) vengano sostituiti quando arrivano al perimetro di una rete privata (sostituisce la coppia porta/IP di ogni pacchetto in transito con un'altra coppia porta/IP, e sarà tenuto in un db l'associazione tra IP privati e IP pubblici che vengono ad essi sostituiti).
NAT consente agli host delle reti private di comunicare con gli host su internet, ed è attivo sui router.
L'utilizzo del ***numero di porta*** a livello 4 permette di muliplexare e demultiplexare il traffico su un unico IP di uscita, in questo caso prende il nome di NAPT, o PAT.
Il NAT prevede comunque il ricalcolo dei checksum IP (cambiando il contenuto del pacchetto). Inoltre, la frammentazione è critica (potrebbero essere assegnati IP diversi per i frammenti).

M2 U4 L1/2/3 Aspetti aggiuntivi del Nat

Il NAT permette:

- cambiare in maniera semplice il provider senza eseguire il renumbering (la riconfigurazione di tutti gli host della rete). Sarà sufficiente la riconfigurazione del router di NAT.
- di mascherare l'IP, ammesso che si utilizzi PAT (infatti non ci si presenta col proprio IP, ma con un IP pubblico), e mascherare così anche la struttura della rete.
- bilanciare il carico dei server, avendo un unico IP pubblico il NAT riceverà tutte le richieste e potrà smistarle.

Linux usa il pacchetto Netfilter/iptable per aggiungere delle regole di filtraggio allo stack IP. Questo permette di stabilire tramite le opzioni di linux che permettono il routing, come utilizzare il nat (quale indirizzo pubblico utilizzare in base all'indirizzo privato di provenienza)

Esempi comando iptables:
Primo esempio, mapping ip-ip

```
iptables -t nat -A POSTROUTING -s 10.0.1.2 -j SNAT -to-source 128.143.71.21
```

Pool di indirizzi IP:

```
iptables -t nat -A POSTROUTING -s 10.0.1.0/24 -j SNAT -to-source
128.128.71.0-128.128.71.30
```

Migrazione ISP:

```
iptables -t nat -R POSTROUTING -s 10.0.1.0/24 -j SNAT -to-source
128.195.4.0-128.195.4.254
```

IP Masquerading:

```
iptables -t nat -A POSTROUTING -s 10.0.1.0/24 -o eth1 -j MASQUERADE
```

Bilanciamento del carico:

```
iptables -t nat -A PREROUTING -i eth1 -j DNAT -to-destination 10.0.1.2-10.0.1.4
```

M2 U3 L4 Introduzione all'instradamento IP Multicast

MBONE

Gli indirizzi di classe D corrispondo ad indirizzi di router multicast, che ricevendo i pacchetti ne genera tante copie per quanti sono gli abbonati per quel gruppo (che devono aver fatto un'operazione di JOIN sul router multicast).
Per gestire la funzionalità di multicast, si utilizza la tecnica di tunneling (un pacchetto IP sta dentro un altro pacchetto IP).
Quando il router di destinazione si trova in un'isola multicast, nessun problema, diversamente il pacchetto con indirizzo di classe D sarà incapsulato dentro un altro pacchetto IP che avrà come indirizzo di destinazione quello del router di confine dell'isola.
La serie di router multicast, le loro sottoreti direttamente connesse e i tunnel di interconnessione definiscono al dorsale multicast (MBONE).
Nel particolare caso in cui i membri di un gruppo multicast siano connessi tutti alla stessa rete locale, un indirizzo di classe D si può tradurre in un indirizzo di multicast ethernet MAC (sfruttando le potenzialità dello switch).
L'indirizzo multicast ethernet sarà composto da

- un prefisso 01-00-5E
- Il primo bit del 4° byte a 0
- Gli ultimi 23 bit dell'ip di multicast (componendo così gli ultimi 3 byte del mac address)

In questo modo, lo switch distribuirà il frame in multicast sulle porte dello switch dedicate, ottenendo prestazioni molto elevate

M2 U5 L1/2/3 Codici di controllo degli errori

Il controllo dell'errore, necessario per avere un trasporto affidabile si basa su:

- somme di controllo (checksum)
- correzione degli errori

Queste si applicano sui livelli 3 e 4 di TCP/IP.
Il ***checksum ip*** si calcola:

- Trattando l'intestazione come una sequenza di interi a 16 bit dove il checksum vale 0
- Sommando gli interi (con aritmetica in complemento a 1)
- Prendendo il complemento a 1 (cioè il not binario) del risultato

Alla ricezione del pacchetto, si ricalcola per vedere se è stato cambiato qualcosa
TCP usa un meccanismo analogo, aggiungendo una pseudo-intestazione che contiene anche IP sorgente, IP Destinazione. Prende in considerazione i dati nel calcolo.

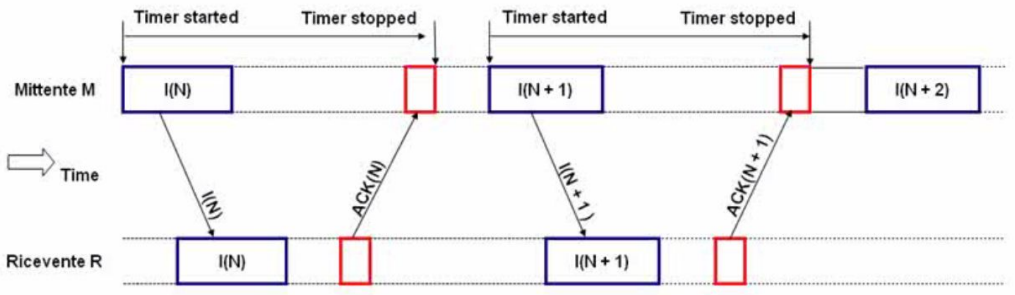
Quando il servizio desidera garantire che tutti i pacchetti siano consegnati (prima o poi) al destinatario senza contenere errori non ci si potrà limitare a scartare i pacchetti non buoni, ma bisognerà applicare delle tecniche specifiche:

- FEC (Forward Error Correction): Vengono utilizzati particolari codici di ridondanza (come il CRC del frame Ethernet) che permettono, non solo di rilevare l'esistenza di un errore, ma anche di correggerlo. Non si può basare su checksum.
- ARQ (Automatic Repeat reQuest): se il pacchetto contiene errori, si richiede al mittente di inviare di nuovo il pacchetto

Tipi di ARQ

IDLE RQ

detto anche "Stop & Wait" o "Send & Wait" → "Manda pacchetto e aspetta conferma prima di mandare il successivo. Se corrotto ritrasmetti."
Sia il destinatario che il mittente possono avere buffer minimi: basta che siano in grado di memorizzare un singolo pacchetto.

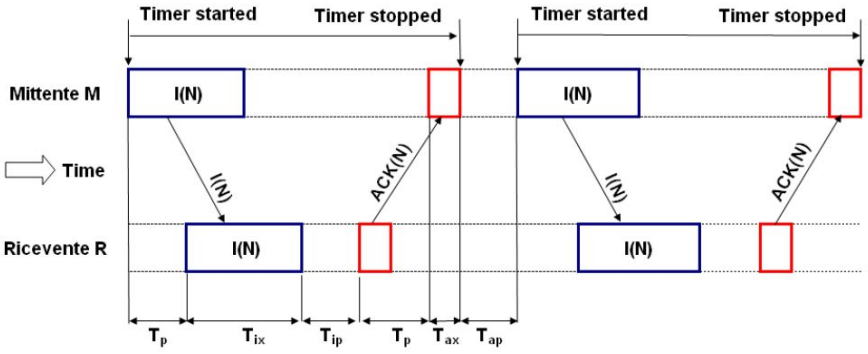


Il mittente spedisce il pacchetto e attende la conferma. Se entro un certo tempo T (timeout) non arriva la conferma (ACK), si prende come una notifica implicita di pacchetto arrivato danneggiato.
Si possono applicare due strategie:

- Non invio di un ACK per segnalare i pacchetti arrivati danneggiati
- Invio di un NAK per segnalare esplicitamente i pacchetti arrivati danneggiati

Se va perso un ACK, il destinatario dovrà essere in grado di scartare le copie dello stesso pacchetto in arrivo.

Temporizzazione Idle RQ



T_p = Ritardo di propagazione del Frame (M → R)
T_{ix} = Tempo di trasmissione del Frame (M → R)
T_{ip} = Tempo di processazione del Frame in M
T_p = Ritardo di propagazione di ACK (M → R)
T_{ax} = Tempo di trasmissione di ACK (M → R)
T_{ap} = Tempo di processazione di ACK in R

Il tempo di timeout non può essere fisso, dovrà essere adattivo, tenendo conto dell'entità di tempi di propagazione, trasmissione e elaborazione

Efficienza di Idle RQ

L'efficienza di utilizzo di Idle RQ è definita dalla percentuale di utilizzo della linea, e si esprime con:
$U = T_{ix} / T_t$
Con **T_{ix} = tempo di invio del pacchetto (il tempo che il mittente impiega ad inviarlo)** e T_t che è il tempo totale che intercorre tra l'invio di un frame e il successivo
Quindi
$T_t = T_p + T_{ix} + T_{ip} + T_p + T_{ax} + T_{ap}$ → Trascurando i termini di secondo ordine **$T_t = T_{ix} + 2\,T_p = RTT$**
→ Da qua si può ottenere che

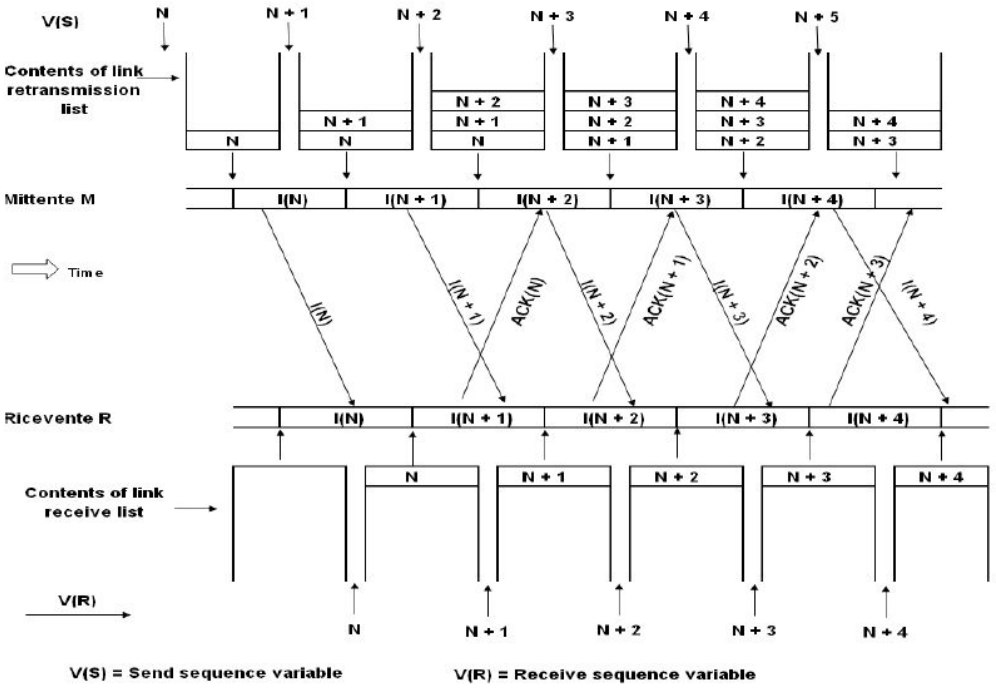
$U = T_{ix} / (T_{ix} + 2\,T_p) = 1 / (1 + 2\,T_p / T_{ix}) = 1 / (1 + 2*alfa)$

T_{ix} = (numero di bit nel frame) / (bit rate in bps)
$T_p = L/v$ è il tempo di propagazione del segnale da Mitt a Ric, ottenuto dividendo la **lunghezza L della linea** per **la velocità v** di propagazione del segnale (**su cavo di rete 2*10^8 m/s**)
Idle RQ non si presta molto bene su linee lunghe e veloci, dato il suo rapporto di efficienza

Continuous RQ

"Manda una sequenza di pacchetti. Se alcuni sono corrotti, ritrasmettili."
L'idea alla base è comunque cercare di ottenere un flusso ininterrotto, dove le notifiche non fermano il traffico.
Il mittente trasmette continuamente pacchetti, che vengono numerati per essere identificati.
Viene conservato ogni frame spedito in un buffer detto lista di ritrasmissione o retransmission list.
Il meccanismo del timer per la ritrasmissione funziona come in Idle RQ.
Il ricevente, quando riceve un pacchetto corretto, invia un ACK che contenga il numero del frame ricevuto.
Viene conservato il pacchetto ricevuto in una lista di attesa: la lista di ricezione o receive list.
Ci sarà una variabile V(R) che conterrà il numero del successivo pacchetto da trasferire al pacchetto superiore.

Nella situazione ideale di ***non necessità di ritrasmissione dei pacchetti***, Continuous RQ si comporta così



Nella situazione ideale, possiamo stabilire il numero di pacchetti che il mittente può spedire prima di ricevere il primo acknowledgement:

$$K = (T_{ix}+2T_p)/ T_{ix} = 1+2RL/Dv = 1+2*alfa$$

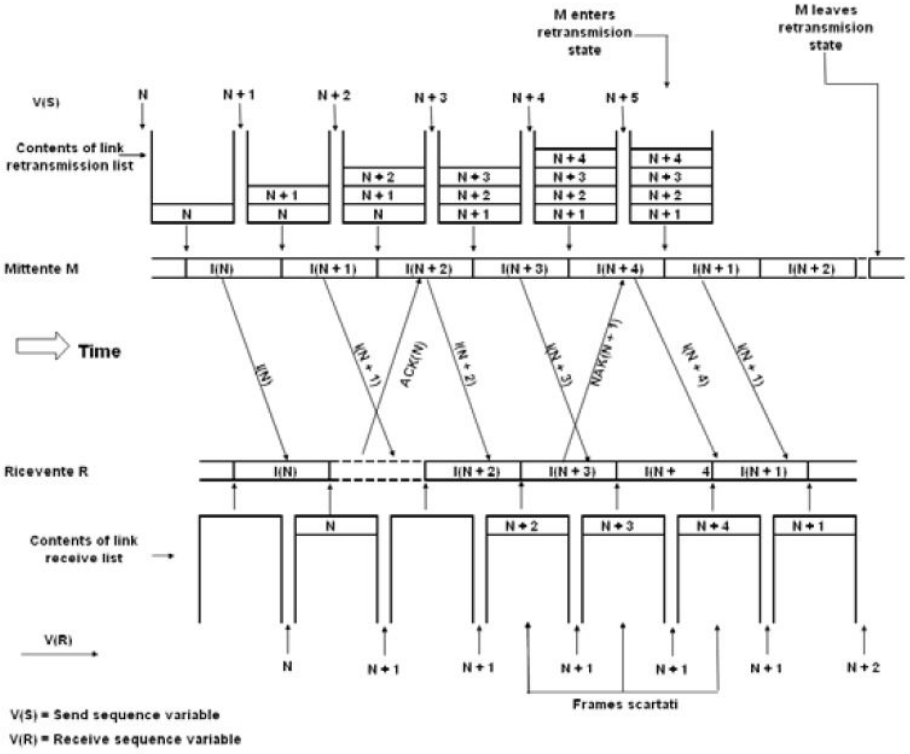
dove:
T_p = L/v è il tempo di propagazione del segnale da Mitt a Ric, ottenuto dividendo la **lunghezza L della linea** per **la velocità v** di propagazione del segnale (**su cavo di rete 2*10^8 m/s**) *(qua si stima T_p "a priori", ma si può misurare anche tramite campionamento)*
T_{ix} = D/R è il tempo di trasmissione, cioè il tempo che M impiega a trasmettere il pacchetto, ottenuto dividendo la **dimensione D del pacchetto** per il **bit-rate R della linea**
→ La retransmission list deve contenere almeno **K** frame → M deve avere un buffer ben più grande rispetto a Idle RQ.

Se si stabilisce un K assoluto, l'efficienza U si misurerà come il U dell'idle RQ, moltiplicando K per T_{ix}, e quindi: **$3 / (1 + 2 T_p / T_{ix})$** (e quindi 3U)

Arrivo di un frame corrotto in Continuous RQ

Quando R(icevente) riceve il pacchetto corrotto lo scarta, e o invia un ack negativo, o attende la fine del timeout.
Quando il mittente si rende conto della situazione, sono applicabili due strategie di ritrasmissione:

- ***Selective Repeat***: R attende che M rimandi il pacchetto N corrotto, mentre nel frattempo accumula i frames che arrivano nella receive list (i pacchetti devono essere trasmessi al livello superiore in ordine, quindi bisogna attendere il pacchetto N)
  → Man mano che si ricevono gli ACK, M rimuove i pacchetti dalla lista di ritrasmissione
  → La lista di ritrasmissione è una lista FIFO
  → La ritrasmissione avviene se il timer associato ad un pacchetto scade oppure si riceve un ACK relativo ad un pacchetto che non è il primo della coda della lista di ritrasmissione (in genere sono equivalenti, in quanto il timeout impostato in genere è $2T_p$ (tempo di andata e ritorno)
  Nel caso di richiesta esplicita di ritrasmissione, dopo l'invio del NAK, R smette di inviare ACK per non rischiare la perdita del pacchetto
  Questo metodo ha una ottimalità nella banda utilizzata (ritrasmette solo i pacchetti corrotti o non ricevuti), però richiede una certa complessità di gestione.
- ***Go-Back-N***: R chiede ad M di rimandare il frame N e tutti i pacchetti successivi tramite un pacchetto speciale di NAK, chiamato *reject*, scartando tutti i pacchetti corretti che riceve dopo



Se si passa in retransmission state, vengono ritrasmessi tutti i pacchetti presenti nella lista di ritrasmissione.

Nota bene: se si subisce perdita di ACK (ad es. si perdono ACK(N) e ACK(N+1) ma si riceve un ACK prima che M passi in retransmission state (es. ACK(N+2)), M interpreterà quell'ultimo ACK come ACK su tutti i pacchetti fino a quel momento, e quindi scarterà tutti i pacchetti precedenti dalla lista di ritrasmissione (quindi N, N+1 e N+2).

Nello schema Go-Back-N, al ricevente è richiesta una lista di ricezione di un solo frame.
Il mittente deve avere invece tutti i pacchetti in attesa di ACK nella sua lista di ritrasmissione. Quindi per valutare quanta memoria serve per la lista di ritrasmissione serve sapere quanti pacchetti si possono ritrasmettere prima dell'arrivo dell'ack. Questo è il numero $(T_{ix} + 2T_p) / T_{ix}$

Piggybaking
Si è parlato degli schemi ARQ come se la comunicazione fosse half-duplex.
Tuttavia le connessioni sono full-duplex. e ciascuno dei due host si comporta sia da mittente che da destinatario (quindi in ciascuna delle due direzione quindi fluiscono ACK e NAK).

Una tecnica per ridurre il flusso di pacchetti consiste nell'uso del piggyback, che prevede che *tutti i pacchetti contengano il codice I(N) che identifica il loro numero quali pacchetti in trasmissione, e un codice ACK o NAK con un numero N(R) che indicherà il numero del pacchetto a cui si riferisce ACK o NAK.*

M2 U5 L4 TCP

TCP è una comunicazione punto-punto (un M, un R), ed è un flusso *affidabile* suddiviso in segmenti che dovranno essere confermati singolarmente.
Questo flusso viene inviato tramite una pipeline controllata (un continuous RQ, in cui la quantità di segmenti trasmissibile prima di ricevere il primo ACK (detta "finestra") adattata per contenere sempre il numero giusto di segmenti. Inoltre utilizza un buffer di invio e uno di ricezione.
Inoltre è:

- Full duplex: bidirezionale a singola connessione
- Orientato alla connessione: handshake tra M e R prima dell'invio dei dati
- Flusso controllato per adattarlo alle diverse velocità relative di M e R.

Struttura segmento TCP

<table>
<tr><td colspan="3">Porta Sorgente(16)</td><td>Porta destinazione(16)</td></tr>
<tr><td colspan="4">Numero di Sequenza(32)</td></tr>
<tr><td colspan="4">Numero di Acknowledgement(32)</td></tr>
<tr><td>HLEN(4)</td><td>Riservati(6)</td><td>Flag(6)</td><td>Window(16)</td></tr>
<tr><td colspan="3">Checksum(16)</td><td>Urgent Pointer(16)</td></tr>
<tr><td colspan="3">Opzioni</td><td>Padding</td></tr>
<tr><td colspan="4">Dati</td></tr>
</table>

I campi:

- Porta sorgente e destinazione: porta per multiplexing e demultiplexing applicativo
- Numero di sequenza (posizione ***in byte*** all'interno dal flusso di invio, che partirà da un numero pseudocasuale generato in fase di apertura della connessione)
- Numero di Ack: la posizione ***in byte*** del flusso in ricezione in cui il mittente si aspetta la risposta dal destinatario
- Window: dimensione in byte della finestra, che dice il numero totale di byte che possono comporre i segmenti da inviare prima di ricevere l'ack sul primo (variando sul T_p campionato).
  La dimensione della finestra deve essere sempre <= di ½ dello SpazioNumeriSequenza (condizione semplicemente soddisfatta, in quanto lo SpazioNumeriSequenza è 2^{32}
- Lunghezza di intestazione (4 bit): identifica la lunghezza dell'intestazione in termini di ***parole da 32 bit***. Serve perché l'header ha lunghezza variabile (a causa delle opzioni). La lunghezza massima per ***l'header sarà quindi al max 64 byte*** (il campo opzioni sarà 44 (o meglio, 43) byte per le opzioni)
- Flags: per distinguere i segmenti che portano i dati dai segmenti di handshake (***l'intestazione di UDP è invece lunga solo 8 byte***)
- Checksum
- Opzioni: utili per portare info di servizio ed implementare schemi di ritrasmissione sofisticati
- Dati applicativi

Numeri di sequenza e ACK TCP

Esempio:

1. A → [Seq = 42, ACK = 79, data = 'C'] → B
2. B → [Seq = 79, ACK = 43, data = 'C'] → A
3. A → [Seq = 42, ACK = 80] → B

Da ciò si evince che
- Seq(2) = Ack(1)
- Ack(2) = Seq(1) + 1
- Seq(3) = Ack(2)
- Ack(3) = Seq(2) + 1

La gestione dei segmenti fuori ordine (cosa che può succedere) la gestione varierà a seconda dell'implementazione.

Osservazioni su numeri di sequenza e finestra
La dimensione della finestra è massimo 64 KB, che non è sempre sufficiente per riempire la pipeline. Infatti se si ha una rete molto veloce (100 Mbps) con un round-trip delay lento (es. 100ms) si potrebbe scrivere fino a 1.19 MB **(((100kk / 0.1) / 8 / 1024) / 1024)**
ma TCP non lo consente (la lunghezza della pipeline sarà **delay*RTT**).

Il numero di sequenza è un campo da 32 bit (4 GB). Considerando il tempo massimo di vita di un segmento di 120sec (**RFC 1122**) possibile incorrere (se il livello 2 è molto veloce) nella scrittura di una sequenza più elevata di quanto la sequenza possa sopportarne (la velocità massima supportata sarebbe 4GB / 120 sec → 273Mbps (34.125 MBps), che è < della Gigabit Ethernet e di STS-12 a 622 Mbps)

M2 U5 L5 FLUSSO TCP

Dati interattivi: pacchetti piccoli
Se si pensa ad un contesto dove vengono inviati pacchetti molto piccoli (es. server di echo: dati = 1byte) si ha un header di dimensione > dei dati.
In questo caso non ci sarà bisogno di tenere due connessioni per garantire un full-duplex, ma sarà sufficiente usare il piggybacking.
Una tecnica per migliorare la situazione consiste negli ACK ritardati. Questa tecnica prevede che si attenda un po' prima dell'invio di un ACK, così se ci sono altri pacchetti in ricezione, sarà possibile confermarli tutti con un unico ACK.
Questa tecnica ha aperto la strada per ***l'algoritmo di Nagle*** per i dati interattivi:
questo prevede che, se una connessione TCP ha dei dati in sospeso per cui non è stato ricevuto ancora un ACK, non vengono inviati segmenti piccoli, ma aspetta un ACK e invia tutti i dati accumulati fino a quel momento (ovviamente dopo un timeout i dati vengono inviati comunque). Quindi
→ se gli ACK fluiscono rapidamente i dati verranno inviati rapidamente
→ se gli ACK ritornano lentamente (rete geografica), la tecnica incoraggerà la creazione di segmenti più corposi, accumulando i dati prima di trasmetterli
Es. algoritmo
1. A → [Seq = 42, ACK = 79, data = 'C'] → B
   (Nel frattempo, su A si digitano A, e poi T)
2. B → [Seq = 79, ACK = 43, data = 'C'] → A
3. A → [Seq = 42, ACK = 80, data = 'AT'] → B
4. B → [Seq = 80, ACK = 45, data = 'C'] → A

Il tempo di attesa di Nagle aumenterà quindi le dimensioni dei pacchetti

Trasferimento di grandi quantità di dati (bulk transfer)
Il problema che deriva dalla generazione di grandi quantità di dati da inviare (soprattutto se presenti contemporaneamente) è quello di una generazione eccessiva di throughput, in modo da rendere difficoltosa la memorizzazione da parte del ricevente (soprattutto se necessita di interventi dell'utente).
La soluzione è l'uso delle *advertised window (AW)* per rallentare il mittente, e più nello specifico quando si sta riempiendo lo spazio di memorizzazione, restituire progressivamente una dimensione dell'AW sempre minore fino ad annullarlo, così bloccando l'invio da parte del mittente.

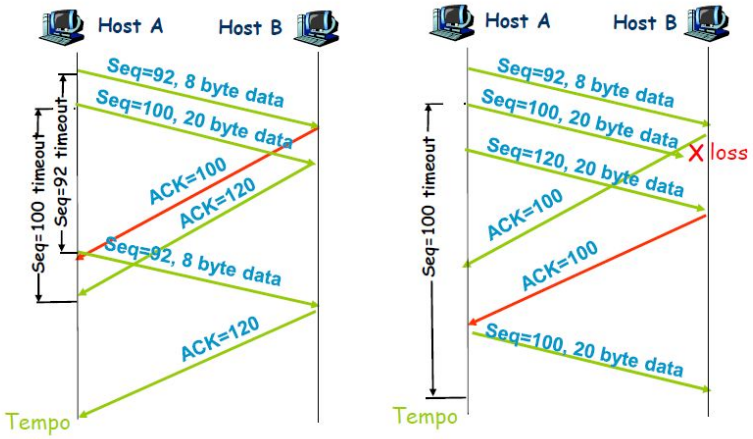
Questo però potrebbe portare ad uno stallo, infatti i pacchetti di ACK sono comunque pacchetti, e se viene mandato un valore “0” per la finestra ciò semanticamente bloccherebbe l’invio di pacchetti da parte del mittente, non fornendo input per far sì che il destinatario generi un nuovo ACK con una nuova finestra a seguito dello svuotamento dei buffer.
La soluzione di TCP è l’utilizzo di ***pacchetti di sondaggio*** (pacchetti di 1 byte) che vengono inviati ripetutamente anche quando la finestra è a 0, permettendo così al ricevente di rispondere un ACK con finestra > 0. Questi pacchetti non vengono memorizzati, e il byte dei pacchetti di sondaggio non vengono considerati parte del flusso TCP.
Il timer che regola l’invio di questi pacchetti si chiama “***Persist Timer TCP***”, impostato dal mittente quando la finestra è nulla.

Questa tecnica introduce però un’ulteriore problema nascosto: la ***sindrome della “silly window”***, e cioè il fatto che il ricevente, a seguito dei pacchetti di sondaggio, apre una finestra sì, ma così piccola che è inutile per la quantità di dati che il mittente deve trasmettere.
Le possibili soluzioni sono:

- Il destinatario non annuncia finestre piccole (continua ad annunciare 0), ma aspetta fino a quando se ne aprono di più grandi (di dimensione di metà del suo bugger o la dimensione di un segmento medio della comunicazione avvenuta fino a quel momento)
- Il mittente evita di inviare dati finché non può inviare almeno un segmento di dimensioni medie, almeno metà della finestra più grande mai annunciata dal destinatario , oppure tutto il buffer.

M2 U5 L6 Calcolo Timeout

Per utilizzare completamente un collegamento la condizione basilare è quella che sia in grado di inviare un sufficiente numero di pacchetti in modo da riempire la pipeline, così da non dover attendere il tempo di round trip dei segmenti.
Questo si può esprimere dicendo che: **finestra >= larghezza di banda * round trip time**
Il calcolo del timer alla quale scadenza un pacchetto viene definito “da trasmettere” dipende in base alle proprietà statistiche del round trip timer.



Nello scenario in cui il timeout è prematuro - a sinistra - (scada prima dell’arrivo di un ack), il segmento viene ritrasmesso e si utilizzeranno degli ACK cumulativi che includeranno anche i segmenti che invece hanno rispettato il relativo timeout e che sono stati recapitati correttamente.

Nel secondo scenario, viene trasmesso due volte un ACK sul primo segmento, a seguito della ricezione corretta del terzo pacchetto, *per segnalare al mittente il fatto che manca un segmento all’appello*, senza aspettare che scada il suo timeout (generalmente, nella pratica, saranno due o tre gli ACK duplicati inviati prima che il mittente si accorga del problema).

Il valore del timeout del TCP si basa su RTT (Round Trip Time), ma deve essere maggiore di esso per evitare un timeout prematuro, considerando la variabilità di RTT.

Inoltre non può essere troppo lungo, in quanto ci sarebbe una reazione lenta alla perdita dei segmenti.

Valutazione del RTT
Per il modo campionato (SampleRTT) si effettua facendo partire un timer al momento in cui viene inviato un segmento e fermandolo quando viene ricevuto l'ACK (si effettua *currentTime* - *sentTime*).
Il fatto che SampleRTT può avere degli sbalzi notevoli impedisce di porre il timeout uguale ad esso.
Un'altra idea consiste nel calcolare una stima con media pesata esponenziale del RTT (EstimatedRTT) tale che:
EstimatedRTT = (1-x)*EstimatedRTT + x*SampleRTT
Il valore tipico di x è 0.1, dando il 90% di peso alla media accumulata e il 10% alla nuova misura, così da non influenzare troppo i valori a seguito della nuova valutazione.

Impostazione del timeout
Quando si ha una forte variazione al valore di EstimatedRTT serve un margine di sicurezza maggiore
→ **Timeout = EstimatedRTT * DelayVarianceFactor (generalmente raccomandato a 2)**

Timeout Jacobson/Karels
Il calcolo qua si basa su media e varianza.
Per semplificare i calcoli si utilizza la deviazione media è una buona approssimazione della deviazione standard:

```
EstimatedRTT = (1-x)*EstimatedRTT + x*SampleRTT
Error = |SampleRTT-EstimatedRTT|
Deviation = Deviation + h*(Error - Deviation)
Timeout =  EstimatedRTT + 4*Deviation
```

(Qua si cerca di tenere più vicino il timeout all'EstimateRTT).

In questo modello, i valori consigliati sono: x = 0.125 e h = 0.25

M2 U5 L7 Andamento di RTT

In uno scenario di RTT costante esaminato sperimentalmente:

- L'original timeout descrive diverse limitazioni, in quanto ha fai dove è sotto al RTT (quindi porta ad una scadenza prematura), e poi si stabilizza su un valore che è comunque il doppio del RTT.
- Diversamente, Jacobson/Karles sta sempre sopra al timeout, e cerca di approssimarsi proprio sulla costante RTT

Se RTT ha una brusca diminuzione invece:

- L'original timeout tenderà fin da subito ad avvicinarsi alla curva di RTT
- J/K invece avrà una prima tendenza ad aumentare (a causa della maggior deviazione media), e poi però tenderà diminuire verso la costante RTT

Se la variazione di RTT è periodica invece:

- L'original timeout tenderà a stare molto vicino alla curva RTT, seppur con dei primi timeout prematuri
- J/K invece terrà una curva molto più alta, scongiurando sempre i timeout, ma essendo meno efficiente dell'original

Se la variazione è verso il basso ed è periodica, comunque sia l'OT che J/K stanno abbastanza lontani.

M2 U5 L8 Controllo congestione

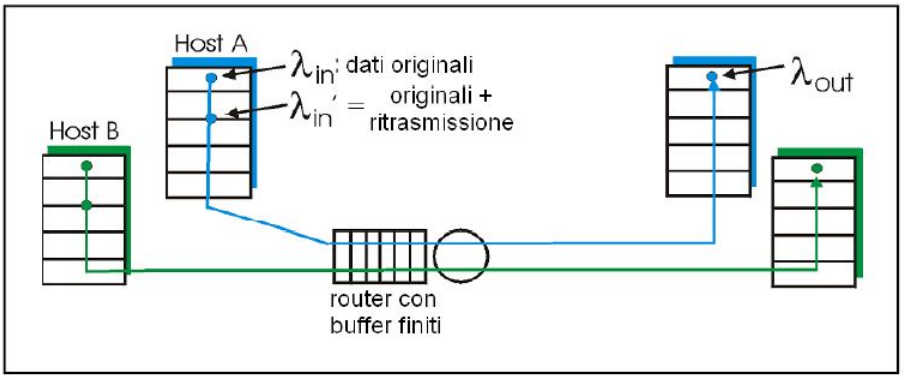
Controllo della congestione: Evitare che più mittenti inseriscano troppi dati nella rete complessivamente (prevenendo che collegamenti o dispositivi di commutazione diventino sovraccarichi).
A differenza del controllo di congestione, il ***controllo di flusso*** evita al mittente di superare la capacità del destinatario di elaborare i dati in ingresso.

In una rete orientata alla connessione il controllo consiste nell'evitare di attribuire più canali di quelli che la rete riesce a sopportare, e quindi che le risorse vengano riservate anticipatamente. (ex-ante)
In una rete non orientata alla connessione non si può attuare alcuna prevenzione: si rileva la congestione quando avviene e si reagisce appropriatamente (controllo della congestione). (ex-post)

I router possono alterare l'intestazione dei pacchetti o inviare dei messaggi per avvisare gli altri di una congestione.
I mittenti possono accorgersi della congestione in base a delle metriche:

- ***Pacchetti persi***: se lungo un percorso ci sono più pacchetti che risorse (p.es. spazio nei buffer), non c'è alcuna alternativa se non scartarne alcuni.
- ***Pacchetti ritardati***: le code dei router sono piene e i pacchetti aspettano più a lungo il servizio.

Un modello di congestione (con ritrasmissione) può essere il seguente:



Lambda in (L(in)): tasso di produzione dei dati originali
Lambda in' (L(in)'): tasso di produzione dei dati originali + ritrasmissione dei vecchi
Lambda out (L(out): tasso di smaltimento dei dati del destinatario

Se L(in) supera L(out) il candidato a ricevere i dati è il buffer del router.
Se il buffer del router si riempie, si comincia a perdere dei pacchetti, così che il mittente debba ritrasmetterli (e quindi parte della capacità di produzione dei dati sarà dedicata alla ritrasmissione, uno sforzo aggiuntivo senza risultati).
Inoltre, se più di un host passa per lo stesso router è possibile che si ottenga invece una congestione sul buffer del router, tale che questo, una volta pieno, scarti i pacchetti in arrivo per un host di destinazione che invece poteva ancora comodamente smaltirli (***la capacità di upstream sprecata***)

Le conseguenze sono:

- Collasso della rete
- Man mano che i pacchetti entrano nella rete, il numero dei pacchetti che arriva a destinazione cresce anche, ma non proporzionalmente
- Un pacchetto viene scartato in rete => tutte le risorse che ha usato lungo il percorso vengono scartate.

M2 U5 L9 Flusso e congestione TCP

Il mittente cerca di non sovraccaricare i buffer del destinatario trasmettendo troppo e troppo in fretta (quando il processo mittente è più veloce del sistema di smaltimento destinatario).
Per far questo, è necessario dimensionare la finestra (vedi riferimenti precedenti a TCP).
RcvWindow = quantità di spazio libero rimasto nel buffer, che dovrebbe essere la dimensione massima della finestra!
RcvBuffer = dimensione complessiva del buffer.

In questo scenario:

- Il destinatario deve informare il mittente della dimensione attuale del buffer, così che il mittente abbia un limite superiore sul dimensionamento della finestra → Campo *RcvWindow* nel segmento TCP
- Il mittente dovrà mantenere i dati trasmessi dal destinatario che identificano il RcvWindow più recente.

La dimensione della finestra TCP è decisa quindi sulla base di due fatto:
- L'advertised window del mittente usata nel controllo di flusso
- Da una finestra di congestione

La dimensione della finestra effettiva sarà quindi
→ **ActualWindow <= min(Receiver Advertised Window, Congestion Window)**

Questi valori non sono disponibili all'inizio della connessione. Quindi si dovrà partire da un valore iniziale di finestra e raggiungere il valore di advertised window del ricevente. Ci sono due modi:
- Slow Start (incremento moltiplicativo) Per ogni ACK ricevuto, la finestra di congestione raddoppia fino ad arrivare alla soglia.
- Congestion Avoidance (Incremento additivo): Per ogni calcolo di RTT, la finestra di congestione aumenta di 1.

In realtà si usa un approccio ibrido, stabilendo una soglia (***Threshold***) che fa da spartiacque tra incremento additivo e incremento moltiplicativo (rispettivamente sopra il threshold e sotto). Il threshold cambierà nel corso della vita della connessione
Idealmente l'incremento additivo dovrebbe far raggiungere il valore più grande possibile della finestra di congestione, ma in realtà la finestra arriva continuamente ad un valore troppo grande e bisogna diminuirlo.

Nel caso in cui la variazione della finestra di congestione o dell'AW fanno sì che il minimo diventi inferiore alla finestra attualmente utilizzata (***Congwin***), e quindi si oltrepassa il ***bordo della congestione*** e cominciano le perdite, si diminuisce bruscamente il valore della finestra attuale, per poi incrementarlo di nuovo gradualmente

M2 U5 L10 Controllo timeout

Il modo più ovvio per individuare delle perdite è usare il timeout del timer di ritrasmissione.
→ Con valori elevati di Congwin e RTT alti si avranno grossi svantaggi.

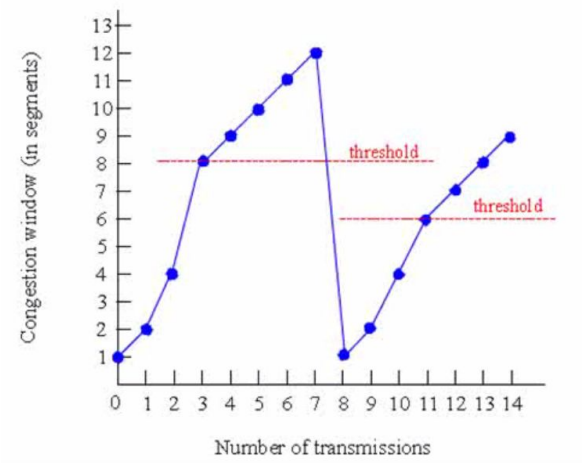
Prendiamo una finestra di 5 segmenti MSS: il mittente trasmette i segmenti 1-5 ma il secondo va perso.
→ nel migliore dei casi, il timer di ritrasmissione scade a **t > ~ 2*RTT**. Poi il pacchetto ritrasmesso deve attraversare la rete e l'ACK corrispondente deve ritornare indietro (altro RTT).

Conseguenze:
- vanno perse quindi più di due finestre piene di dati (2*RTT) (ricordiamoci che si usa Go-Back-N)
- Inoltre si va a riportare il valore di Congwin al valore minore possibile (valore uno), e in più viene diminuito il Threshold (½ congwin).

Quindi si torna ad una situazione di lentezza per un po' di tempo

Congestion Avoidance

```
/* slowstart is over
*/
/* Congwin > threshold
*/
Until (loss event) {
  every w segments
ACKed:
      Congwin++
  }
threshold = Congwin/2
Congwin = 1
perform slowstart
```



Il picco verso il basso si ottiene quando si ha una perdita.
Questo fa vedere come questa tecnica tende a lavorare in regime ottimale (a raso bordo di congestione) solo all'inizio, dopo di che si continua a lavorare in modo subottimo.

M2 U5 L11 Fast Retransmit

Sfruttando l'arrivo di diversi (quattro) ACK per lo stesso pacchetto come indizio di perdita dei pacchetti si può evitare di dover attendere il timeout, e quindi ritrasmettere immediatamente il pacchetto.
Questo comunque non è molto affidabile come metodo, ci possono essere altre casistiche dove di possono avere delle raffiche di ACK.

A tal proposito si è pensato allo scenario di ***Fast Retransmit***:
Si utilizzano ACK duplicati come indizio che sono andati persi dei pacchetti. Per questo motivo si potrà iniziare immediatamente la ritrasmissione (così da evitare di attendere tutto il tempo del timeout, che porterebbe a dover ritrasmettere tutti i dati che nel frattempo sono stati già trasmessi.

Dopo un fast retransmit si applica un ***Fast Recovery***:

- Si imposta il Threshold = ½ (Congwin)
- NON impostiamo la finestra di congestione a 1 ma
  → Congwin = Nuovo Threshold + 3 * (Dimensione Media Segmento (o "MSS"))
- La finestra non viene così diminuita di molto

Quindi questa tecnica tiene artificalmente più grande la finestra di congestione, tenendo così conto dei pacchetti persi.
Quando arriverà l'ACK per un nuovo pacchetto, si "sgonfierà" la finestra di congestione riportandola al valore di Threshold (che comunque è meglio di 1)

M2 U6 L1 UDP

Protocollo di trasporto del gruppo TCP/IP, che fornisce un semplice servizio datagram non affidabile.
I pacchetti possono andare persi o essere consegnati non in ordine.
Gli utenti scambiano datagram (e non flussi come in TCP), e sono possibili trasferimenti in parallelo in entrambe le direzioni (full duplex).

UDP fornisce un servizio di recapito NON affidabile:

- senza connessione;
- senza buffering: accetta i dati e li trasmette immediatamente (nessun servizio di buffering prima della trasmissione).

Per certe applicazioni la mancanza di affidabilità non costituisce un problema (esempio: trasmissione di flussi audio).

Formato dei datagram

- Porta di destinazione: identifica il processo di destinazione.
- Porta sorgente (opzionale): identifica il processo sorgente per le risposte, oppure vale zero.
- Lunghezza del messaggio: lunghezza del datagram in byte, compresi intestazione e dati.
- Checksum (opzionale): checksum di 16 bit su intestazione e dati, oppure zero.

M2 U6 L2 UDP vs TCP

Prestazioni a confronto:

- lo schema di controllo del flusso di TCP basato sulle finestre porta a bulk transfer a ondate;
- l'algoritmo "slow start" può ridurre il throughput;
- TCP ha un elevato overhead per segmento (header grande);
- d'altro canto, UDP può inviare datagram piccoli e non efficaci.

Affidabilità

- TCP fornisce trasferimenti affidabili e ordinati.
- UDP fornisce un servizio inaffidabile l'applicazione, deve accettare o considerare perdite di pacchetti e datagram non in ordine

Multicast e broadcast

- Supportati solo da UDP.
- Lo schema di controllo degli errori di TCP non si presta al multicast affidabile.

Dimensione dei dati

- Datagram UDP limitati a MTU IP (64 KB).

Complessità delle applicazioni

- Il framing a livello applicativo può essere reso difficile da TCP a causa dell'algoritmo di Nagle.
- I dati, però, possono essere ricevuti e letti da applicazioni in aggregazioni diverse rispetto a come sono stati inviati (ad esempio le informazioni delle coordintate del puntatore del mouse. La loro trasmissione raggruppata non è efficace)

UDP e "middleware"

UDP si usa soprattutto quando parte dei compiti di gestione delle connessioni sono delegati al livello applicativo, attraverso uno strato di Middleware (in cui a livello applicativo ci si fa carico di una serie di funzioni gestite a livello di trasporto, come inizio della connessione, autenticazione ecc., così da poter richiedere a livello di trasporto il minimo overhead tramite UDP).

Velocità download e TCP

There's certain amount of your bandwidth used by your modem and ISP in just in keeping the line open! This background use known as "physical signaling overhead" isn't available to you during your transfer.
For example, the advertised transfer speed of a T1 line is 1,536 Kbps however 192 Kbps is usually tied up in the T1 line signaling protocol so the best you'll ever get is 1,344 Kbps.
As a computer myself, let me express my condolences at your loss of bandwidth.

But wait, there's more background use! There's Layer 4 transport and transmission protocol maintainence/overhead. Amazingly, these below processes can eat up to 10% of your transfer speed.

- Handshaking negotiation procedures between you and the place you are transferring the file such as "slow start" described technically in RFC 2001.
- TCP overhead, error checking and sending of protocol header(e.g. Each IPv4 header is 160 bits, each UDP header is 64 bits, etc.)

M2 U7 L1 - Richiami di instradamento statico

Tre configurazioni comuni di instradamento:

- instradamento con default gateway
- instradamento statico con tabella
- instradamento dinamico

Esempio di aggiunta di entry nella tabella di route

```
- #route add 207.25.98.0 172.16.12.1 1
- #route add 192.0.2.32/27 somegateway
- route [-fnvq] add | delete | change | get [-net | -host] destination gateway
[args]
```

Per evitare la redirezione si possono installare rotte specifiche per ogni sottorete usando singole istruzioni di route, così che gli host possano comunicare efficacemente tra di loro.

Installazione di rotte statiche all'avvio:

- In Unix il file è denominato /etc/init.d/inetinit:
- In Linux, il file è denominato /etc/rc.d/rc.local.

M2 U7 L2 - Instradamento Intradominio

Un dominio di instradamento è un insieme di reti collegate da router che sono tutte sotto la stessa responsabilità amministrativa della stessa organizzazione.
Questi domini costruiscono automaticamente le loro tabelle di instradamento attraverso protocolli distribuiti di comunicazione tra i router e che prendono il nome **Interior Gateway Protocols (IGP)**.
Ci sono due di protocolli principali:

- **RIP** (Routing Information Protocol)
- **OSPF** (Open Shortest Path First)

C'è poi un terzo protocollo che è stato utilizzato: **IS-IS** (Intermediate System to Intermediate System)

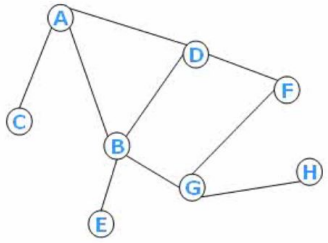
RIP

Basato su UDP, alla base di RIP c'è l'idea del Distance Vector, che prevede che ciascun nodo costruisca una tabella con i vari indirizzi di destinazione, il next-hop (primo noto successivo verso la destinazione) e il costo complessivo per raggiungere il nodo, attraverso lo scambio di messaggi tra i vari nodi

Formato di un pacchetto RIP (da 24 a 504 byte).

8	16	32	
Command	Version (1)	0	
Address Family Identifier		0	
IP Address			
0			$\times N \leq 25$
0			
Metric			

- Command: identifica se il pacchetto è di tipo Request o Response (1 o 2)
- Version: Identifica la versione di RIP
- Address Family Identificare (nel caso di IP: 2)
- IP address (è chiaro)
- Metric (è chiaro)



A

Indirizzo	Hop succ.vo	Costo
A	A	0
B	B	1
C	C	1
D	D	1
E	B	2
F	D	2
G	B	2
H	B	3

D

Indirizzo	Hop succ.vo	Costo
A	A	1
B	B	1
C	A	2
D	D	0
E	B	2
F	F	1
G	B	2
H	B	3

Quando viene aggiunto un nuovo nodo, questo comunica con i nodi immediatamente adiacenti a lui.
Il primo con cui comunicherà gli passerà tutte le righe della sua tabella, andando a fare una prima versione della tabella segnando i costi complessivi come quelli del nodo di provenienza + 1.
Quando comunicherà con gli altri, sostituirà le righe per cui il costo complessivo risulta minore rispetto a quello che ha attualmente, e aggiungerà le destinazioni che fino a quel momento non sapeva esistessero.

Formalmente, **l'algoritmo Distance Vector** funziona così:

- Inizia con tutte le destinazioni a distanza infinita, tranne che per il nodo stesso, la cui distanza è 0.
- Ogni 30 secondi, o quando si verifica un cambiamento nella tabella, ogni nodo invia la tabella ai vicini.
- Se un nodo j riceve da un nodo i la distanza d_{ip} verso un prefisso di rete P e questa distanza (più la distanza d_{ji} fino a i) risulta inferiore alla distanza d_{jp} (già nota fino a P), ossia:

  $d_{jp} >= d_{ji} + d_{ip}$

  si aggiorna la distanza verso P e si inviano al nodo "*i*" i pacchetti diretti a P.

Problemi di RIP: limiti

- Diametro di rete limitato: solo 15 hop nella sua implementazione iniziale, non adatto a grandi interreti.
- Convergenza lenta: protocollo multiround (solo dopo molti messaggi si hanno tabelle complete)
- Instradamento con classi: non utilizzabile in ambiente CIDR.

Problema del count-to-infinity
Si tratta di un problema caratteristico nelle "catene" di reti.

1. Il tratto da B a A si interrompe e B cerca un nuovo cammino per A.
2. C annuncia a B che A risulta raggiungibile tramite lui stesso e la distanza è 2 ma non dice che il cammino comprende proprio il tratto B-A interrotto.
3. Il cammino non è accettabile, ma al round successivo D annuncia a B che A risulta raggiungibile tramite lui stesso e la distanza è 3. Ancora, il cammino comprende il tratto B-A
interrotto.. E così via!

<u>>> Praticamente l'host per cui si interrompe il tratto si considerano buoni i cammini forniti da tutti gli altri host, anche se questi cammini prevedono comunque che si passi per l'host per cui è stata interrotta la connessione</u>

	1	2	3	4	Iniziale
	3	2	3	4	1 iterazione
	3	4	3	4	2 iterazioni
	5	3	5	4	3 iterazioni
	5	6	5	6	4 iterazioni

Il count-to-infinity è un problema:
- genera moltissimi aggiornamenti di instradamento, quindi causa troppo traffico;
- la rete dovrebbe rilevare che una destinazione è irraggiungibile.

Una possibile soluzione:
- Limitare superiormente il diametro della rete (rimane finito il numero di cammini)

RIP2 ha introdotto dei miglioramenti:
Split horizon: un router non comunica una distanza al vicino da cui ha appreso la distanza distanza (se C ha saputo da B l'esistenza di A, si impedisce a C di annunciare distanze a B)
Split horizon con poison reverse: se B annuncia a C la distanza minima verso un nodo A, C comunica ad B una distanza infinita verso A.
Queste tecniche impediscono però di scoprire dei percorsi alternativi (in caso ci siano) per arrivare al nodo per cui il collegamento è interrotto, e quindi funzionano solo per cicli che coinvolgono due nodi, su reti a catena.
Aggiornamento automatico
Invece di aspettare, l'aggiornamento viene inviato immediatamente, senza più l'attesa periodica per l'invio degli aggiornamenti.
Inoltre, RIP2 aggiunge una maschera di rete e un indirizzo next-hop nei pacchetti originali (così da permettere instradamento su subnet)

M2 U7 L3 - Da RIP a OSPF

Rotte attive:
- possono essere aggiornate da RIP;
- ci si aspetta che un gateway attivo fornisca ai vicini informazioni sull'instradamento;
- il gateway attivo sarà eliminato se per un po' non fornisce aggiornamenti sull'instradamento.

Rotte passive:
- rotte statiche permanenti;
- il protocollo di instradamento non aggiorna dinamicamente queste rotte per riflettere le condizioni di cambiamento della rete.

OSPF (Open Shortest Path First)
Basato sul link-state: ogni router condivide le informazioni sui suoi vicini con tutta la rete, così che ogni router si costruisca una immagine complessiva di tutta la rete.
OSPF definisce una gerarchia dell'area di instradamento in termini di ***sistemi autonomi (autonomous systems)***:
- ***stub o pozzo***: tutti i pacchetti che entrano nel dominio sono diretti nel dominio stesso (es. interrete azienda)
- ***transito***: *alcuni* dei pacchetti che entrano nel dominio sono diretti a altri domini (es. domini provider)
- ***backbone***: tutti i pacchetti che entrano nel dominio sono diretti a altri domini.

Instradamento "link state": IS-IS e OSPF
- Ogni router conosce l'intera rete, diversamente dal distance vector (intera rete → in termini di dominio di instradamento)

- Se l'intera rete è nota, ogni router calcola tutti gli instradamenti di costo minimo (e da questa si genererà la tabella di instradamento)
- Ogni router annuncia ai suoi vicini a *chi è connesso*.
- Ogni router manda a tutti gli altri router *qualsiasi annuncio che gli arriva* (invece che mandare solamente le righe della tabella SUA, così istituendo un ambito di broadcast)

Fasi di un algoritmo di instradamento di tipo "link state":
1. Invio del messaggio HELLO per stabilire a chi è connesso il router.
2. Invio, in modalità flooding affidabile (quando raggiungono un destinatario questi vengono scritti su tutte le porte di uscita esclusa quella di provenienza), di messaggi LSA (Link State Advertisement) per diffondere informazioni aggiornate sullo stato dei collegamenti.
3. Calcolo del cammino più breve verso tutti i nodi conosciuti.

M2 U7 L4 - OSPF Avanzato

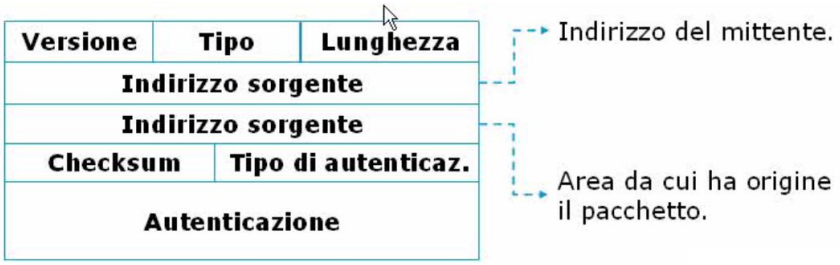
In OSPF ogni router partecipante invia periodicamente un messaggio HELLO detto LSP (Link State Packet) ai suoi vicini, e ogni vicino risponde con un messaggio HELLO.
In questo modo il router può stabilire a quali altri router è connesso.
Il periodo di default (HELLO time) va dai 10 ai 30 sec.

Ogni LSP contiene:
- ID del nodo che ha creato il messaggio LSP;
- lista dei vicini direttamente connessi a quel nodo con il costo di ogni collegamento;
    - ogni elemento della lista dei collegamenti si chiama Link State Assessment (LSA).
- un numero di sequenza;
- un tempo di vita

Questo permette un flooding affidabile. Chi riceve un LSP, risponde con un ACK.

Il numero di sequenza funge come una specie di timestamp, ed è lungo 32 bit. Quando un nodo genera un **LSA** (Link State Assessment, e cioè un Link State Packet che contiene lo stato del collegamento verso uno specifico vicino), questo viene numerato.
Questo fa sì che se si riceve 2 volte uno stesso LSA su uno stesso collegamento si sa sempre qual è il più recente, mentre il più vecchio viene scartato.
È possibile inviare anche degli **LSR** (richieste esplicite da parte di un router ad un altro di un LSA).
Ciascun LSA è contrassegnato da un TTL, e quando arriva a 0 viene scartato e il router che lo ha scartato invia a tutti un LSA con TTL = 0 per comunicare agli altri router di cancellare queste informazioni.

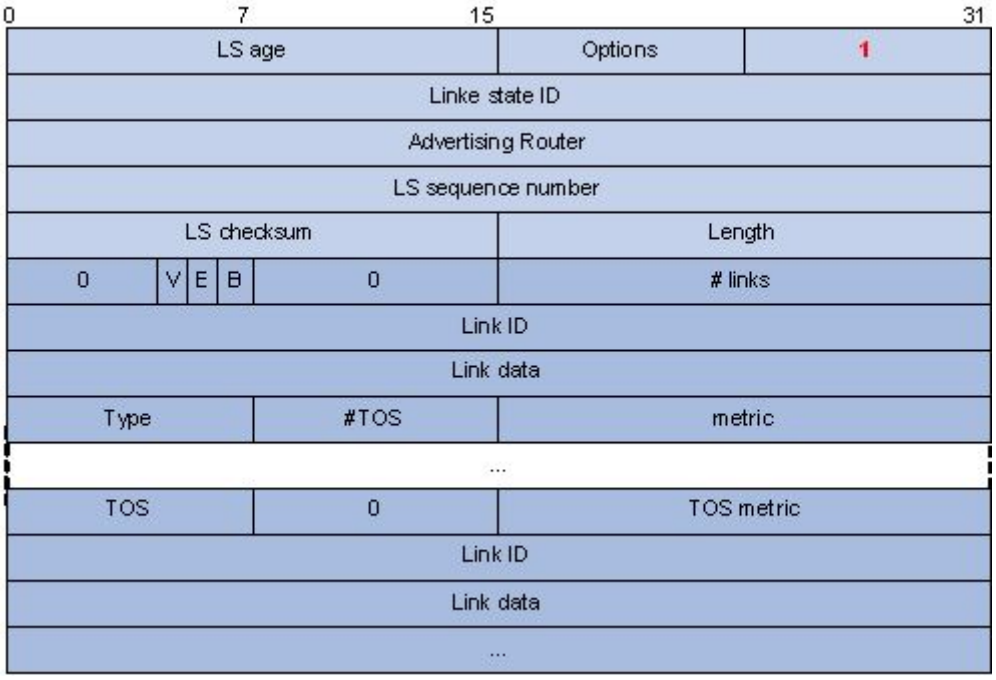
Intestazione OSPF



Tipo:
1 = messaggio di HELLO
2 = descrizione del database (DD, Database Description)
3 = richiesta dello stato dei collegamenti (LSR, Link State Request)
4 = aggiornamento sullo stato dei collegamenti (LSU, Link State Update)
5 = conferma di ricezione dello stato dei collegamenti (LSA, Link State Ack)

L’autenticazione permetterà di autenticare il mittente del messaggio, in modo da evitare che ci siano mittenti malevoli.

M2 U7 L5 - Metriche OSPF



Speigazioni sui campi:
Link ID

1 Neighboring router's Router ID
2 IP address of Designated Router
3 IP network/subnet number
4 Neighboring router's Router ID

Link Type (o “Type”)

1 Point-to-point connection to another router
2 Connection to a transit network
3 Connection to a stub network
4 Virtual link

TOS

0 Default se non specificato l’optional Tos

Quando un nodo viene reinizializzato, non conosce il suo numero ID. Quindi:

- Manda a tutti il suo stato di collegamento con numero di sequenza 0.
- Invia una richiesta dello stato dei collegamenti ai suoi vicini.
- I vicini rispondono con gli LSA più aggiornati che hanno, compreso l’LSA di quel nodo rilevato prima del momento del crash (in questo caso, il nodo aggiornerà il suo ID)
- Ogni volta che un router ottiene una nuova informazione la manda a tutti.

Scoperta dei vicini e sincronizzazione DB

- Un router appena reinizializzato esegue il multicast di pacchetti Hello OSPF su tutte le interfacce in grado di gestire OSPF.
- I vicini risponderanno confermando di esserci, consentendo al router reinizializzato di ricostruire l’elenco dei suoi vicini.

- Il router invierà quindi ai suoi vicini una richiesta “Database Description” vuota, e questi risponderanno con i loro Database Description (che sarà un insieme delle sole intestazioni degli LSA, senza i dati di costo).
- A sua volta, il router precedentemente reinizializzato risponderà con la sua DB Description, che parlerà solo di lui stesso (in quanto appena reinizializzato).
- I router vicini invieranno quindi conferma della ricezione.

Problemi di sicurezza OSPF
Un attaccante potrebbe inviare LSA falsi, fingendo di essere router conosciuti, compromettendo le funzionalità della rete.
La contromisura a questo è accettare LSA solo dai vicini, ma questo può essere un palliativo, in quanto i router vicini potrebbero essere stati già compromessi.
Quindi l’unica tecnica di difesa è l’uso dell’autenticazione.

Metriche
L’uso di LSA consente di usare metriche più sofisticate e pesate per i link.
- ***coda del router*** (vista come misura della congestione). Inizialmente sviluppato per Arpanet, sono accettabili ma molto mutevoli e reattive.
- ***costo statico (Juniper)***, come il ritardo di propagazione, ma questa a sua volta non tiene conto per nulla delle condizioni del traffico sullo stesso tratto.ì
- ***metrica dinamica***: basate sul campionamento, ad esempio misurando il round trip tra un router e l’altro. Tuttavia anche qui bisogna prestare attenzione:
    - Non può esistere più di un fattore di differenza 7 tra il collegamento più costoso e quello meno costoso, per non creare esagerate differenze tra i percorsi
    - Il costo di un collegamento può variare solo di un fattore di 3 tra due rilevazioni (i round trip tendono ad avere variazioni elevate)
    - Il costo dipende solo dall’utilizzo a carichi moderati o alti
    - Gli aggiornamenti si inviano quando il costo supera una certa soglia

Una caratteristica innovativa che ha OSPF è la possibilità di annunciare costi diversi per uno stesso collegamento in base al tipo di traffico che dovrà correre sul collegamento
Grazie al TOS (Type Of Service) si potrà definire quindi un costo per ogni determinata tipologia di pacchetto, così da avere una rete selettiva.

M2 U7 L6 - Instradamento interdominio e BGP
Lo scopo di questi protocolli è lo scambio di informazioni di instradamento tra sistemi autonomi, in specifico per far presente ai quali altri sistemi sono raggiungibili attraversi i propri vicini.
La famiglia di protocolli che si occupa di questi instradamenti si chiama **EGP (Exterior Gateway Protocol)**
Il funzionamento di questi protocolli è il seguente:
- un router di confine acquisisce un vicino al di là del confine: stabilisce il dialogo tra due gateway EGP usando Hello e I-heard-you;
- successivamente i due si scambiano degli annunci, cioè la lista dei sistemi autonomi raggiungibili tramite ciascuno di essi (questi avviene tramite una serie di colloqui di tipo poll update)

EGP non tenta di scegliere il cammino migliore

BGP (Border Gateway Protocol)
Il BGP, che è un protocollo della famiglia EGP, che si occupa degli annunci (di quali rotte annunciare se ne dovranno occupare i gestori dei domini autonomi).
Non tocca a BGP decidere cosa annunciare né quali annunci accettare.
Dunque, le politiche di instradamento non fanno parte del protocollo BGP: vengono fornite esternamente come informazioni di configurazione.

Il router di confine utilizzerà la politica per decidere quali rotte annuncerà tra le annunciabili e quali rotte in arrivo considererà valide.
BGP viaggia su TCP sulla porta 179, ed è un protocollo path-vector (le rotte sono scambiate nella forma di sequenze di ID di sistemi autonomi)
Questo protocollo funziona solo su ***router di backbone***, che connettono un sistema autonomo (AS) ad un altro AS.

Scelta di un protocollo di instradamento

- Reti piccole, la scelta è RIP (tra l'altro, già implementato su linux)
- Inter-reti più estese, la scelta è OSPF.
- Se serve un protocollo di instradamento esterno, si ricorre a BGP o ciò che l'altra parte sta utilizzando

Certi router supportano tutti i protocolli, ma non è realistico pensare che un host implementi nativamente dei protocolli avanzati (seppur alcuni linux fanno questo sporco lavoro).

Esiste un software versatile sotto linux, appunto, che permette di implementare diverse tecniche di instradamento, il **Gateway Daemon**. Tutti i protocolli sono configurati da un solo file (/etc/gated.conf).

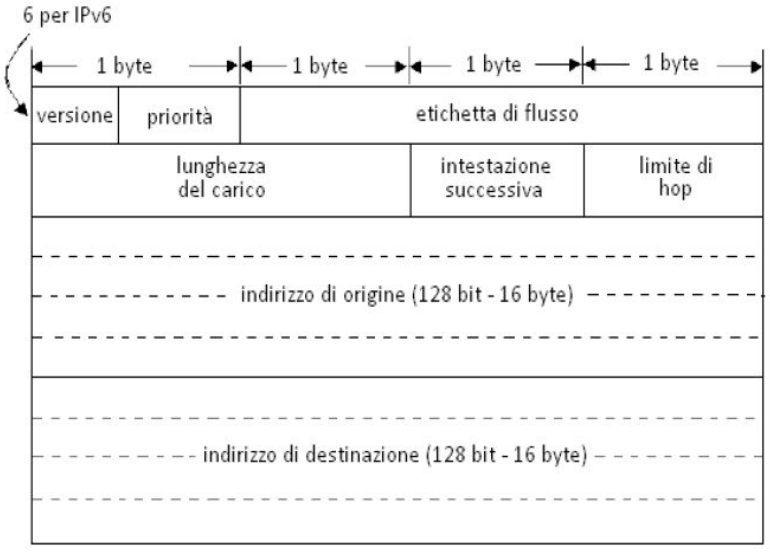
M2 U8 L1 - IPv6

Il protocollo IPv6 è supportato dalla maggior parte dei SO.
IPv6 nasce per far fronte ai diversi problemi derivati da IPv4:

- Scarsità dello spazio di indirizzamento di IPv4
- Meccanismo di transizione graduale e flessibile (a isole e piccole reti)
- Nuove capacità d'instradamento
- Qualità del servizio (capacità di assicurare una banda garantita a certe classi applicative).
- Sicurezza (IPv4 non fornisce alcuna funzionalità per preservare la confidenzialità)
- Capacità di aggiungere nuove funzioni in futuro

IPv6 ha l'intestazione fissa più semplice e piccola (solo 40 byte), e permette di gestire tramite le "chained header" diverse tipologie di intestazione.

Intestazione IPv6



> Campo di intestazione successiva: campo che permette di collegare dopo questo header un altro header contenente altri campi, così potendo comporre un'intestazione "custom"
> Versione: contiene 6 (versione di IP)
> Priorità: che verrà usata nel controllo della congestione
> Etichetta del flusso: serve per identificare una sequenza di pacchetti IPv6 come se fossero nello stesso flusso (così da fare traffic shaping direttamente a livello IP)

> Lunghezza del carico: dimensione in byte del carico. Se è 0, identifica il pacchetto di dimensione massima
> Limite in hop: simile a TTL di IPv4

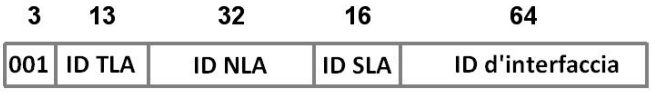
Intestazioni di estensione

- Instestazione di instradamento: contiene il percorso che deve fare il percorso per arrivare alla sua destinazione (così da permettere al pacchetto di seguire la strada definita dal mittente)
- Intestazione di frammentazione: supporta la frammentazione dei datagram IPv6
- Intestazione di autenticazione: consente al pacchetto di autenticarne la provenienza
- Intestazione ESP (Encapsulating Security Payload): contiene tutti i campi necessari per la cifratura e decifratura del payload IPv6

M2 U8 L2 - IPv6, parte 2

Gli indirizzi IP sono scritti come otto numeri esadecimali da 16 bit (128 bit totali)
→ 5f1b:df00:ce3e:e200:0020:0800:2078:e3e3
I bit più significativi determinano il tipo di indirizzo

Indirizzi unicast globali aggregabili



TLA: Top-Level Aggregation
NLA: Next-Level Aggregation
SLA: Site-Level Aggregation

(L'ID d'interfaccia si basa tipicamente sull'indirizzo MAC dell'hardware)

→ Il che permette di non necessitare più di ARP e affini

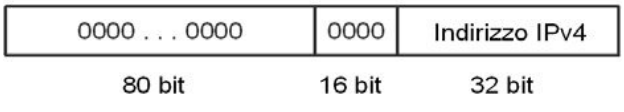
Indirizzo IPv6 mappato IPv4

Gli indirizzi mappati IPv4 consentono a un host che supporta sia IPv4 sia IPv6 di comunicare con un host che supporta solo IPv4
L'indirizzo IPv6 si basa completamente sull'indirizzo IPv4 e consiste di 80 bit posti a 0 seguiti da 16 bit a uno, seguiti da un indirizzo IPv4 a 32 bit (è praticamente un padding dell'IPv4

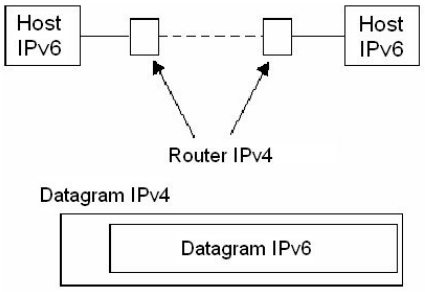
Questo si integra bene con il funzionamento di DNS, dove se si richiede da un'app IPv6 un nome che è associato ad un indirizzo IPv4, DNS crea automaticamente l'indirizzo IPv6 mappato IPv4.
Il kernel capirà che si tratta di un indirizzo IPv4 e si comporterà di conseguenza.

Indirizzo IPv6 compatibile IPv4

Un indirizzo compatibile IPv4 consente a un host che supporta IPv6 di parlare IPv6 anche se il router o i router locali non parlano IPv6
Gli indirizzi compatibili IPv4 avvisano il software del mittente di creare un tunnel, incapsulando il pacchetto IPv6 in un pacchetto IPv4, così permettendo al pacchetto di transitare in un tratto di rete compatibile solo con IPv4.



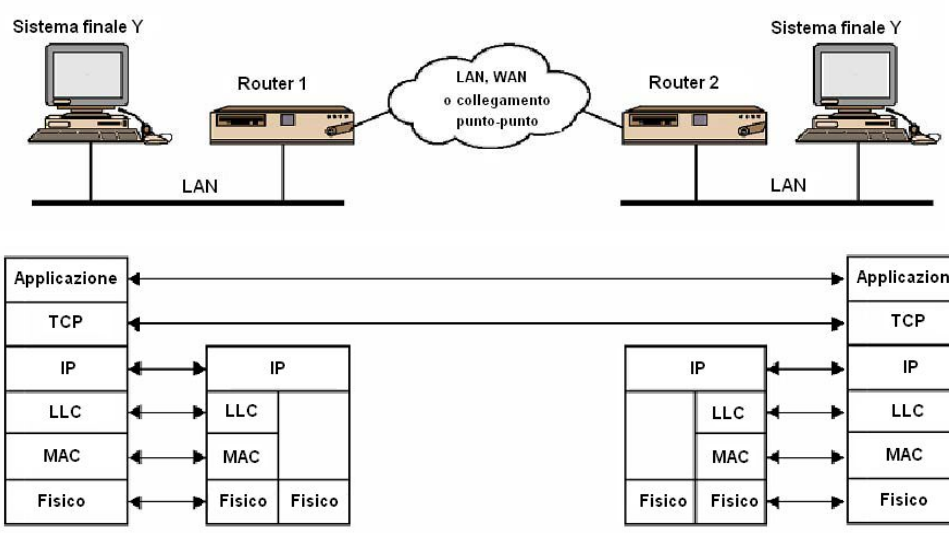
(differisce dal "mappato" per i 16 bit a 0 invece che a 1)



L'uso degli indirizzi mappati e compatibili IPv6 - IPv4 permetterà di conservare l'interoperabilità tra le varie applicazioni di rete che ancora utilizzano una rete IPv4.
L'introduzione di IPv6 prevede comunque che i programmatori utilizzino delle librerie socket estese appositamente.
Queste librerie consentono di creare "**server doppi**", che sono in grado di gestire due tipologie di traffico (IPv6 e IPv4), a patto che il server utilizzi un indirizzo IPv6 mappato IPv4 (così da essere raggiungibile da tutti).

M2 U9 L1 - Sicurezza IP

Scenario:



IPsec
IPsec è un insieme di protocolli aggiuntivi (framework) per consentire ad una coppia di host che usano IP di comunicare secondo certe proprietà di sicurezza.
Questo può diventare necessario quando una comunicazione che si vuole mantenere sicura deve passare per l'internet pubblica.

Viene aggiunta all'intestazione IP una intestazione per la gestione dei servizi di sicurezza (facile con IPv6, meno con IPv4). L'host ricevente (che gestisce IPsec) utilizzerà le informazioni contenute nell'intestazione IPsec per gestire tutte le fasi della sicurezza.

Vantaggi di IPsec

- È trasparente alle applicazioni: posto sotto al livello trasporto (TCP, UDP), e gestite nello stack
- Mette a disposizione servizi (sceglibili arbitrariamente) di sicurezza ai singoli utenti
- Può assicurare che:
    - un advertisement di un router o di un dispositivo vicino arrivi da un router autorizzato
    - un messaggio di reindirizzamento provenga dal router a cui è stato inviato il pacchetto iniziale
    - un aggiornamento d'instradamento non sia contraffatto

IKE (Internet Key Exchange)
IPsec permette di condividere una chiave di cifratura segreta tra M e R.

Il protocollo IKE, integrato con diverse RFC conseguenti permette di autenticare i due interlocutori, concordare il metodo di cifratura ecc. Esso prevede però alcune condizioni che potrebbero non essere soddisfatte da parte delle macchine della rete, come la presenza per ciascuna di essere di certificati, così da usare chiave pubblica e privata per scambiare una chiave simmetrica di cifratura.

Oltre a IKE, IPSec comprende altri servizi di sicurezza:

- Controllo degli accessi (comunicare con un router se si fa parte dell'ACL di quel router)
- Integrità (garanzia di non modifca dei pacchetti IP in transito)
- Autenticazione dell'origine dei dati
- Confidenzialità (cifratura) del flusso di traffico (proteggerli dallo sniffing)
- Rifiuto dei pacchetti originati da un attacco replay

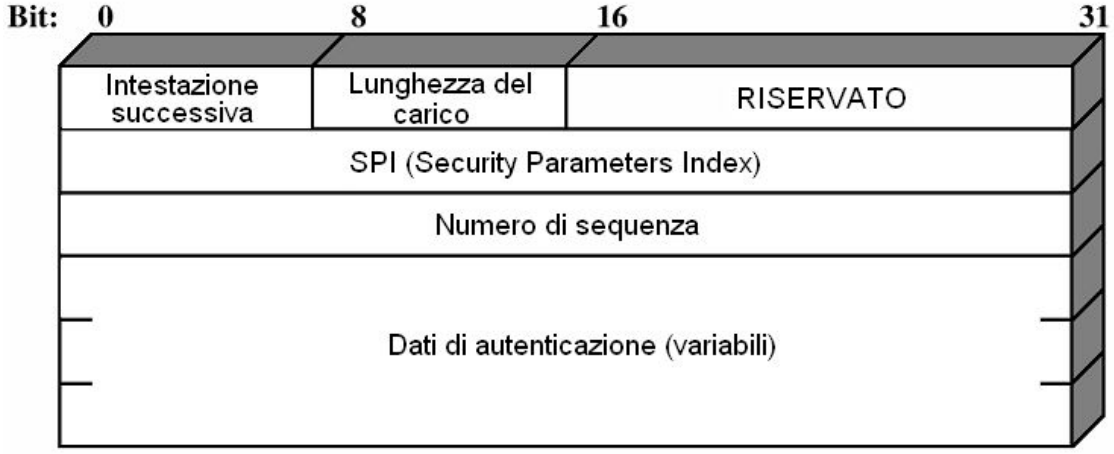
L'obiettivo di IPsec è quindi instaurare una SA (Security Association), ossia una connessione sicura tra mittente e destinatario (unidirezionale).
Una SA è identificata da tre parametri:

- SPI (Security Parameter Index, Indice dei parametri di sicurezza)
- Indirizzo IP di destinazione
- Identificatore del protocollo di sicurezza (AH o ESP)

Intestazione di autenticazione

- Fornisce supporto per l'integrità dei dati e l'autenticazione (codice MAC) di pacchetti IP
- Protegge da attacchi di tipo replay (garantendo la temporizzazione del pacchetto)

Header di autenticazione

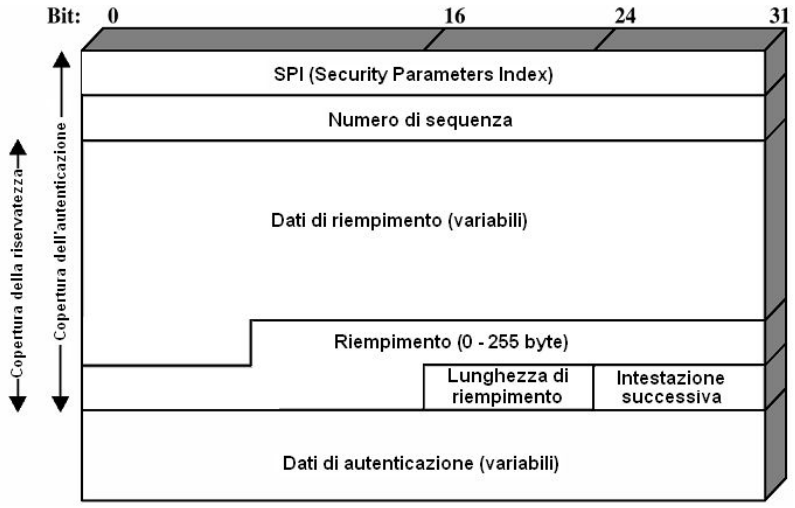


I dati di autenticazione conterranno i dati che servono al destinatario per garantire le proprietà che gli SPI elencano (ad esempio, per l'integrità, si dovrà avere una firma sul digest del pacchetto IP).

Le intestazioni ausiliarie di IPsec possono essere usate per garantire la sicurezza end-to-end (host a host). IPSec permette l'autenticazione anche tra router e host e viceversa (end-to-intermediate). Ciò torna utile quando un host di quella rete non supporta IPsec, ma il router a cui è collegato sì.

ESP (Encapsulating Security Payload)
Lo header aggiuntivo IPsec può gestire anche la confidenzialità del messaggio attraverso la sua cifratura. Chiaramente, la confidenzialità dovrà essere accompagnata dall'integrità.
Si usa la tecnica ESP. Questo permette sia di cifrare il payload di IP che di autenticare e garantire l'integrità non solo del pacchetto ma anche dell'intera intestazione (proteggendo anche agli attacchi di modifica dell'header).

Dati per ESP



Chiaramente il livello di sicurezza non dipenderà da IPsec in sé, ma dalle specifiche che vengono date ad esso (algoritmi di cifratura e di autenticazione ecc.)

Algoritmi di cifratura e di autenticazione
• Cifratura: Triplo DES a tre chiavi, RC5, IDEA, Triplo IDEA a tre chiavi, CAST, Blowfish)
• Autenticazione: HMAC-MD5-96, HMAC-SHA-1-96

APPENDICE

APPENDICE 1: iptables
Feature principali:
- Packet filtering stateless e stateful
- Supporto IPv4 e IPv6
- Ogni tipo di natting (NAT/NAPT)
- Infrastruttura flessibile ed estendibile
- Numerosi plug-in e moduli aggiuntivi per funzionalità aggiuntive

Iptables lavora su 3 tabelle (tables) di default:
`filter` - Regola il firewalling: quali pacchetti accettare, quali bloccare
`nat` - Regola le attività di natting
`mangle` - Interviene sulla alterazione dei pacchetti.

Ogni tabella ha delle catene (chains) predefinite (INPUT, OUTPUT, FORWARD ...) a cui possono essere aggiunte catene custom.
Ogni catena è composta da un elenco di regole (rules) che identificano pacchetti di rete secondo criteri diversi (es: `-p tcp --dport 80 -d 10.0.0.45`)
Ogni regola termina con una indicazione (target) su cosa fare dei pacchetti identificati dalla regola stessa (es: `-j ACCEPT, -j DROP` ...)

E' quella implicita e predefinita (-t filter)
Riguarda le attività di filtraggio del traffico.
Ha 3 catene di default:
INPUT - Riguarda tutti i pacchetti destinati al sistema. In entrata da ogni interfaccia.
OUTPUT - Riguarda i pacchetti che sono originati dal sistema e destinati ad uscire.
FORWARD - Riguarda i pacchetti che attraversano il sistema, con IP sorgente e destinazione esterni.

Esempio per permettere accesso alla porta 80 locale:
`iptables -t filter -I INPUT -p tcp --dport 80 -j ACCEPT`
Analoga a: `iptables -I INPUT -p tcp --dport 80 -j ACCEPT`

Esempio per permettere ad un pacchetto con IP sorgente 10.0.0.4 di raggiungere il server 192.168.0.1 attraversando il firewall:
`iptables -I FORWARD -s 10.0.0.4 -d 192.168.0.1 -j ACCEPT`

Il comando iptables viene usato per ogni attività di gestione e configurazione.

Inserimento regole:
`iptables -A CATENA ...` - Aggiunge una regola alla fine della catena indicata
`iptables -I CATENA [#] ...` - Inserisce alla riga # (default 1) una regola nella catena indicata
`iptables -N CATENA` - Crea una nuova catena custom
`iptables -P CATENA TARGET` - Imposta il target di default per la catena indicata

Rimozione regole e azzeramenti:
`iptables -F [catena]` - Ripulisce tutte le catene (o quella indicata)
`iptables -X [catena]` - Ripulisce tutte le catene custom (o quella indicata)
`iptables -Z [catena]` - Azzera i contatori sulle catene
`iptables -D catena #` - Cancella la regola numero # dalla catena indicata

Interrogazione:
`iptables -L` - Elenca le regole esistenti
`iptables -L -n -v` - Elenca, senza risolvere gli host, in modo verboso le regole esistenti

Target e Jump
Ogni regola termina con la definizione di un TARGET che indica cosa viene fatto del pacchetto. Molti dei target principali (ACCEPT, DROP, REJECT...) determinano l'interruzione della catena: il pacchetto matchato segue le indicazioni del Target e non vengono considerate le catene successive.
Come target si può anche impostare una catena custom nella quale "saltare" (jump `-j`) per procedere nell'attraversamento delle regole.

Target principali

`-j ACCEPT` - Il pacchetto matchato viene accettato e procede verso la sua destinazione. Si usa per definire il traffico permesso.
`-j DROP` - Il pacchetto viene rifiutato e scartato, senza alcuna notifica al mittente. Si usa, in alternativa a REJECT, per bloccare traffico.
`-j REJECT` - Il pacchetto viene rifiutato. Al mittente viene mandato un pacchetto (configurabile) di notifica tipo ICMP port-unreachable (**`--reject-with icmp-port-unreachable`**)
`-t LOG` - Il pacchetto viene loggato via syslog e procede l'attraversamento della catena. Opzioni: (**`--log-level`**, **`--log-prefix`**, **`--log-tcp-sequence`**, **`--log-tcp-options`**, **`--log-ip-options`**)
`-j DNAT` - Viene modificato l'IP di destinazione del pacchetto. Target disponibile solo in nat / PREROUTING e nat / OUTPUT. L'opzione **`--to-destination IP:porta`** definisce il nuovo IP di destinazione. Si usa tipicamente su network firewall che nattano server di una DMZ
`-j SNAT` - Viene modificato l'IP sorgente. Solo in nat / POSTROUTING. Prevede l'opzione **`--to-source IP:porta`**. Si usa per permettere l'accesso a Internet da una rete locale con IP privati.

-j MASQUERADE - Simile a SNAT, si applica quando i pacchetti escono da interfacce con IP dinamico (dialup, adsl, dhcp...). Si usa solo in nat / POSTROUTING e prevede l'opzione **--to-ports porte**.
-j REDIRECT - Redirige il pacchetto ad una porta locale. Usabile solo in nat / PREROUTING e nat / OUTPUT è previsto per fare un transparent proxy (con proxy server in esecuzione sulla macchina con iptables)
-j RETURN - Interrompe l'attraversamento della catena. Se questa è una secondaria, il pacchetto torna ad attraversare la catena madre da punto in cui aveva fatto il salto nella secondaria. Se il RETURN è in una delle catene di default, il pacchetto interrompe l'attraversamento e segue la policy di default.
-j TOS - Usabile solo nella tabella mangle, permette di cambiare il TOS (Type Of Service) di un pacchetto con l'opzione **--set-tos**. Per un elenco dei parametri disponibili: **iptables -j TOS -h**
-j MIRROR - Curioso e sperimentale, questo target invia un pacchetto speculare al mittente. In pratica è come se facesse da specchio per tutti i pacchetti ricevuti. Da usare con cautela, per evitare attacchi DOS indiretti.