## Reti di Calcolatori

# Funzionalità e servizi del livello Rete

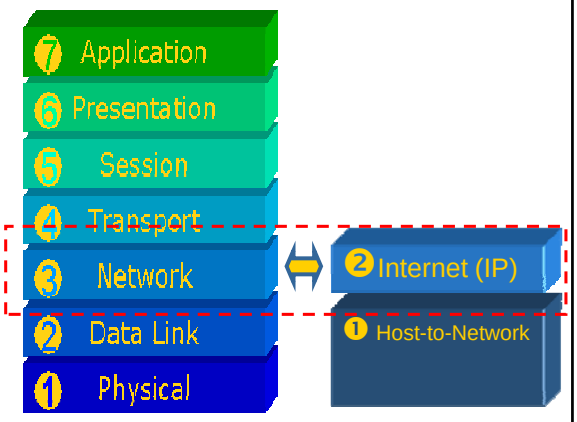
## Il livello Rete

- Funzionalità del livello Rete
- Organizzazione interna: Circuiti Virtuali e Datagrammi
- Algoritmi di Routing
- Controllo della congestione
- Internetworking

# Livello Rete: funzionalità

Funzionalità del livello Rete:

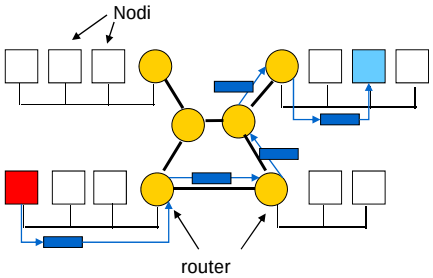
1. Interconnette gli host e le reti locali.
2. Controlla il cammino ed il flusso dei pacchetti (funzionalità di indirizzamento e di routing).
3. Gestisce la congestione della rete.
4. Consente la comunicazione tra reti di tipo diverso (internetworking)




3-3

# Indirizzamento e routing

- Deve essere utilizzato uno schema di indirizzamento globale
- I router instradano i pacchetti sulle linee di comunicazione in modo da consegnarli alla destinazione
- I percorsi sono scelti in base agli algoritmi di routing (più in là)




3-4

## Organizzazione del livello Rete

- **Circuiti virtuali (servizio con connessione)**
  - ➔ La creazione della connessione prevede la definizione del percorso di consegna dalla sorgente alla destinazione.
  - ➔ I pacchetti seguono tutti lo stesso percorso.
  - ➔ I pacchetti contengono indicazione dei circuiti virtuali a cui appartengono.
  - ➔ I router devono mantenere memoria dei circuiti virtuali che passano attraverso di essi.
- **Datagrammi (servizio senza connessione)**
  - ➔ I router hanno tabelle che indicano su quale linea di uscita inoltrare un pacchetto, a seconda dell'indirizzo del destinatario.
  - ➔ I pacchetti fra due host possono seguire percorsi diversi, perché non sono associati ad un circuito virtuale predefinito.
  - ➔ Non c'è garanzia sulla qualità del servizio offerto.



## Routing con una sottorete a datagrammi

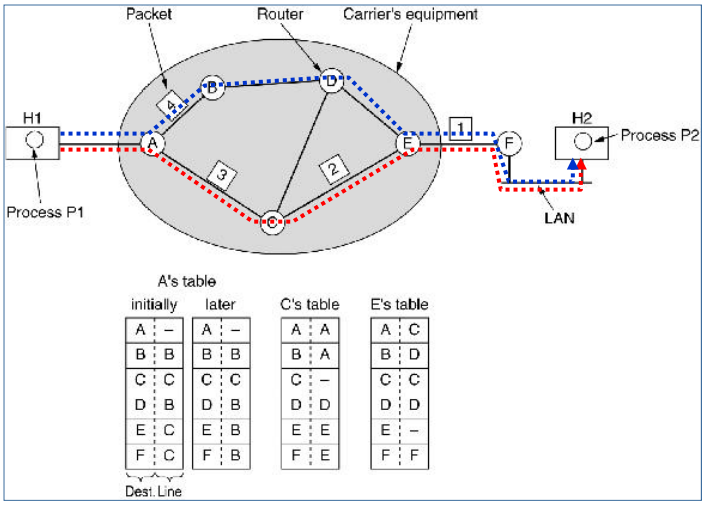



**Tabelle di instradamento**

## Routing con una sottorete a circuiti virtuali

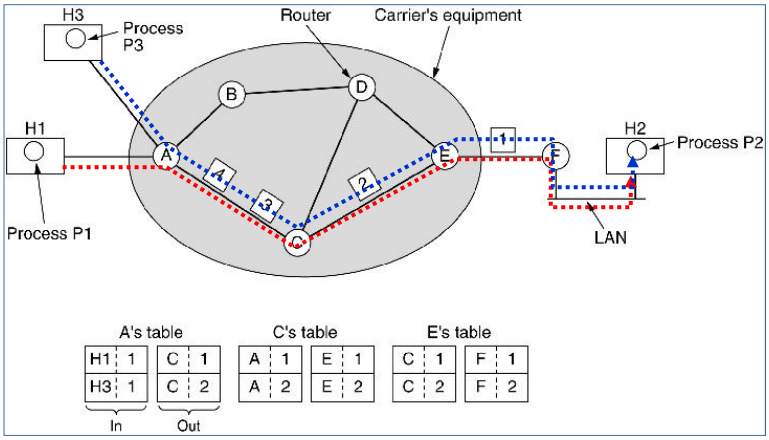



**Tabelle dei circuiti virtuali**



## Confronto: Datagrammi e Circuiti Virtuali

| Caratteristica | Reti basate su datagrammi | Reti basate su circuito virtuale |
|---|---|---|
| Creazione circuito | Non richiesto | Richiesto |
| Indirizzamento | Ogni pacchetto contiene gli indirizzi sorgente e destinazione completi | Ogni pacchetto contiene un piccolo numero VC (Virtual Circuit) |
| Informazioni di stato | La sottorete non conserva informazioni di stato | I router devono gestire le tabelle dei CV attivi |
| Instradamento | Ogni pacchetto è instradato indipendentemente | Percorso scelto alla creazione del circuito virtuale: tutti i pacchetti seguono questo percorso |
| Effetti dei guasti nei router | Nessuno, a parte i pacchetti persi durante il guasto | Tutti i circuiti virtuali che passano attraverso il router guasto vengono terminati |
| Controllo della congestione | Complesso | Semplice se può essere allocato spazio sufficiente in anticipo per ogni circuito virtuale |

# Algoritmi di routing

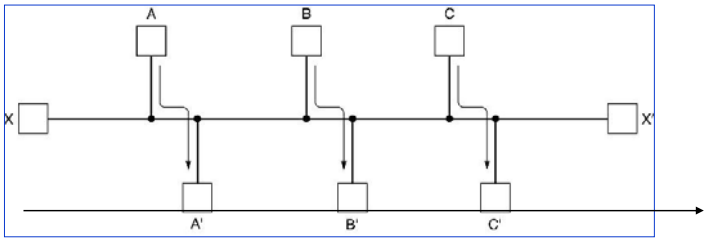
- **Definizone**

  Un algoritmo di routing decide quali percorsi usare per instradare i pacchetti dalla macchina sorgente alla macchina destinazione.
  Output dell'algoritmo: **tabelle di routing** sui vari router

- **Requisiti degli algoritmi di routing**
  - Correttezza: i pacchetti devono giungere a destinazione
  - Semplicità e efficienza
  - Robustezza: la rete deve continuare a funzionare anche in presenza di guasti
  - Stabilità: le informazioni nelle tabelle di routing devono raggiungere un punto di equilibrio
  - Imparzialità ed Ottimalità



# Compromesso tra imparzialità ed ottimalità




Si supponga che i dati trasmessi tra A e A', tra B e B', e tra C e C' siano in grado di saturare i collegamenti orizzontali.

Se si privilegia solo l'ottimalità (cioè il throughput complessivo) la comunicazione tra X e X' dove essere interrotta del tutto!

Occorre un compromesso tra efficienza globale e imparzialità (fairness) verso le singole connessioni.

# Generazione delle Tabelle di Routing

## Manuale

- Tabella creata ad hoc
- Utile in piccole reti
- Utile se l'instradamento non cambia.

## Automatica

- Un software crea/aggiorna le tabelle
- Necessario in grandi reti
- L'instradamento cambia in caso di guasti

# Principio di ottimalità e sink tree

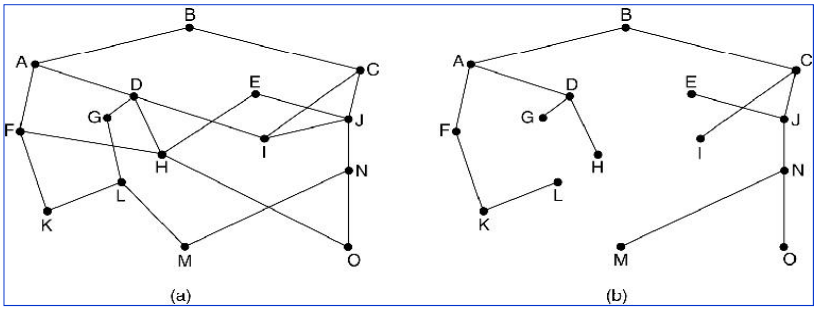
**Principio di ottimalità**: se il router **j** è nel cammino ottimo tra **i** e **k**, allora il cammino ottimo fra **j** e **k** segue lo stesso percorso, cioè si sovrappone al cammino ottimo tra **i** e **k**.

In conseguenza del principio di ottimalità, si ha che l'insieme dei cammini ottimi da tutte le sorgenti ad una destinazione forma una struttura ad albero, detta **sink tree**.

L'obiettivo degli algoritmi di routing è quello di ricostruire i sink tree o quanto meno di approssimarli il più possibile.

I sink tree sono quindi uno strumento per valutare le prestazioni degli algoritmi di routing.

## Sink tree: esempio




(a) Esempio di rete (b) Un sink tree per il router B

Il **principio di ottimalità** ed i **sink tree** forniscono un modello per misurare la bontà degli algoritmi di routing.

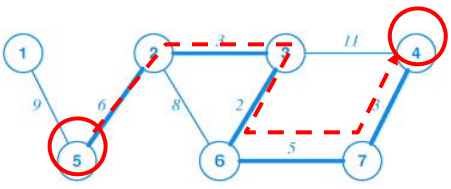


## Routing su cammino minimo

- Il principio di base è quello di costruire un grafo della rete, dove ogni nodo del grafo rappresenta un router ed ogni arco del grafo rappresenta una linea di comunicazione.
- Per scegliere un percorso tra due router, l'algoritmo cerca nel grafo il cammino più breve tra di essi.
- Metriche possibili (significato dei pesi sugli archi)
  - distanza
  - costi
  - capacità delle linee
  - traffico sulle linee
  - code sui router

## Cammino minimo: l'algoritmo di Dijkstra

- **L'algoritmo di Dijkstra** (1959) lavora su grafi orientati, che hanno pesi non negativi sui collegamenti.
- Questo algoritmo trova i percorsi più brevi tra un nodo di partenza e tutti gli altri.
- L'algoritmo è applicabile se si conosce lo stato di tutta la sottorete




Cammino minimo tra 5 e 4 (*19*).



## Algoritmo di "flooding"

Ogni pacchetto in arrivo viene inoltrato dal router su ogni linea in uscita eccetto quella da cui è arrivato il pacchetto. Per prevenire una duplicazione eccessiva:

- I pacchetti possono essere dotati di un contatore (**TTL**: Time To Live). Quando questo contatore raggiunge lo 0, il pacchetto viene eliminato.
- I pacchetti possono avere dei **numeri di sequenza**. I router tengono traccia dei messaggi ricevuti e già ritrasmessi, così da ignorare i duplicati.
- Nel **flooding selettivo**, i pacchetti in arrivo vengono inoltrati solo sulle linee che approssimativamente vanno nella direzione richiesta dalla sorgente.

Gli algoritmi di flooding vengono utilizzati come benchmark per il tempo di consegna dei pacchetti: infatti trovano sempre il cammino più breve, in quanto lo ricercano in parallelo.

# Routing su Internet: i Sistemi Autonomi

- Problema di scalabilità: il numero di host di Internet diventa sempre più grande. Le tabelle di routing crescono di conseguenza.
- La rete è perciò suddivisa in diversi **sistemi autonomi** (**AS**: Autonomous Systems). I router di uno stesso **AS** implementano tutti lo stesso algoritmo di routing.
- Almeno un router di ogni AS deve essere in grado di instradare i pacchetti verso l'esterno: si tratta dei **router di confine** (router gateway)
- L'algoritmo di routing interno ad un AS è detto **interior gateway protocol** (o intra-AS)
- L'algoritmo di routing utilizzato dai router di confine è detto **exterior gateway protocol** (o inter-AS)

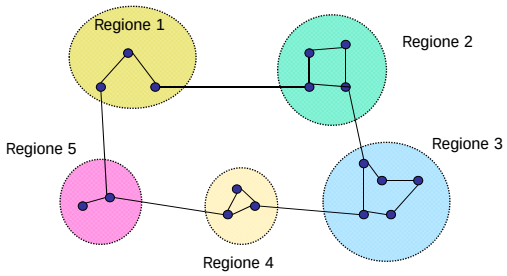


# Routing gerarchico

Due (o più) livelli di routing:

- un primo livello di routing **interno** ad ogni AS (o *regione*)
- un secondo livello di routing fra tutti i **router di confine**.

- **I router interni** sanno come arrivare a tutti gli altri router della stessa AS, compresi i router di confine
- **I router di confine** sono utilizzati per inviare i dati che devono pervenire ai router di confine di un'altra AS



# Protocolli di routing in Internet

- **Interior Gateway Protocol**

  I protocolli più utilizzati sono 2:

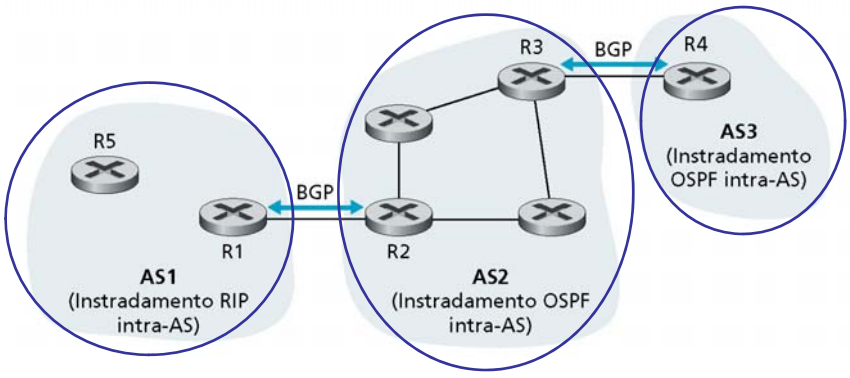
  **RIP** (Routing Information Protocol), di tipo **distance vector**
  **OSPF** (Open Shortest Path First), di tipo **link state**

- **Exterior Gateway Protocol**

  **BGP** (Border Gateway Protocol), di tipo **path vector**



# Esempio di routing gerarchico su Internet



# Routing Information Protocol (RIP)

Caratteristiche di RIP:

- È un algoritmo adattativo, di tipo **distance vector**
- Utilizza il conteggio degli hop come metrica di costo
- Ogni router gestisce la sua **tabella di routing** (o *vettore delle distanze*) contenente, per ogni rete destinazione, il numero di hop necessari per raggiungerla, ed il router successivo verso cui inoltrare i pacchetti
- Periodicamente (es. ogni 10 secondi) ogni router invia ai propri vicini la sua tabella di routing
- Quando riceve una tabella da un vicino, il router aggiorna la sua tabella



# Esempio di funzionamento di RIP

Le figure mostrano:
- parte di un AS
- la tabella di routing del router **D**

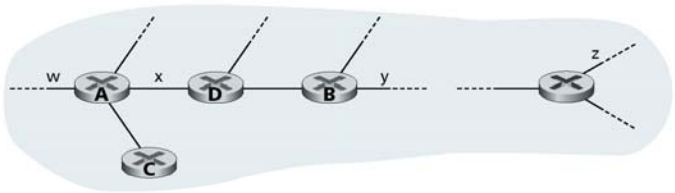



Tabella di routing di **D**

| Rete di destinazione | Router successivo | Numero di hop verso la destinazione |
|---|---|---|
| w | A | 2 |
| y | B | 2 |
| z | B | 7 |
| x | – | 1 |
| .... | .... | .... |

# Esempio di funzionamento di RIP

Le figure mostrano:

- la tabella di routing che il router **A** invia al router **D**
- la nuova tabella di routing del router **D**

Il router **D** ha imparato che esiste un percorso più corto verso la rete **z**: tale percorso passa per il router **A** (e poi per **C**) invece che per il router **B**

Tabella di routing di **A**

| Rete di destinazione | Router successivo | Numero di hop verso la destinazione |
|---|---|---|
| z | C | 4 |
| w | – | 1 |
| x | – | 1 |
| .... | .... | .... |

Nuova tabella di routing di **D**

| Rete di destinazione | Router successivo | Numero di hop verso la destinazione |
|---|---|---|
| w | A | 2 |
| y | B | 2 |
| z | A | 5 |
| .... | .... | .... |



# Open Shortest Path First (OSPF)

- È un algoritmo adattativo, di tipo **link state**
- Con il link state routing, ogni router costruisce una mappa topologica completa (un grafo) dell'intero AS
- Il costo di ogni link può essere uguale a 1, o può essere determinato dinamicamente tenendo conto del traffico: OSPF dà libertà di scelta su questo punto
- Ogni router invia informazioni a tutti i router dell'AS, non solo ai vicini, tramite un algoritmo di **flooding**
- Le informazioni sono inviate periodicamente, oppure quando si rileva un cambiamento nello **stato di un link** (es. costo del link)
- Ogni router calcola i percorsi minimi verso tutti gli altri router
- La tabella di routing è derivata dall'albero dei percorsi minimi (**Dijkstra**)

# OSPF: struttura dell'algoritmo

Con OSPF, ogni router effettua periodicamente i seguenti passi:

1. Scopre i router dell'AS ed i loro indirizzi inviando uno speciale pacchetto di **HELLO** su ogni linea di uscita.
2. Misura il costo del percorso inviando verso ognuno dei router conosciuti un pacchetto **ECHO**. Il pacchetto viene rimandato al mittente, che così può calcolare il tempo di ritardo.
3. Costruisce un pacchetto contenente tutto quello che ha appena scoperto sui percorsi. Per risolvere i problemi di consistenza e di convergenza, in ogni pacchetto vengono inseriti **l'identità** del router, un **numero di sequenza** progressivo (per capire se le informazioni sono aggiornate) e **l'età** del pacchetto (l'età è un intero che viene decrementato ad ogni inoltro: quando arriva a 0, il pacchetto viene scartato).
4. Spedisce il pacchetto a tutti i router tramite un algoritmo di **flooding**.
5. Grazie alle informazioni ricevute dagli altri router, il router calcola il **cammino minimo** verso tutti gli altri router della stessa AS con l'algoritmo di **Dijkstra**.

# Border Gateway Protocol

È il protocollo utilizzato su Internet per realizzare il routing inter-AS:

- è di tipo **path vector**, una variante del tipo distance vector
- gestisce il costo ed il cammino completo per le altre destinazioni
- un router può scegliere il prossimo router in base ai cammini che gli vengono proposti
- spesso il cammino è scelto per ragioni "politiche" oltre che tecniche

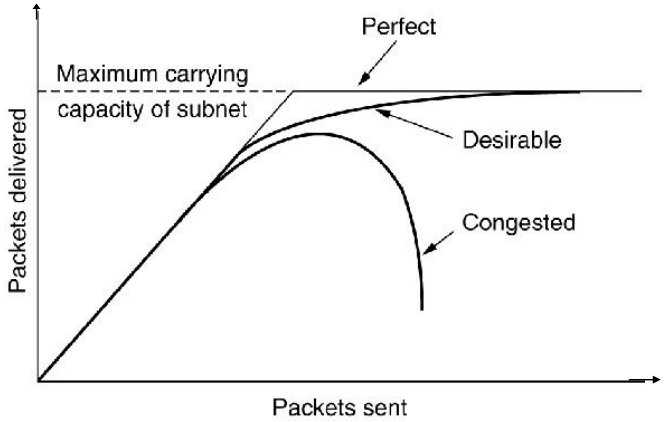
## Routing Multicast

- E' necessario quando un host deve inviare un messaggio ad un insieme di nodi di una rete.
- Approccio:
  1. i processi interessati chiedono di appartenere ad un gruppo, cui è assegnato un indirizzo logico (in Internet un **indirizzo IP di classe D**);
  2. gli host comunicano ai router di riferimento i gruppi cui sono iscritti (in Internet si usa il protocollo **Internet Group Management Protocol**);
  3. le informazioni si propagano sulla rete;
  4. i router elaborano uno **spanning tree** per ogni gruppo multicast: questo è possibile solo se è utilizzato il link state routing;
  5. i pacchetti destinati ad un gruppo sono inviati solo sulle linee che appartengono allo spanning tree;
  6. ogni router consegna il pacchetto agli host della sua rete locale iscritti al gruppo, e lo inoltra sulle linee dello spanning tree.

3-27

## Controllo della congestione

- Quando nella rete (o in una sottorete) sono presenti troppi pacchetti le prestazioni degradano (si ha **congestione**).




3-28

# Controllo della congestione

Il **controllo della congestione** è cosa diversa dal **controllo di flusso** che viene effettuato nei livelli 2 e 4 (per una singola connessione sorgente-destinazione)

La congestione è dovuta non solo ad un traffico troppo intenso, ma anche ad altre concause:

- i buffer dei router sono limitati: se sono pieni, i pacchetti in arrivo vengono scartati
- i processori dei router troppo lenti nell'elaborazione dei pacchetti
- le linee di trasmissione sono troppo lente

La congestione alimenta se stessa, perché i timeout del trasmettitore scadono (es. in TCP), ed i pacchetti vengono ritrasmessi aumentando così il traffico e peggiorando la congestione

# Controllo della congestione sui vari livelli

| Layer | Policies |
|---|---|
| Transport | • Retransmission policy<br>• Out-of-order caching policy<br>• Acknowledgement policy<br>• Flow control policy<br>• Timeout determination |
| Network | • Virtual circuits versus datagram inside the subnet<br>• Packet queueing and service policy<br>• Packet discard policy<br>• Routing algorithm<br>• Packet lifetime management |
| Data link | • Retransmission policy<br>• Out-of-order caching policy<br>• Acknowledgement policy<br>• Flow control policy |

# Traffic shaping

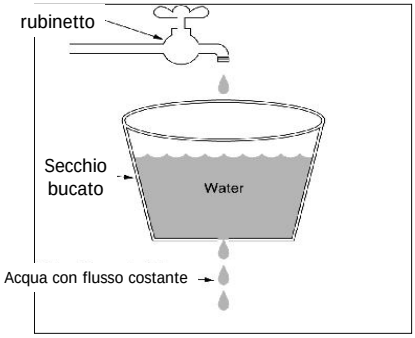
- La congestione è spesso dovuta al traffico irregolare (**bursty**).
- Se gli host trasmettessero a velocità (**data rate**) uniforme, la rete avrebbe maggiori chance di sostenere il traffico
  - *questo comportamento è dimostrabile anche analiticamente, mediante la teoria delle Reti di Code*
- Le tecniche di **Traffic Shaping** sono utilizzate per regolamentare il flusso di pacchetti che un host, durante la trasmissione di dati, immette nella rete. Due tecniche ben note sono:
  - **Leaky Bucket** (secchio bucato)
  - **Token Bucket** (secchio a gettoni)



# Algoritmo del secchio bucato

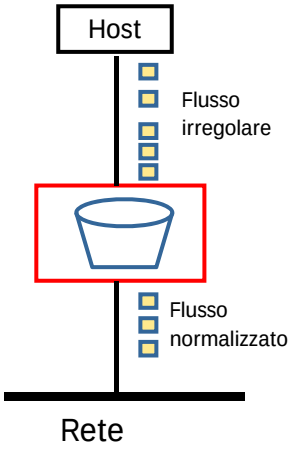
## Algoritmo leaky bucket

- consente di immettere i pacchetti nella rete secondo un *data rate* fissato
- mantiene nel buffer (il secchio) i pacchetti accodati per la trasmissione
- ad ogni $\Delta T$, un pacchetto viene prelevato dal buffer e trasmesso sulla rete
- possibile perdita di pacchetti, se il buffer va in overflow



# Algoritmo del secchio bucato

- Supponiamo che il *data rate* **in uscita** dal secchio, in bit per secondo, sia *R*
- Se il data rate medio **in ingresso** al secchio (cioè il traffico dati proveniente dai livelli superiori) è minore di *R,* va tutto bene
- Altrimenti dopo un po' si ha perdita di pacchetti perché il secchio si riempie





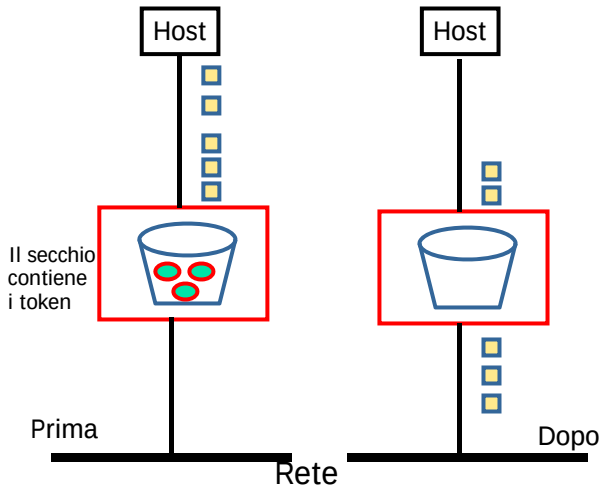

# Algoritmo del token bucket

Consente irregolarità controllate nel flusso di pacchetti immessi nella rete

- secchio di **token**, creati con cadenza prefissata, fino ad un numero massimo (M).
- per trasmettere un pacchetto deve essere disponibile un token (che viene eliminato al momento della trasmissione del pacchetto)
- quando non si trasmette nulla, si accumula un credito trasmissivo (un numero di token)
- quando si trasmette si può sfruttare tutto il credito di token disponibile, alla massima velocità consentita dalla linea
- dati k token nel secchio e h>k pacchetti da trasmettere, i primi k sono trasmessi subito e gli altri aspettano in una coda

## Algoritmo del token bucket

- I pacchetti non vengono mai scartati
- Se necessario, si avverte il livello superiore per fermare la produzione dati




3-35

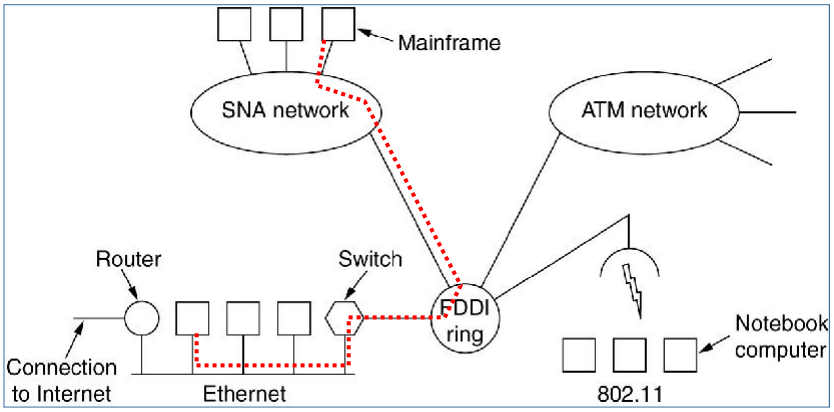
## Internetworking

- Connessione di reti eterogenee che si differenziano per:
  - tipi di servizio (con connessione o senza connessione)
  - formati e dimensione dei pacchetti
  - indirizzamento
  - meccanismi di controllo dell'errore
  - meccanismi di controllo della congestione
  - multicast (può essere supportato o non supportato)
  - politiche di sicurezza
  - politiche di accounting (per il pagamento)
  - ...

3-36

# Internetworking

Interconnessione di reti diverse




3-37

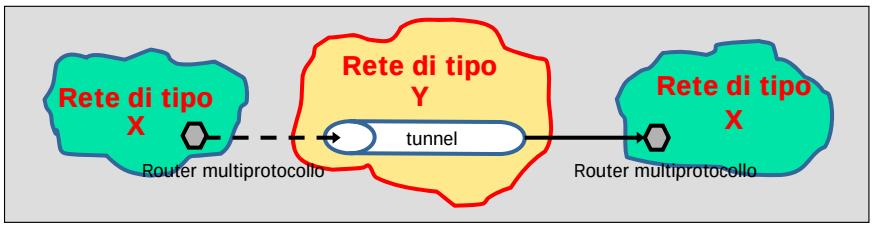
# Tecniche e soluzioni per l'internetworking

- Livello 1: i ripetitori copiano bit tra i cavi di interconnessione
- Livello 2: i bridge memorizzano e inviano frame di data link tra reti differenti (es. tra due reti LAN)
- Livello 3: i router multiprotocollo inviano pacchetti tra reti differenti, effettuando trasformazioni di formato e di indirizzo
- Livello 4: i gateway di trasporto interfacciano due connessioni di trasporto
- Livello applicativo: i gateway di applicazione esaminano i dati a livello delle applicazioni, interpretandone la semantica (es. gateway di posta elettronica)

3-38

# Tunneling

- Quando l'host sorgente e l'host destinazione appartengono allo stesso tipo di rete ma i pacchetti devono attraversare reti differenti si può usare la tecnica del tunneling (incapsulamento).

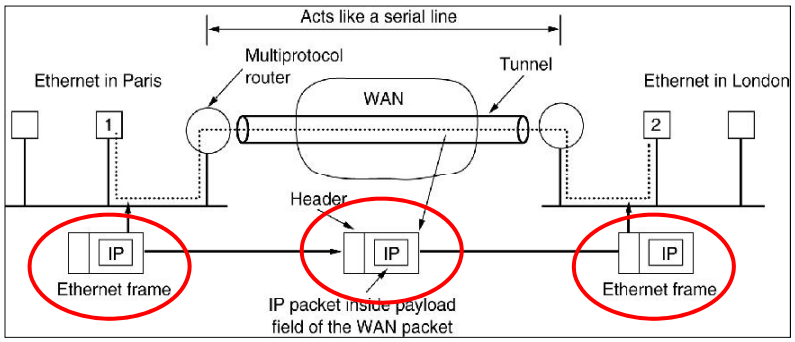



- la rete Y non ha router multiprotocollo.
- un router nelle due reti X è multiprotocollo, e incapsula i pacchetti delle reti di tipo X dentro pacchetti di tipo Y, consegnandoli alla rete di tipo Y.

3-39

# Esempio di Tunneling

- Invio di dati da una rete Ethernet ad un'altra rete Ethernet tramite una WAN.




3-40