

# **Reti** **(Computer Networks)**

## **Capitolo 4** **Livello di rete**

Docente: Paolo Casari

TA: Andrea Rosani

# Sommario

- Visione d'insieme
  - ❖ Data plane e control plane
- Come è fatto un router
- Il protocollo IP
  - ❖ Formato datagrammi
  - ❖ Frammentazione
  - ❖ Indirizzamento e NAT
  - ❖ Indirizzamento classless
  - ❖ Tipi di indirizzi
  - ❖ Address Resolution Protocol (ARP)
  - ❖ Internet Control Message Protocol (ICMP)
- Il protocollo IP (segue)
  - ❖ Dynamic Host Configuration Protocol (DHCP)
  - ❖ Il viaggio di un pacchetto
  - ❖ IPv6
- Protocolli di instradamento
  - ❖ Link state: Dijkstra
  - ❖ Internet, AS, OSPF
  - ❖ Distance vector: Bellman-Ford
  - ❖ RIP
- BGP

# **Parte 1**

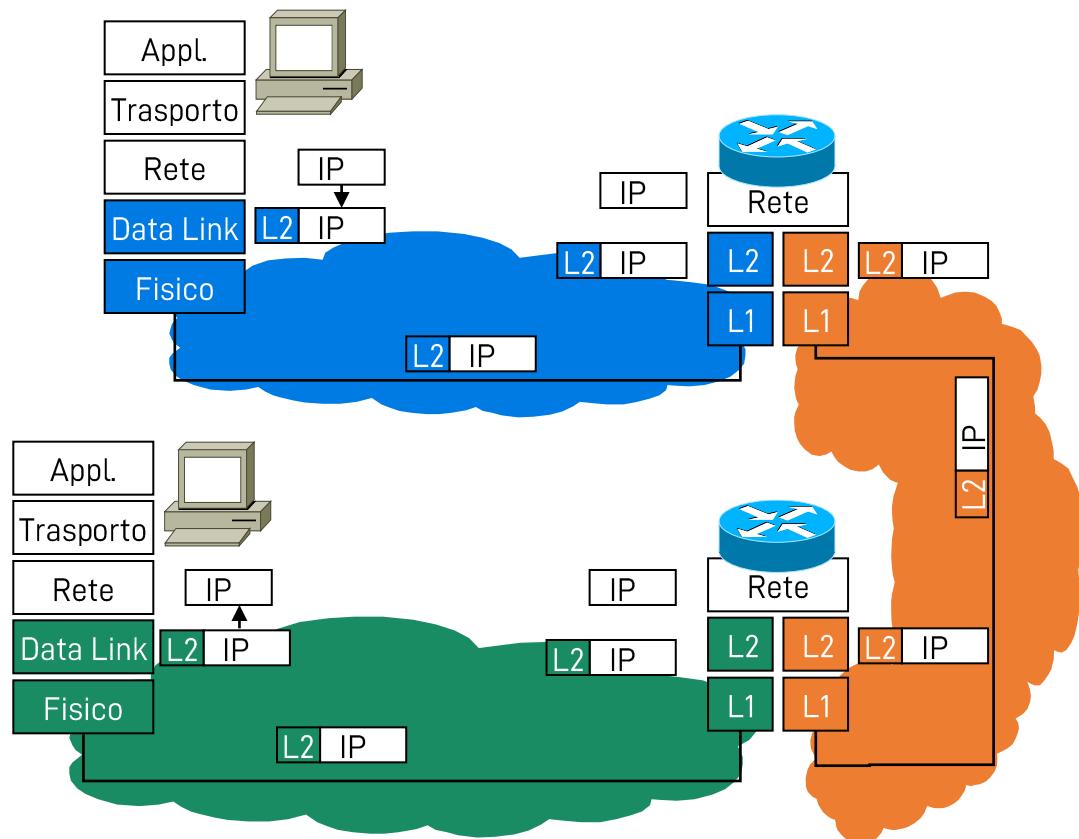
Livello di rete e protocollo IPv4

# Sommario

- Visione d'insieme
  - ❖ Data plane e control plane
- Come è fatto un router
- Il protocollo IP
  - ❖ Formato datagrammi
  - ❖ Frammentazione
  - ❖ Indirizzamento e NAT
  - ❖ Indirizzamento classless
  - ❖ Tipi di indirizzi
  - ❖ Address Resolution Protocol (ARP)
  - ❖ Internet Control Message Protocol (ICMP)
- Il protocollo IP (segue)
  - ❖ Dynamic Host Configuration Protocol (DHCP)
  - ❖ Il viaggio di un pacchetto
  - ❖ IPv6
- Protocolli di instradamento
  - ❖ Link state: Dijkstra
  - ❖ Internet, AS, OSPF
  - ❖ Distance vector: Bellman-Ford
  - ❖ RIP
- BGP

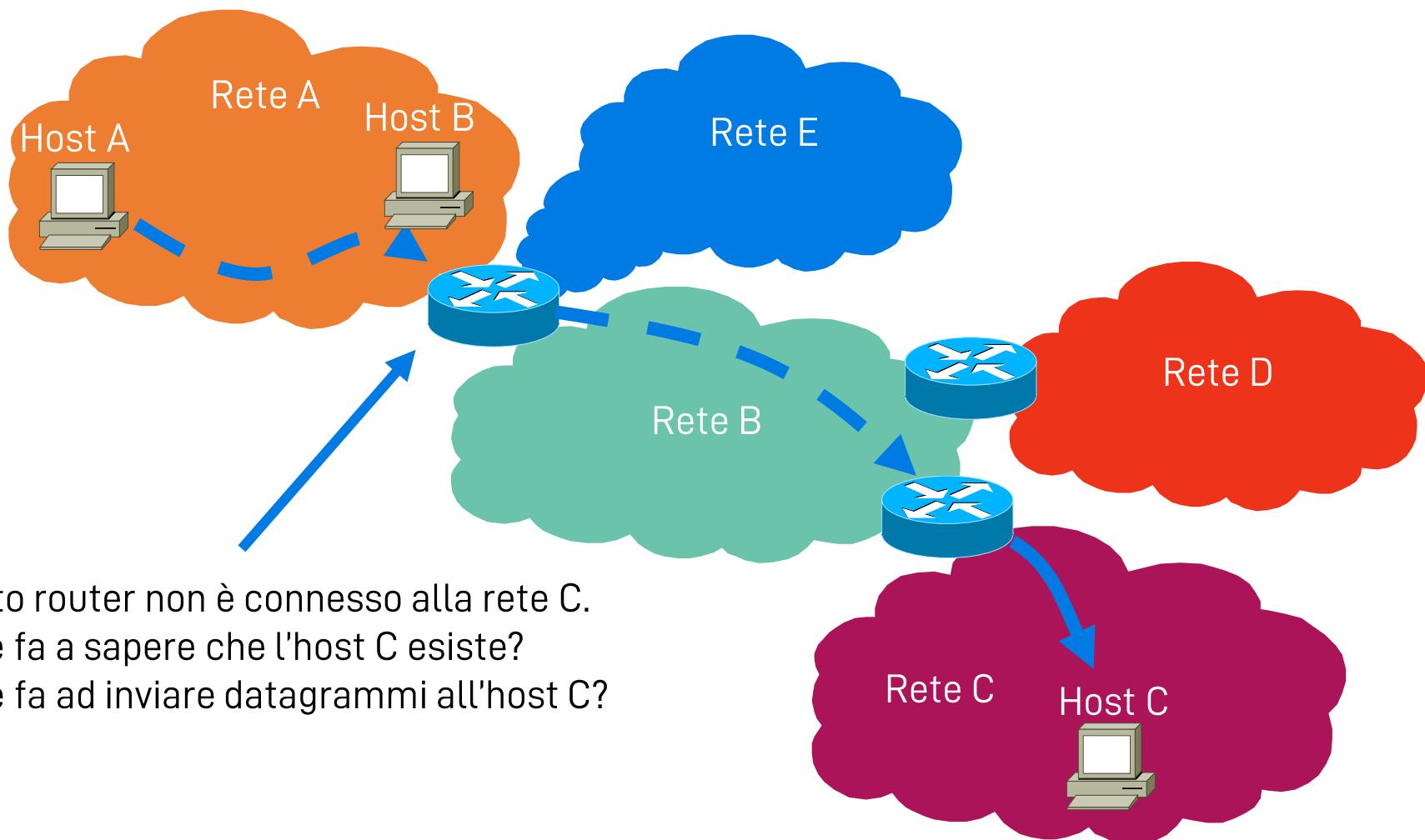
# Visione d'insieme

- Obiettivo: spostare pacchetti dal mittente al destinatario
- Al mittente, incapsula i segmenti in datagrammi
  - ❖ E poi in frame di livello 2
- Al ricevente, recupera i segmenti e li consegna al livello di trasporto
  - ❖ E da lì a un processo
- Ci sono protocolli di livello rete in tutti gli host e router
- I router ispezionano tutti gli header dei datagrammi che li attraversano per prendere decisioni di inoltro



Attenzione: reti diverse potrebbero usare *internamente* protocolli L2 o L3 diversi!

# Consegna diretta e indiretta



# Le due funzioni del livello di rete

## □ Inoltro (forwarding)

- ❖ Operazione locale con scala locale
- ❖ Spostamento del pacchetto da un ingresso del router a un'uscita

## □ Instradamento (routing):

- ❖ Operazione locale con scala globale
- ❖ Determinare il percorso che un pacchetto deve seguire
- ❖ Algoritmi di routing

Analogia: fare un viaggio

## □ Forwarding: spostarsi all'interno di uno snodo

- ❖ Cambiare treno
- ❖ Prendere uno svincolo autostradale
- ❖ ...

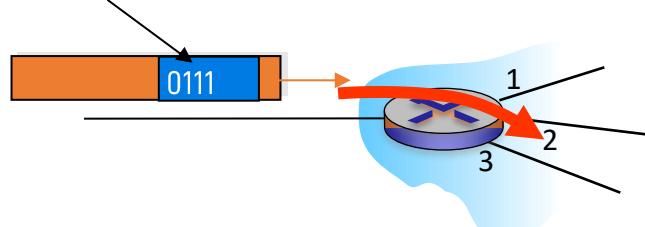
## □ Routing: pianificare tutte le tappe del viaggio dall'inizio alla fine

# Data plane e control plane

## Data plane ("Piano dati")

- ❑ Funzione locale a ogni router
- ❑ Determina come inoltrare un datagramma da una porta di arrivo a una porta di uscita del router
- ❑ Funzione di forwarding

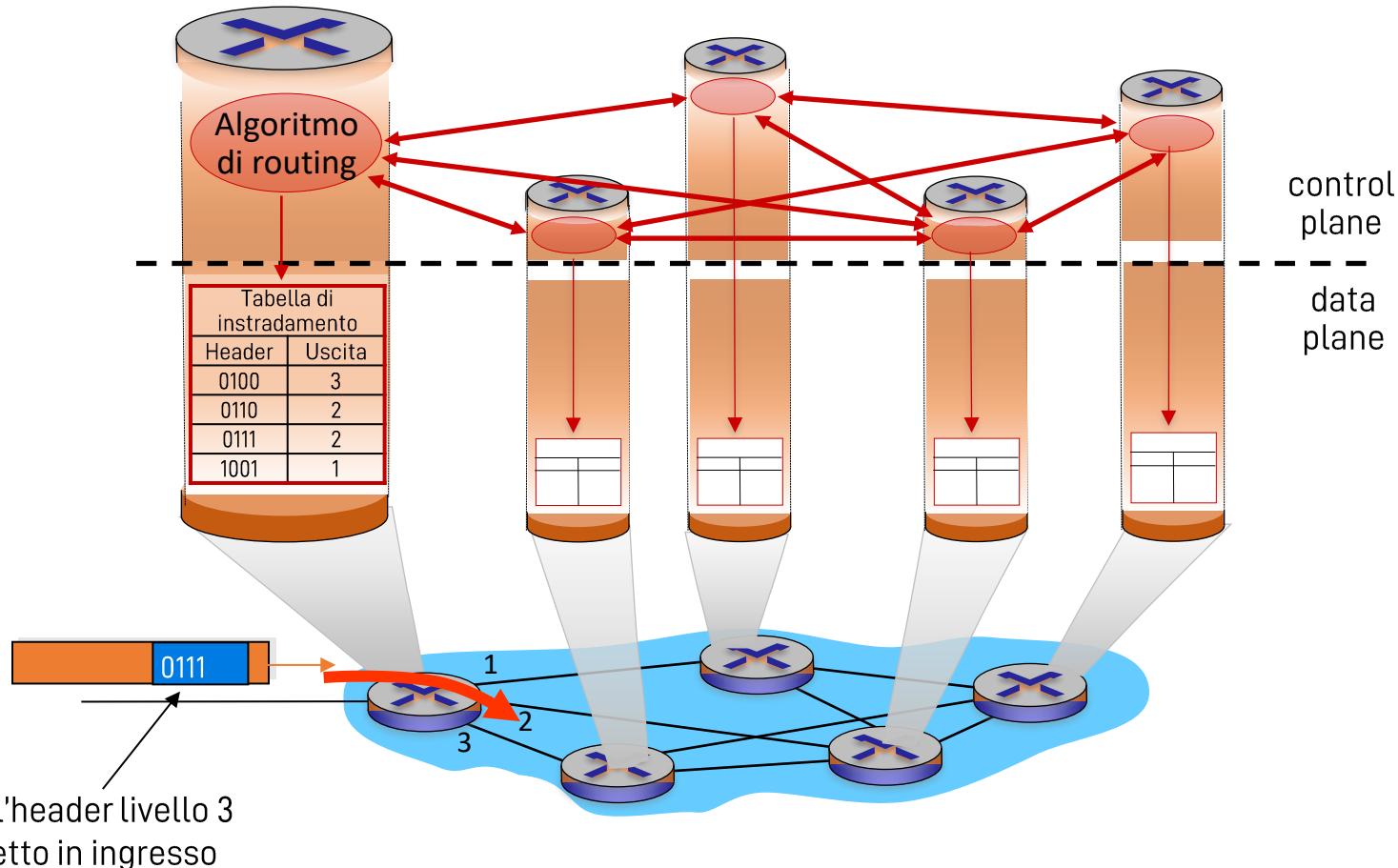
Campi dell'header livello 3  
del pacchetto in ingresso



## Control plane ("Piano controllo")

- ❑ Logica globale di rete
- ❑ Determina come instradare un datagramma in un percorso end-to-end, cioè dall'host mittente all'host destinatario

# Control plane

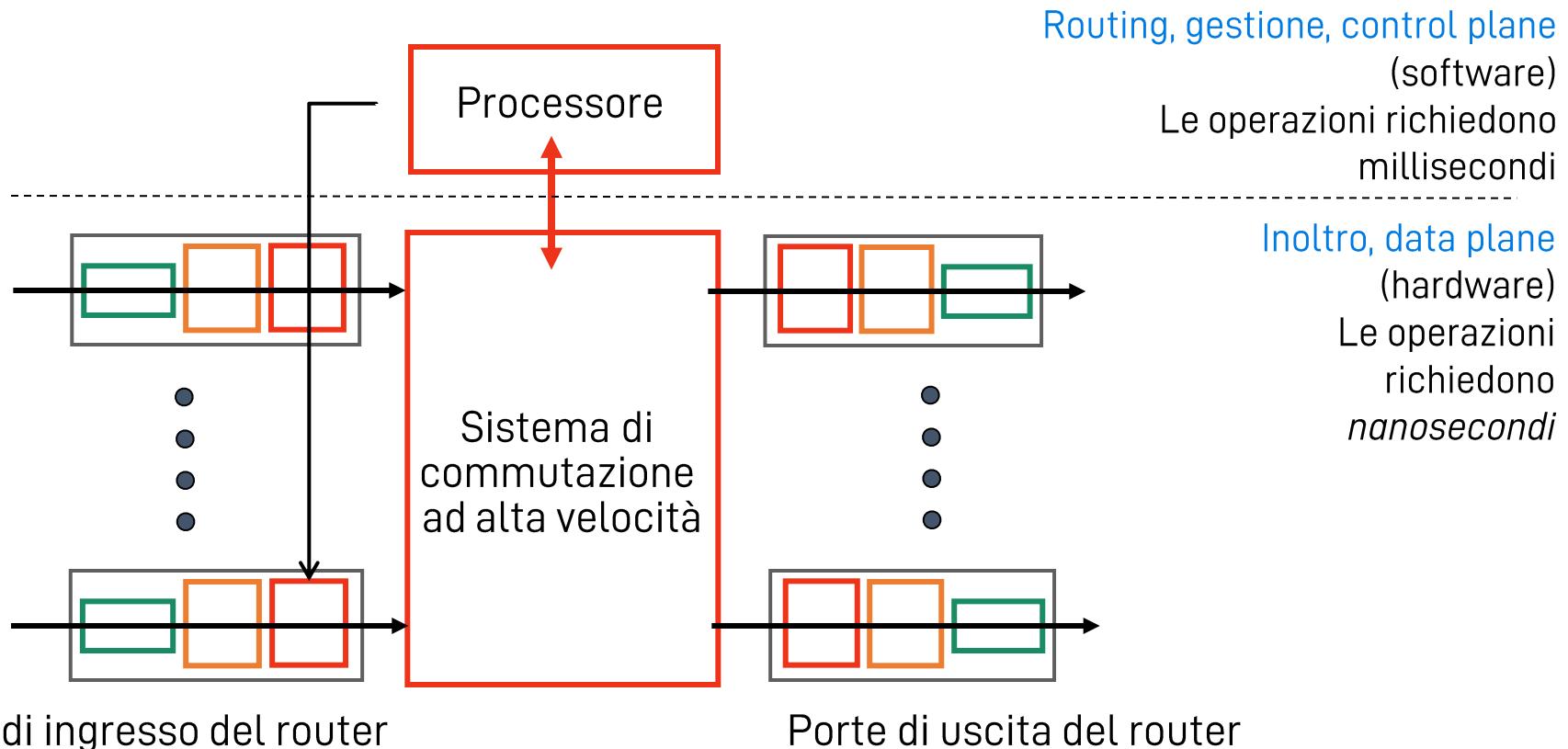


# Sommario

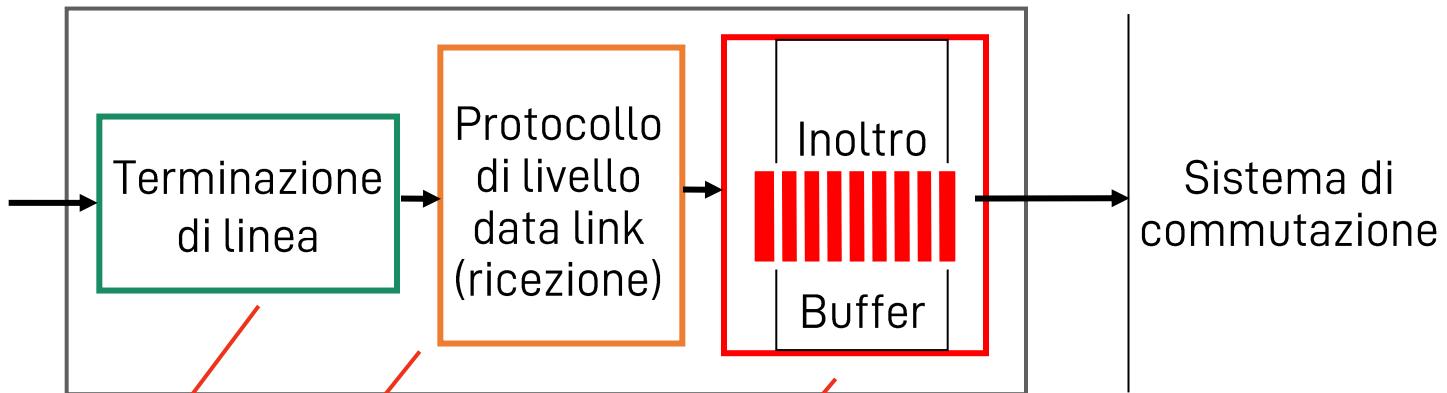
- Visione d'insieme
  - ❖ Data plane e control plane
- Come è fatto un router
- Il protocollo IP
  - ❖ Formato datagrammi
  - ❖ Frammentazione
  - ❖ Indirizzamento e NAT
  - ❖ Indirizzamento classless
  - ❖ Tipi di indirizzi
  - ❖ Address Resolution Protocol (ARP)
  - ❖ Internet Control Message Protocol (ICMP)
- Il protocollo IP (segue)
  - ❖ Dynamic Host Configuration Protocol (DHCP)
  - ❖ Il viaggio di un pacchetto
  - ❖ IPv6
- Protocolli di instradamento
  - ❖ Link state: Dijkstra
  - ❖ Internet, AS, OSPF
  - ❖ Distance vector: Bellman-Ford
  - ❖ RIP
- BGP

# Architetture dei router: panoramica

- Visione di alto livello dell'architettura di un router



# Funzioni delle porte di ingresso



Livello fisico:

Ricezione dei singoli bit

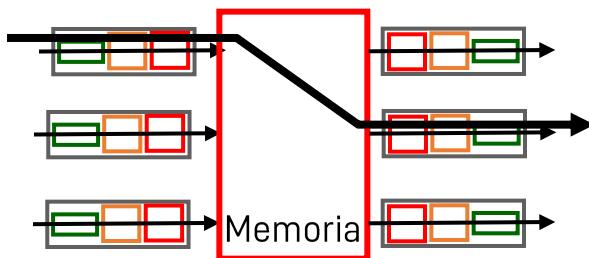
Livello data link:  
ad es., Ethernet

Sistema di commutazione decentralizzato:

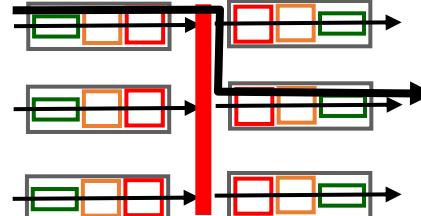
- Utilizzando i valori nei campi dell'header livello 3, trova la porta di uscita corretta usando la tabella di inoltro nella porta di ingresso ("match plus action")
- *Obiettivo:* non introdurre ritardi ulteriori, prendendo una decisione alla stessa velocità con cui la porta di ingresso riceve i dati
- Accodamento nel buffer: se i datagrammi arrivano più in fretta del tasso di smistamento del commutatore

# Sistemi di commutazione

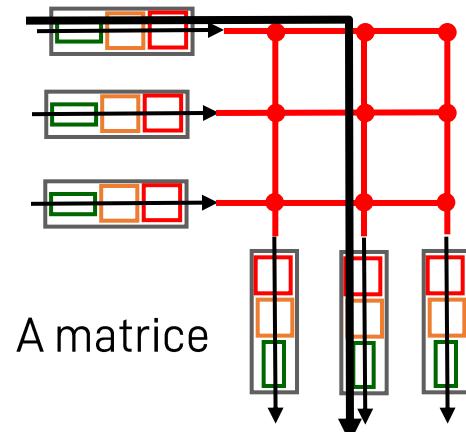
- Trasferiscono i pacchetti dall'ingresso all'uscita appropriata
- Tasso di commutazione: frequenza alla quale i pacchetti vengono trasferiti dagli ingressi alle uscite
  - ❖ Spesso misurato come multiplo della velocità di comunicazione
  - ❖ Con  $N$  ingressi, vorremmo una commutazione  $N$  volte più veloce della comunicazione
- Tre strutture tipo



A memoria



A bus

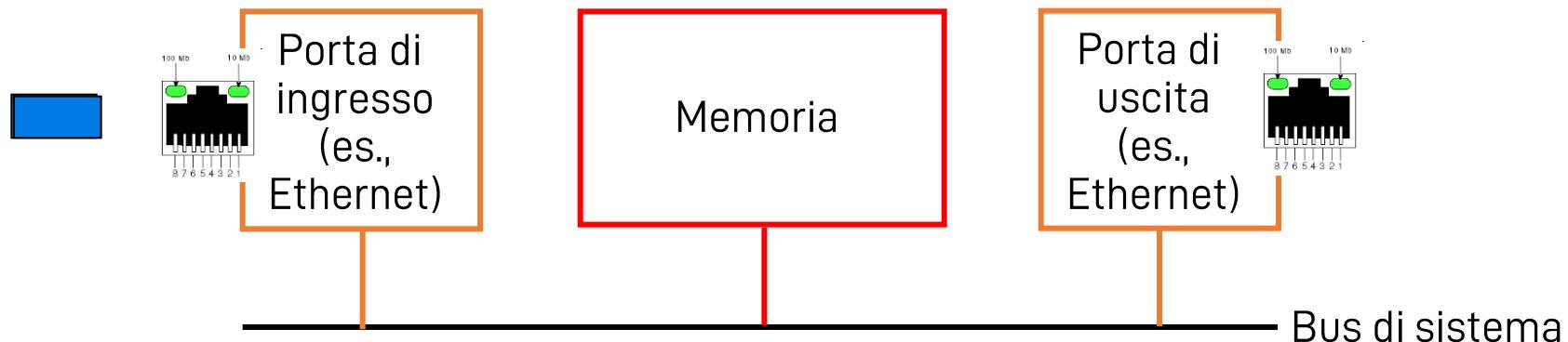


A matrice

# Commutazione a memoria

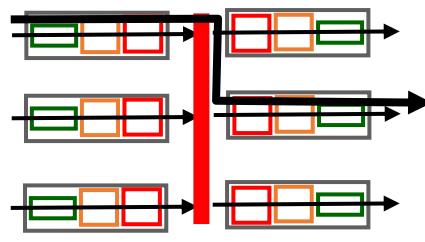
## □ Prime generazioni di router

- ❖ Di fatto, computer tradizionali dove la commutazione è controllata dalla CPU
- ❖ Pacchetti copiati nella memoria del computer
- ❖ Velocità di commutazione limitata dalla banda dati della memoria (servono 2 accessi al bus di sistema per datagramma)



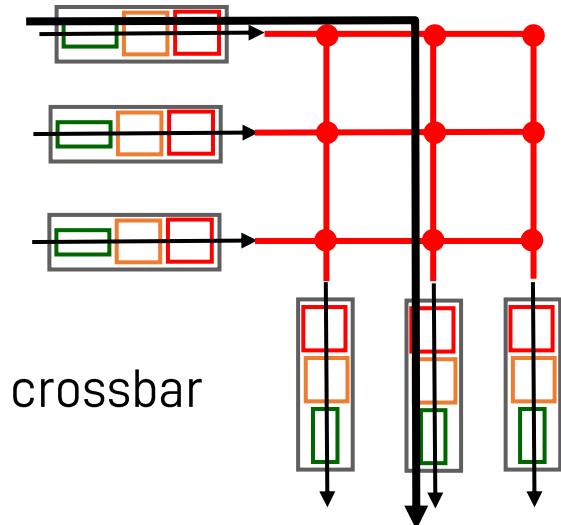
# Commutazione a bus

- Si usa un bus dati condiviso interno al router per trasmettere i datagrammi dagli ingressi alle uscite
- Contesa del bus: la velocità di commutazione è limitata dalla banda del bus (cioè dalla velocità di trasferimento dati sul bus)
- 32 Gbit/s bus, Cisco 5600: sufficiente per router di accesso e router interni alle reti aziendali



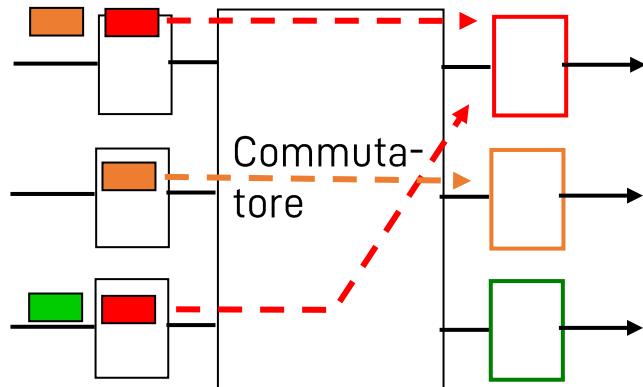
# Commutazione a matrice

- Matrice di punti di interconnessione tra linea di ingresso e linea di uscita
  - ❖ Ispirata ai primi commutatori telefonici elettromeccanici
  - ❖ Reti Banyan, crossbar, ...
  - ❖ Usate anche per connettere i processori nei sistemi di calcolo multiprocessore
- Supera i limiti di velocità della commutazione a bus
- Design avanzato: frammentazione dei datagrammi in celle di lunghezza fissa, che transitano poi attraverso la matrice
- Cisco 12000: commuta 60 Gbit/s attraverso la matrice di interconnessione



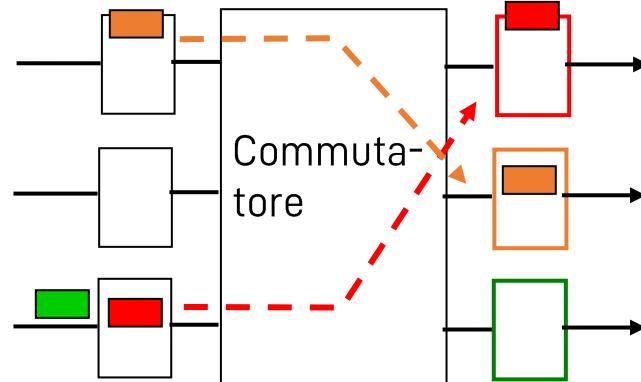
# Accodamento alle porte di ingresso

- Un commutatore più lento della velocità complessiva delle porte di ingresso causa accodamenti agli ingressi
  - ❖ Ritardi, ma anche perdite dovute all'overflow dei buffer
- Blocco "head-of-line" (HOL): un datagramma accodato al primo posto della coda impedisce a quelli successivi di avanzare



## Contesa sull'uscita:

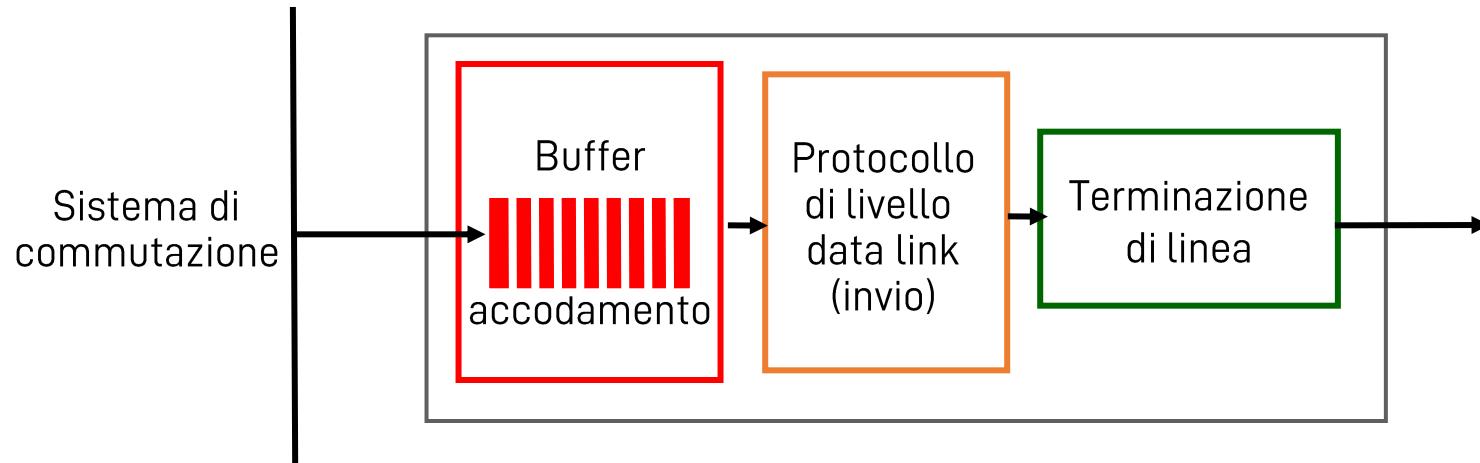
Solo un pacchetto rosso per volta può essere trasferito all'uscita rossa:  
quello sotto rimane bloccato



Il pacchetto verde  
subisce HOL blocking

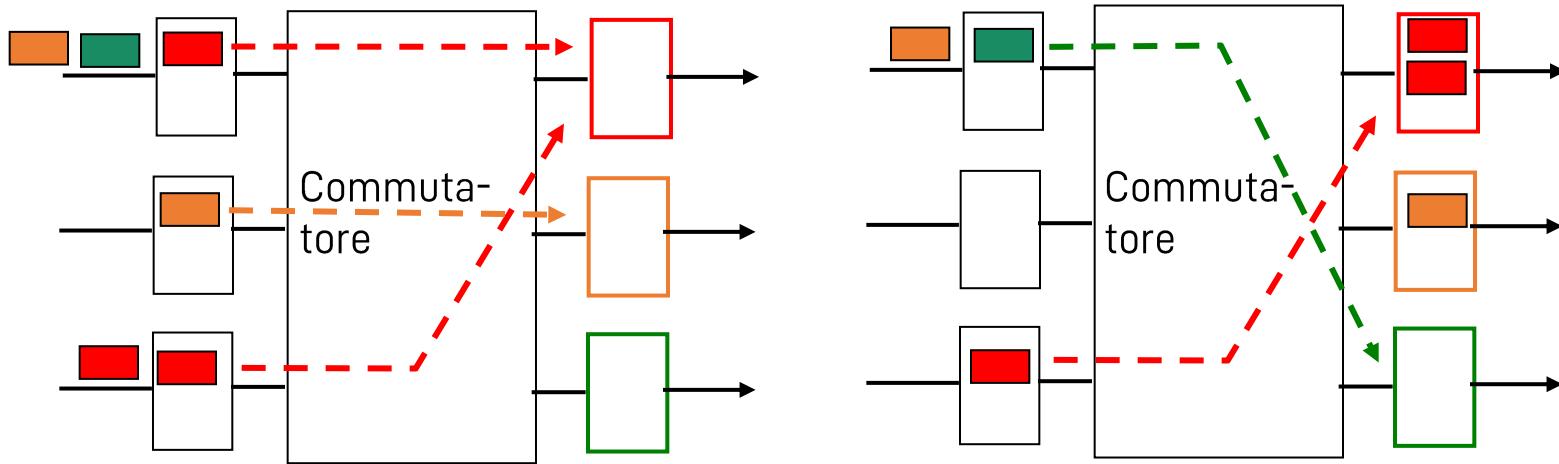
# Porte di uscita

- In questo caso, se i datagrammi arrivano dal commutatore più in fretta di quanto le porte di uscita possano smaltirli:
  - ❖ I datagrammi potrebbero essere scartati se le memorie si riempiono
- Politica di scheduling: quale datagramma deve essere servito?
  - ❖ FIFO? Priority scheduling?
  - ❖ Network neutrality?



# Accodamento alle porte di uscita

- Come prima: se i pacchetti arrivano troppo in fretta, si accodano
- Ritardi di accodamento, o perdite per buffer overflow



Al tempo  $t$ ,  
i datagrammi  
transitano dagli  
ingressi alle uscite

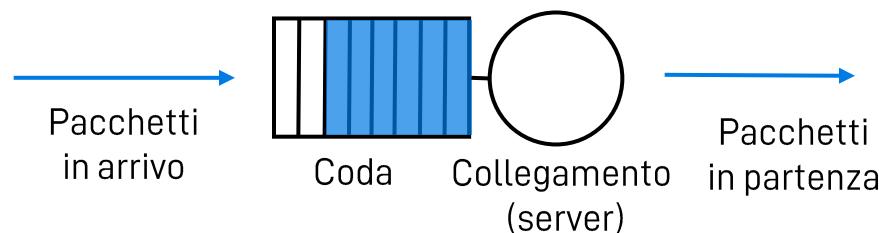
Successivamente

# Quanta memoria serve nei buffer?

- Regola empirica nella RFC 3439:
  - ❖ La memoria richiesta è pari a un "tipico" RTT moltiplicato la velocità del link C
  - ❖ Supponiamo che RTT = 250 [ms] e che C = 10 Gbit/s: servirebbero 2.5 Gbit di buffer!!
- Raccomandazioni più recenti: con N flussi, la quantità di memoria richiesta è  $\frac{RTT \times C}{\sqrt{N}}$

# Meccanismi di scheduling

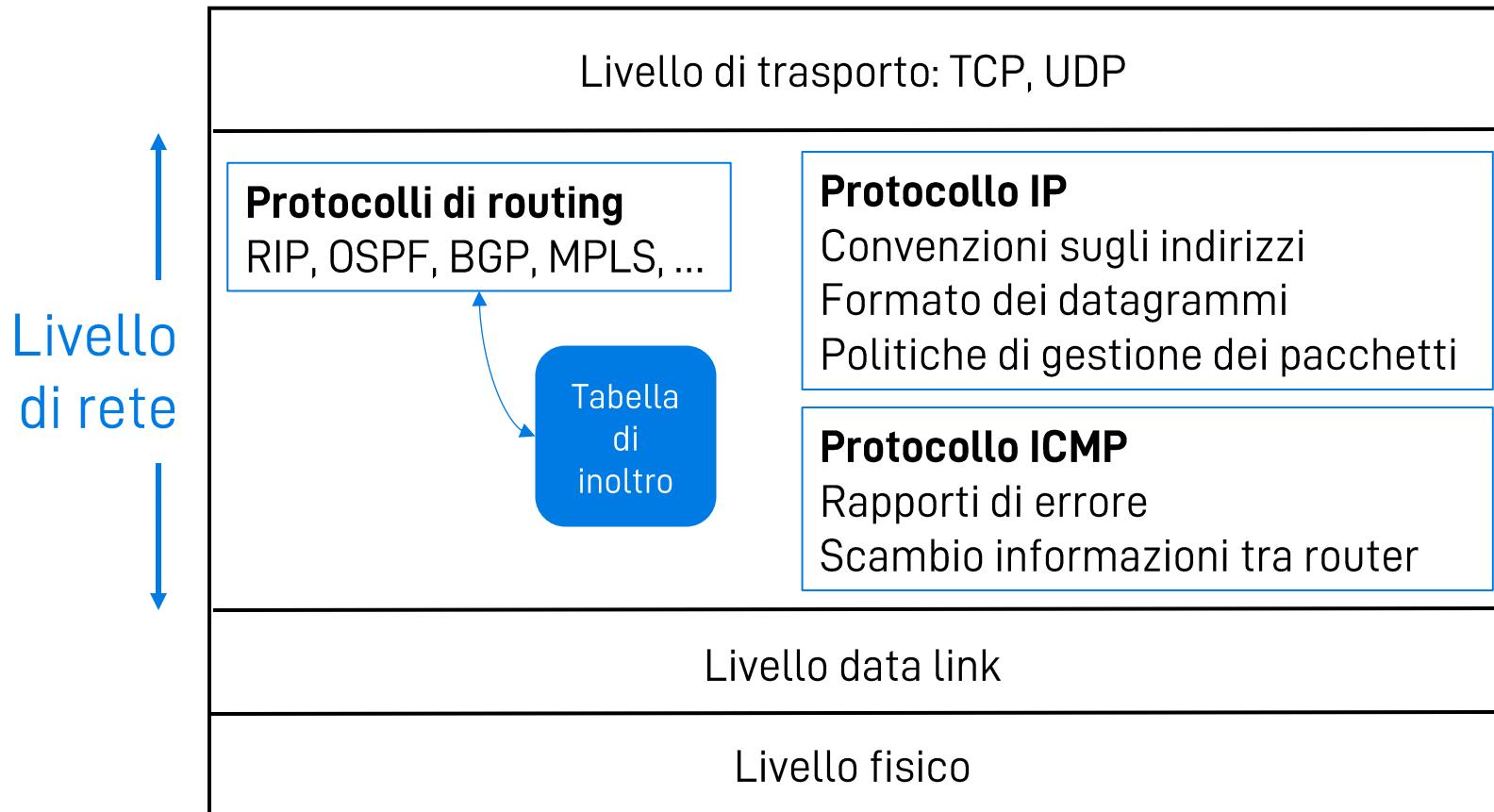
- Scheduling: scelta di quale pacchetto trasmettere su un link
- Scheduling FIFO (first-in-first-out): inviare i pacchetti in ordine di arrivo nella coda
  - ❖ Esempio di sistema FIFO?
  - ❖ Politica di scarto: se la coda è piena, quali pacchetti scarto?
    - “Tail drop”: scarto i pacchetti in arrivo
    - “Priority drop”: scarto o cancello basandomi su un livello di priorità
    - “Random drop”: scarto o cancello casualmente



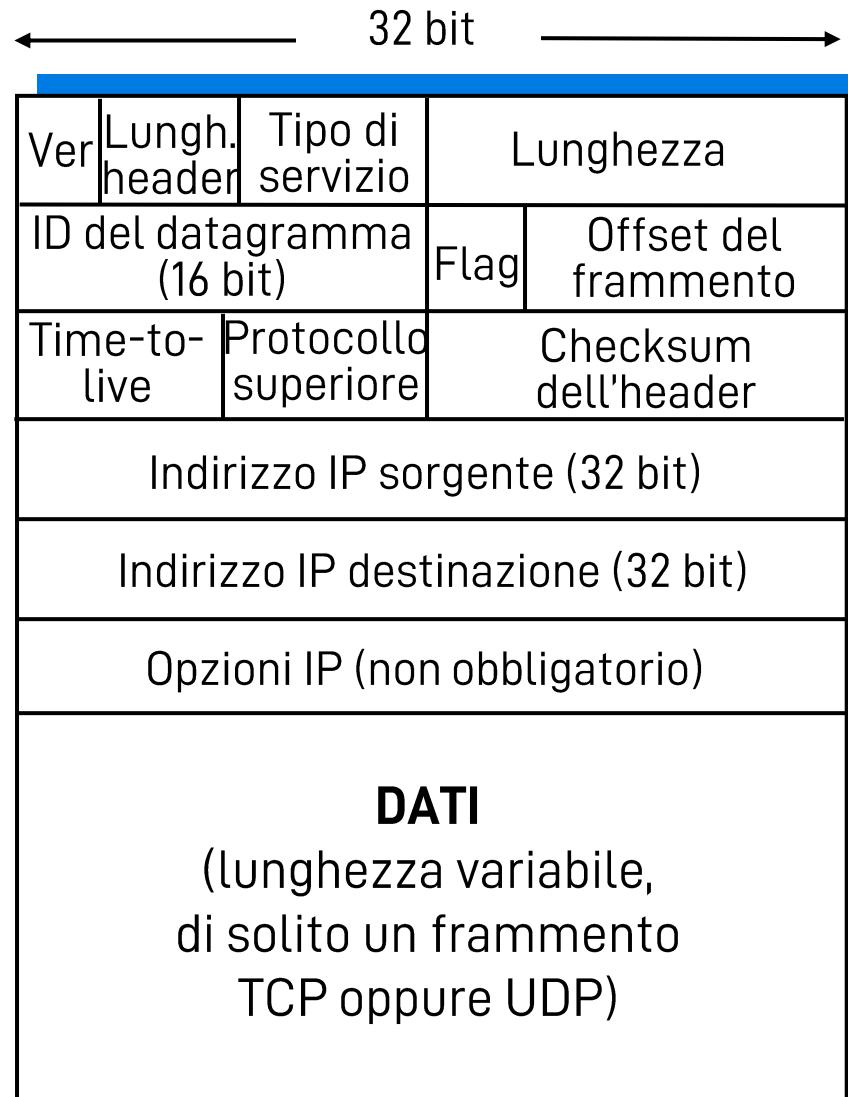
# Sommario

- Visione d'insieme
  - ❖ Data plane e control plane
- Come è fatto un router
- Il protocollo IP
  - ❖ Formato datagrammi
  - ❖ Frammentazione
  - ❖ Indirizzamento e NAT
  - ❖ Indirizzamento classless
  - ❖ Tipi di indirizzi
  - ❖ Address Resolution Protocol (ARP)
  - ❖ Internet Control Message Protocol (ICMP)
- Il protocollo IP (segue)
  - ❖ Dynamic Host Configuration Protocol (DHCP)
  - ❖ Il viaggio di un pacchetto
  - ❖ IPv6
- Protocolli di instradamento
  - ❖ Link state: Dijkstra
  - ❖ Internet, AS, OSPF
  - ❖ Distance vector: Bellman-Ford
  - ❖ RIP
- BGP

# Il livello di rete in Internet



# Formato del datagramma IP (IPv4)



# Campi del datagramma IP

- **VER** (4 bit): numero di versione IP (solo 4 o 6)
- **LUNGHEZZA HEADER** (4 bit): indica il numero di parole di 32 bit nell'header (=5 se non ci sono opzioni)
- **TIPO DI SERVIZIO** (Type of service, ToS) (8 bit): classe di servizio del datagramma
  - ❖ Potenzialmente usato per funzioni chiamate "DiffServ" e "Explicit Congestion Notification" (ECN)
  - ❖ Raramente usato in pratica
- **LUNGHEZZA TOTALE** (16 bit): numero totale di byte nel datagramma, includendo sia l'header sia i dati

# Campi del datagramma IP

- **IDENTIFICAZIONE (ID) DEL DATAGRAMMA** (16 bit): numero (di solito sequenziale) assegnato al datagramma
  - ❖ Usato per raccogliere eventuali frammenti multipli e riassemblare datagramma complessivo
- **FLAG** (3 bit): campo i cui bit specificano se il datagramma è un frammento di un datagramma più lungo
  - ❖ Se è così, dicono anche se questo è l'ultimo frammento o no
- **OFFSET DEL FRAMMENTO** (13 bit): indica in quale punto del datagramma originale va inserito questo frammento
  - ❖ espresso in multipli di 8 Byte

# Campi del datagramma IP

- **TIME TO LIVE (TTL)** (8 bit): intero inizializzato dal mittente che:
  - ❖ Viene ridotto di 1 da ogni router per cui passa il datagramma
  - ❖ Se raggiunge il valore zero, il router scarta il datagramma e invia un messaggio di notifica al mittente
- **PROTOCOLLO SUPERIORE** (8 bit): specifica quale protocollo di livello superior aspettarsi all'inizio dei dati encapsulati nel datagramma (es. 6 = TCP, 17 = UDP)
- **CHECKSUM DELL'HEADER** (16 bit): come in UDP: complemento a 1 della somma con riporto di tutte le parole di 16 bit dell'header
- **INDIRIZZO IP SORGENTE** (32 bit): indirizzo IP del mittente INIZIALE
- **INDIRIZZO IP DESTINAZIONE** (32 bit): indirizzo IP della destinazione FINALE

# Campi del datagramma IP

- **OPZIONI IP:** in alcuni casi, usato per controllare l'elaborazione e instradamento dei datagrammi
  - ❖ Nella maggioranza dei casi, il campo è vuoto
- **PADDING:** bit a zero aggiunto se le opzioni non terminano ad un multiplo di 32 bit, in modo che l'header sia un multiplo di 32 bit

# Sommario

- Visione d'insieme
  - ❖ Data plane e control plane
- Come è fatto un router
- Il protocollo IP
  - ❖ Formato datagrammi
  - ❖ **Frammentazione**
  - ❖ Indirizzamento e NAT
  - ❖ Indirizzamento classless
  - ❖ Tipi di indirizzi
  - ❖ Address Resolution Protocol (ARP)
  - ❖ Internet Control Message Protocol (ICMP)
- Il protocollo IP (segue)
  - ❖ Dynamic Host Configuration Protocol (DHCP)
  - ❖ Il viaggio di un pacchetto
  - ❖ IPv6
- Protocolli di instradamento
  - ❖ Link state: Dijkstra
  - ❖ Internet, AS, OSPF
  - ❖ Distance vector: Bellman-Ford
  - ❖ RIP
- BGP

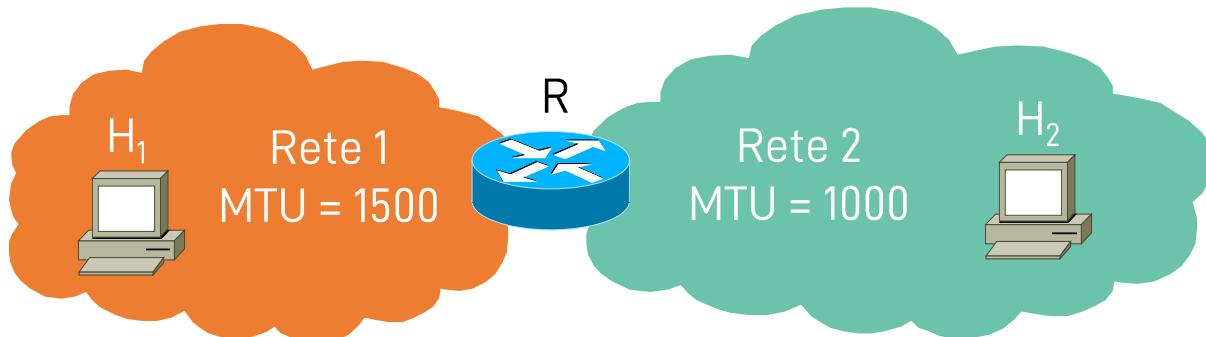
# MTU e frammentazione datagrammi

- Ciascun tipo di hardware specifica un limite di dati che una trasmissione può trasportare (Maximum Transmission Unit, MTU)
  - ❖ Ciascun datagramma deve contenere al più un numero di Byte pari alla MTU, se no l'hardware non può incapsularlo in un frame (livello 2)
- Internet è molto eterogenea, diverse reti specificano diverse MTU
- Un router potrebbe interfacciarsi a reti con MTU diverse
  - ❖ Un datagramma su una di queste reti potrebbe essere troppo grande per inviarlo su una delle altre reti

Protocollo	MTU [byte]
Hyperchannel	65535
Token ring (16 Mbps)	17914
Token ring (4 Mbps)	4464
FDDI	4352
Ethernet	1500
WLAN (802.11 WiFi)	2304
X.25	576
PPP	Variable

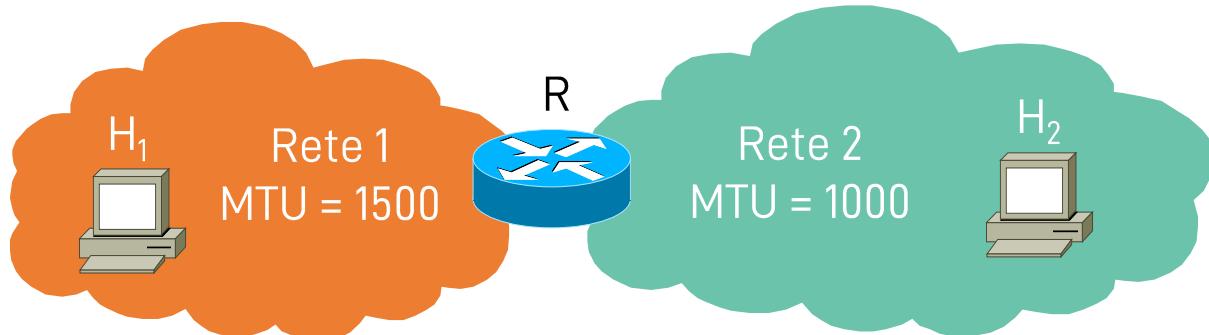
# MTU e frammentazione datagrammi

- Esempio: un router interconnecte due reti con MTU di 1500 e 1000 Byte, rispettivamente
  - Se l'host  $H_1$  invia un datagramma di 1500 Byte all'host  $H_2$ 
    - ❖ Il router R non potrà trasmetterlo direttamente sulla rete 2
  - Quindi, R frammenta il datagramma e invia ogni frammento indipendentemente dagli altri
    - ❖ I frammenti hanno lo stesso formato di un datagramma normale
    - ❖ I flag nell'header indicano se un datagramma è un frammento o no
    - ❖ Il campo "Offset del frammento" specifica come concatenarli



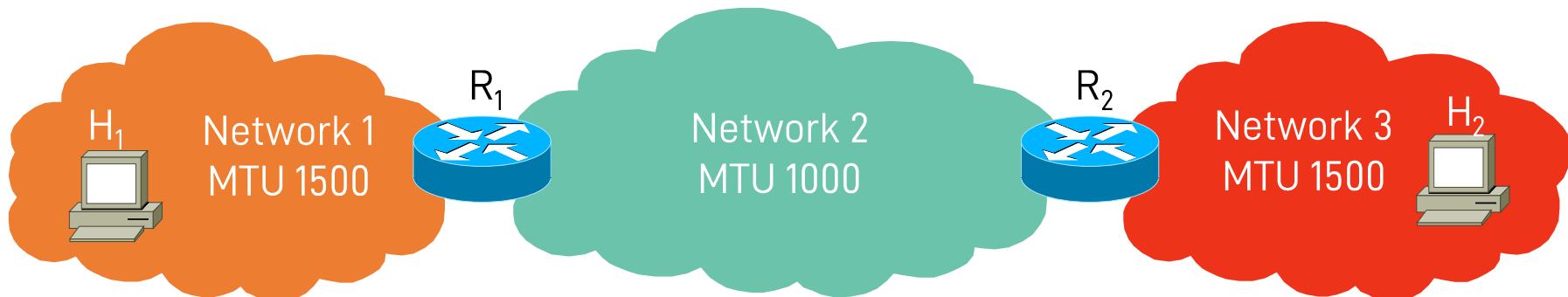
# MTU e frammentazione datagrammi

- Conoscendo l'MTU e la dimensione dell'header, il router calcola quanti frammenti servono, e quanti Byte di dati contiene ciascuno
- Quando crea i frammenti, il router
  - ❖ Copia i campi necessari dall'header del frammento originale
  - ❖ Trasmette tutti i frammenti in sequenza (con identificatori diversi)
- Flag usati: **0 | D | M**
  - ❖ D: "do not fragment" (se servisse frammentare, scarta il pacchetto)
    - Ricevitore che non può gestire la frammentazione, o rilevamento MTU
  - ❖ M: "more fragments" (=1 per ogni frammento)
    - Eccetto l'ultimo: M=0, ma fragment offset > 0



# Chi riassembra il datagramma?

- Esempio: datagramma di 1500 Byte da  $H_1$  to  $H_2$ 
  - ❖  $R_1$  divide il datagramma in due frammenti e li invia a  $R_2$
- $R_2$  NON riassembra i frammenti
  - ❖ Non è la destinazione finale: li inoltra come pacchetti qualsiasi (**stateless**)
- La destinazione finale  $H_2$  raccoglie e riassembra i frammenti
  - ❖ Vantaggi
    - I frammenti possono seguire percorsi diversi
    - $R_2$  non deve ricevere tutti i frammenti prima di inoltrarli



# Conseguenze della perdita di frammenti

- Un datagramma frammentato non può essere ricomposto finché non arrivano tutti i frammenti
- Il ricevitore memorizza i frammenti finché non li riceve tutti
  - ❖ Ma non può memorizzarli per un tempo arbitrariamente lungo
- Il protocollo IP specifica il tempo massimo da attendere dopo la ricezione del primo frammento
  - ❖ I ritardi di consegna potrebbero far eccedere questo tempo

# In pratica...

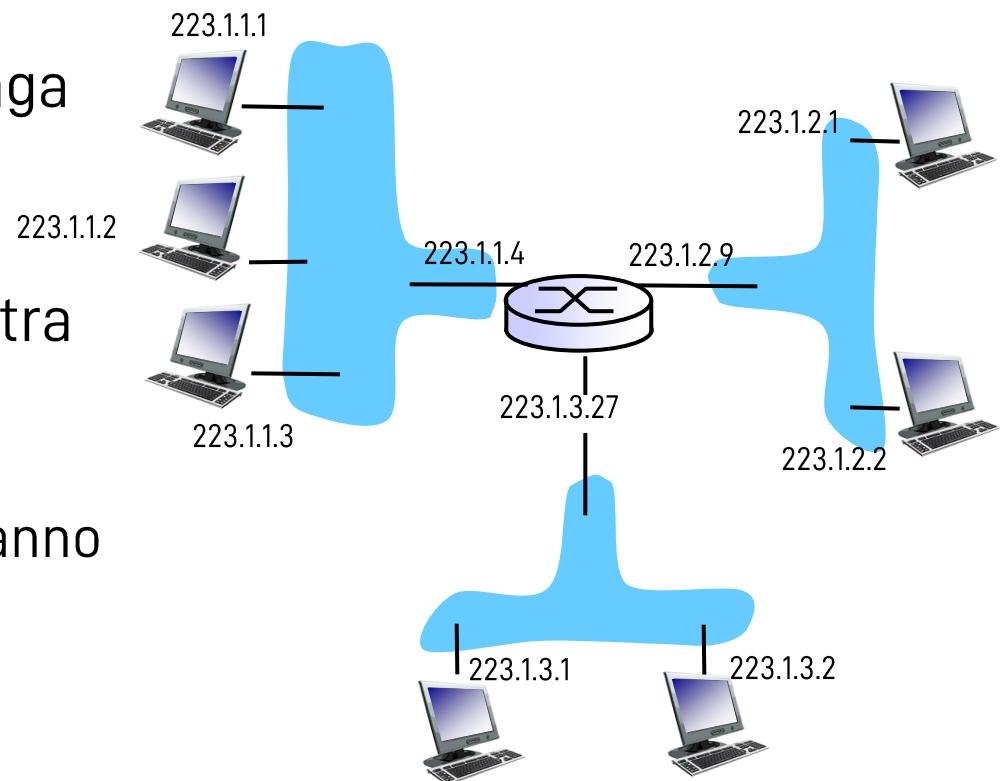
- La frammentazione a livello IP è praticamente disabilitata in Internet
  - ❖ Alcuni router nemmeno la implementano
  - ❖ Se un datagramma è troppo grande → scartato
  - ❖ IPv6 non consente più la frammentazione
- Molte ragioni, per prima cosa la sicurezza
  - ❖ I firewall richiedono l'ispezione dell'header TCP/UDP
  - ❖ Attacco "overlapping fragments"
    - Inganna firewall "stateless" consentendo a traffico illecito di passare
    - Può essere risolto con firewall "stateful", che però hanno vincoli di memoria, complessità e costi maggiori
  - ❖ Attacchi di riempimento memoria (tipo DDoS) ottenuti omettendo di proposito alcuni frammenti
  - ❖ Implementazioni errate del codice di assemblaggio dei frammenti (es. gestione errata offset negativi) che mandavano in crash il software del router e permettevano l'esecuzione di comandi arbitrari

# Sommario

- Visione d'insieme
  - ❖ Data plane e control plane
- Come è fatto un router
- Il protocollo IP
  - ❖ Formato datagrammi
  - ❖ Frammentazione
  - ❖ Indirizzamento e NAT
  - ❖ Indirizzamento classless
  - ❖ Tipi di indirizzi
  - ❖ Address Resolution Protocol (ARP)
  - ❖ Internet Control Message Protocol (ICMP)
- Il protocollo IP (segue)
  - ❖ Dynamic Host Configuration Protocol (DHCP)
  - ❖ Il viaggio di un pacchetto
  - ❖ IPv6
- Protocolli di instradamento
  - ❖ Link state: Dijkstra
  - ❖ Internet, AS, OSPF
  - ❖ Distance vector: Bellman-Ford
  - ❖ RIP
- BGP

# Indirizzi IP: introduzione

- Un indirizzo IP è una stringa di 32 bit **associata ad un'interfaccia di rete**
- Interfaccia: connessione tra l'host o il router e il collegamento fisico
  - ❖ I router normalmente hanno diverse interfacce
  - ❖ Gli host ne hanno 1-2
    - Ethernet, WiFi
- Ogni **interfaccia** ha un indirizzo IP differente



223.1.1.1 = 11011111 00000001 00000001 00000001

                  223           1           1           1

# Indirizzi IP

- Gli host e i router devono usare le stesse convenzioni di indirizzo
- Ogni indirizzo IP pubblicamente raggiungibile deve essere unico
- Ogni interfaccia ha (almeno) un indirizzo IP
- Quando si invia un pacchetto su Internet il software che implementa i protocolli di rete lato mittente deve specificare
  - ❖ Il proprio indirizzo IP (32 bit), cioè il source address
  - ❖ L'indirizzo IP del ricevitore (32 bit), cioè il destination address
- I router prendono decisioni di inoltro e instradamento basandosi *unicamente* sul destination address

# Notazione per gli indirizzi IP

- "Dotted decimal notation"
  - ❖ Ogni sezione da 8 bit (1 Byte) è espressa come un intero decimale
  - ❖ Solo numeri positivi
  - ❖ Separati da punti
- Chiaramente:
  - ❖ 00000000 → 0
  - ❖ 00000001 → 1
  - ❖ 00000010 → 2
  - ❖ ..... ....
  - ❖ 11111111 → 255
- Gamma di indirizzi disponibili:
  - ❖ Da 0.0.0.0 fino a 255.255.255.255

# Gerarchia degli indirizzi IP

- Gli indirizzi IP sono generalmente divisi in due parti
  - ❖ Un **prefisso**: identifica la rete alla quale l'host è allacciato (network ID, o **NetID**)
    - Ogni rete in Internet è identificata da un numero unico al mondo
    - Un NetID specifica una Local Area Network, o LAN
  - ❖ Un **suffisso**: identifica un'interfaccia di rete allacciata a quella rete (ID dell'host, o **HostID**)
    - Ogni interfaccia di rete ha un indirizzo IP unico su quella rete
- Il modo di assegnare gli indirizzi IP garantisce che:
  - ❖ Ogni host riceva un indirizzo univoco
  - ❖ Gli ID di rete (i prefissi) siano coordinati a livello globale
  - ❖ I suffissi possano essere assegnati localmente senza bisogno di coordinazione globale

# Come distinguere NetID e HostID?

- In passato: assegnazioni statiche (ora obsolete)
  - ❖ Diverse "classi" di indirizzi, con diverse lunghezze dei prefissi:
    - Classe A (primo bit = 0): 7 bit per il NetID, 24 bit per l'HostID
    - Classe B (primi bit = 10): 14 bit per il NetID, 16 bit per l'HostID
    - Classe C (primi bit = 110): 21 bit per il NetID, 8 bit per l'HostID
  - ❖ Scelta infelice delle dimensioni delle reti che ne risultavano
    - Classe A: 128 reti, ciascuna con 16777216 host
    - Classe B: 16384 reti, ciascuna con 65536 host
    - Classe C: 2097152 reti, ciascuna con 256 host
    - **Classe D (primi bit = 1110): riservata per 268,435,456 indirizzi multicast**
    - **Classe E (primi bit = 1111): riservata, 268,435,456 indirizzi non assegnati**
- Tutti volevano classi A e B per poter crescere se necessario
  - ❖ Lo spazio degli indirizzi fu presto esaurito

# Classful addressing: netID vs hostID

	bits	0	1	2	3	4	8	16	24	31
<b>Class A</b>		0								
<b>Class B</b>		1	0							
<b>Class C</b>		1	1	0						
<b>Class D</b>		1	1	1	0					
<b>Class E</b>		1	1	1	1					

# Autorità per assegnazione indirizzi

- ❑ Internet Corporation for Assigned Names and Numbers (ICANN)
  - ❖ Creata per gestire l'assegnazione di indirizzi ed arbitrare eventuali dispute tra molteplici "pretendenti"
- ❑ ICANN non assegna i prefissi direttamente
  - ❖ Invece, autorizza diverse entità dette "registrar" a farlo
- ❑ I registrar consentono agli Internet Service Provider (ISP) di accaparrarsi "blocchi" di indirizzi
  - ❖ Gli ISP provvederanno poi ad assegnare sotto-blocchi di questi indirizzi ai loro clienti
- ❑ Quindi, per ottenere un prefisso di rete per sè, un'azienda solitamente contatta un ISP e se ne fa assegnare uno

# Sommario

- Visione d'insieme
  - ❖ Data plane e control plane
- Come è fatto un router
- Il protocollo IP
  - ❖ Formato datagrammi
  - ❖ Frammentazione
  - ❖ Indirizzamento e NAT
  - ❖ **Indirizzamento classless**
  - ❖ Tipi di indirizzi
  - ❖ Address Resolution Protocol (ARP)
  - ❖ Internet Control Message Protocol (ICMP)
- Il protocollo IP (segue)
  - ❖ Dynamic Host Configuration Protocol (DHCP)
  - ❖ Il viaggio di un pacchetto
  - ❖ IPv6
- Protocolli di instradamento
  - ❖ Link state: Dijkstra
  - ❖ Internet, AS, OSPF
  - ❖ Distance vector: Bellman-Ford
  - ❖ RIP
- BGP

# Indirizzamento classless

- Suddivisione tra prefisso e suffisso completamente arbitraria
- Se un cliente di un ISP richiedesse un prefisso di rete per una rete che deve contenere 57 host
  - ❖ Con le classi, servirebbe una classe "C" = 256 indirizzi (197 di troppo)
    - Non 199, poi vediamo perché
  - ❖ Per 57 host, bastano 6 bit ( $2^6 = 64 \geq 57$ )
- Eliminando le classi ("classless", appunto) l'ISP può quindi assegnare
  - ❖ Un prefisso di 26 bit, che identifica univocamente la rete nel mondo
  - ❖ Un suffisso di 6 bit, sufficiente per tutti gli host

# Indirizzamento classless

- Un registrar alloca un prefisso a un ISP, che può gestirlo come vuole
- Es.: supponiamo un ISP avesse un prefisso di classe C
  - ❖ Con l'assegnazione "classful", dovrebbe dare tutti gli indirizzi allo stesso cliente
  - ❖ Con l'indirizzamento classless, può suddividere ulteriormente lo spazio di indirizzi a disposizione
  - ❖ Come? Allungando il prefisso.
- Es.: l'ISP ha il blocco di indirizzi 193.185.15.0 - 193.185.15.255
  - ❖ Da **11000001.10111001.00001111.00000000**
  - ❖ A **11000001.10111001.00001111.11111111**
- Per poterli ripartire tra più clienti, l'ISP genera 4 prefissi più lunghi, es.
  - ❖ Prefisso 1: **11000001.10111001.00001111.00** xxxxxxx
  - ❖ Prefisso 2: **11000001.10111001.00001111.01** xxxxxxx
  - ❖ Prefisso 3: **11000001.10111001.00001111.10** xxxxxxx
  - ❖ Prefisso 4: **11000001.10111001.00001111.11** xxxxxxx
  - ❖ Ogni rete può contenere 62 host
    - Il suffisso con tutti 0 riporta all'indirizzo di rete e quindi non è un indirizzo valido, il suffisso con tutti 1 è un indirizzo speciale detto di "broadcast", ora lo vediamo)
- 4 clienti invece di 1!

# Maschere di rete

- Il problema è ora conoscere il limite tra prefisso e suffisso
- Soluzione: si memorizzano 32 bit dove gli unici bit a 1 sono quelli del prefisso
- In IP, questa informazione è chiamata "subnet mask"
- Perché una "maschera"?
  - ❖ Ai router serve solo il prefisso per prendere decisioni di inoltro
  - ❖ Una maschera permette di estrarre il prefisso da un indirizzo con un AND logico bit-a-bit (efficiente)
- Rete di destinazione in una tabella di inoltro: lista di indirizzi di rete ( $N_i$ ) e delle maschere corrispondenti ( $M_i$ )
  - ❖ Per mandare pacchetti a D, si usa l'interfaccia per cui  $D \& M_i == N_i$

Rete di destinazione	Interfaccia di uscita
$N_1 / M_1$	Interfaccia 1
$N_2 / M_2$	Interfaccia 2
$N_3 / M_3$	Interfaccia 3
...	...

# Subnet mask example

- Prendiamo ad esempio questo prefisso di rete:
  - ❖ 10000000 00001010 00000000 00000000 = 128.10.0.0
- Con questa maschera:
  - ❖ 11111111 11111111 00000000 00000000 = 255.255.0.0
- E questo indirizzo di destinazione
  - ❖ 10000000 00001010 00000010 00000011 = 128.10.2.3
- (tutti di 32 bit)
- L'AND bit-a-bit (&) tra maschera e indirizzo di destinazione estrae i primi 16 bit, cioè proprio il prefisso
  - ❖ 10000000 00001010 00000000 00000000 = 128.10.0.0

# Notazione CIDR

- Classless Inter-Domain Routing (CIDR)
- Consideriamo ancora prefissi di 26 bit, e 6 bit per l'HostID
  - ❖ Nella maschera ci sono 26 bit a 1, seguiti da 6 zeri
  - ❖ Notazione binaria: **11111111.11111111.11111111.11000000**
  - ❖ In notazione decimale: 255.255.255.192
- Con la notazione CIDR, possiamo sintetizzare il tutto scrivendo:  
**ddd.ddd.ddd.ddd/m**
  - ❖ Solita notazione decimale + il /m, dove m indica il numero di bit uguali a 1 nella maschera
- Esempio di indirizzo: **193.185.15.199/26**

# Un altro esempio di CIDR

- Un ISP possiede il blocco di indirizzi **128.211.0.0/16**
- L'ISP ha tre clienti
  - ❖ Il cliente 1 necessita di 12 indirizzi IP
  - ❖ Il cliente 2 necessita di 9 indirizzi IP
  - ❖ Il cliente 3 necessita di 6 indirizzi IP
- L'ISP può assegnare i seguenti prefissi
  - ❖ Cliente 1: **128.211.0.16/28**
  - ❖ Cliente 2: **128.211.0.32/28**
  - ❖ Cliente 3: **128.211.0.48/29**
  - ❖ I clienti 1 e 2 hanno prefissi diversi, ma della stessa lunghezza
  - ❖ Al cliente 3 servono meno indirizzi IP, quindi il prefisso è più lungo

# Altro esempio di CIDR con reti della stessa dimensione

- Un ISP possiede il blocco di indirizzi **128.211.0.0/16**
- L'ISP ha 4 client che necessitano di 12 indirizzi IP ciascuno
- L'ISP può decidere un prefisso di partenza
  - ❖ Es. come prima, **128.211.0.0/28**
- E partire da lì per assegnare i seguenti prefissi
  - ❖ Cliente 1: **128.211.0. 0/28**
  - ❖ Cliente 2: **128.211.0.16/28**
  - ❖ Cliente 3: **128.211.0.32/28**
  - ❖ Cliente 4: **128.211.0.48/28**
- Il numero 16 (la dimensione della rete) viene a volte chiamato "magic number" (sommendolo, si ottengono i vari prefissi)
  - ❖ Con la notazione dotted decimal, attenzione ad eventuali riporti!

# Indirizzi degli host con CIDR

- Quando un ISP assegna un blocco di indirizzi a un cliente, il cliente può assegnare gli indirizzi ai propri host
- Esempio del cliente 1 di prima: **128.211.0.16/28**
  - ❖ Il cliente può usare 4 bit per assegnare indirizzi agli host
- Svantaggio
  - ❖ Bisogna acquisire un po' di dimestichezza con i valori delle maschere in notazione decimale
  - ❖ Es. 255.255.255.X
    - **10000000** = 128 → /25                      **11000000** = 192 → /26
    - **11100000** = 224 → /27                      **11110000** = 240 → /28
    - **11111000** = 248 → /29                      **11111100** = 252 → /30
    - **11111110** = 254 → /31                      **11111111** = 255 → /32
  - ❖ **D**: le ultime hanno senso di esistere?

# Inoltro con CIDR

- Quando riceve un datagramma, un router deve decidere su quale interfaccia inoltrarlo
- L'interfaccia usata dipende dall'indirizzo IP di destinazione
- La corrispondenza viene identificata usando la netmask
- Consideriamo la tabella di inoltro qui sotto e l'IP destinazione **200.23.19.7**: calcoliamo l'AND bit-a-bit con una maschera avente 23 bit a **1** (/23):
  - ❖  $200.23.19.7 \text{ & } 255.255.254.0 \text{ (/23)} = 200.23.18.0 \rightarrow \text{eth1}$

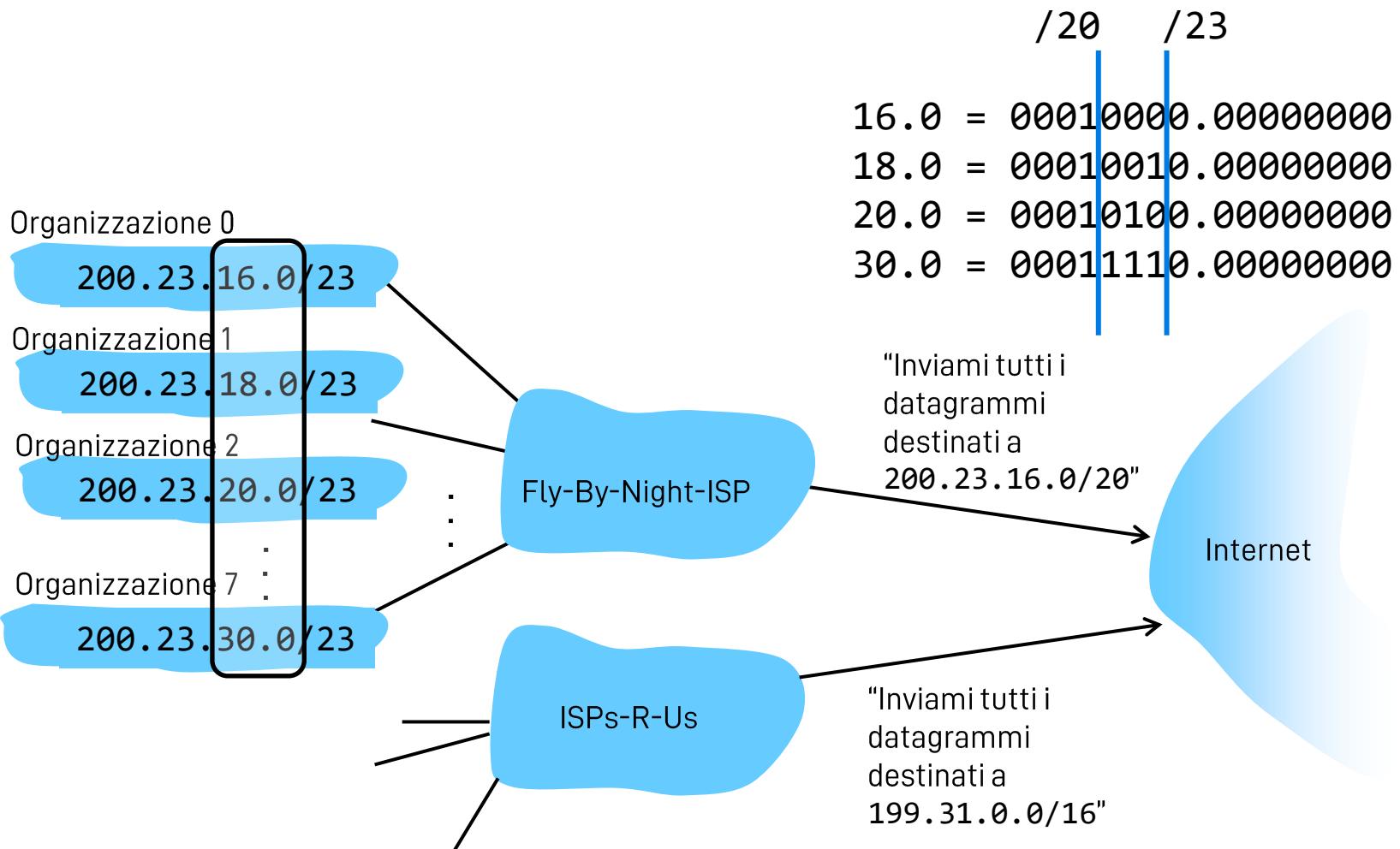
Destinazione	Interfaccia
200.23.16.0/23	eth0
200.23.18.0/23	eth1
200.23.20.0/23	eth2

# E se ci sono corrispondenze multiple?

- Consideriamo ancora la destinazione 200.23.19.7 e la nuova tabella sotto
  - ❖ La tabella contiene elementi con netmask diverse:
    - 200.23.19.7 & 255.255.240.0 (/20) = 200.23.16.0 → OK!
    - 200.23.19.7 & 255.255.254.0 (/23) = 200.23.18.0 → OK!
- Regola: **si usa il prefisso più lungo** ("longest prefix matching")
  - ❖ In questo caso, il datagramma viene inoltrato usando eth1

Destination	Interface
200.23.16.0/20	eth0
200.23.18.0/23	eth1

# Aggregazione dei percorsi

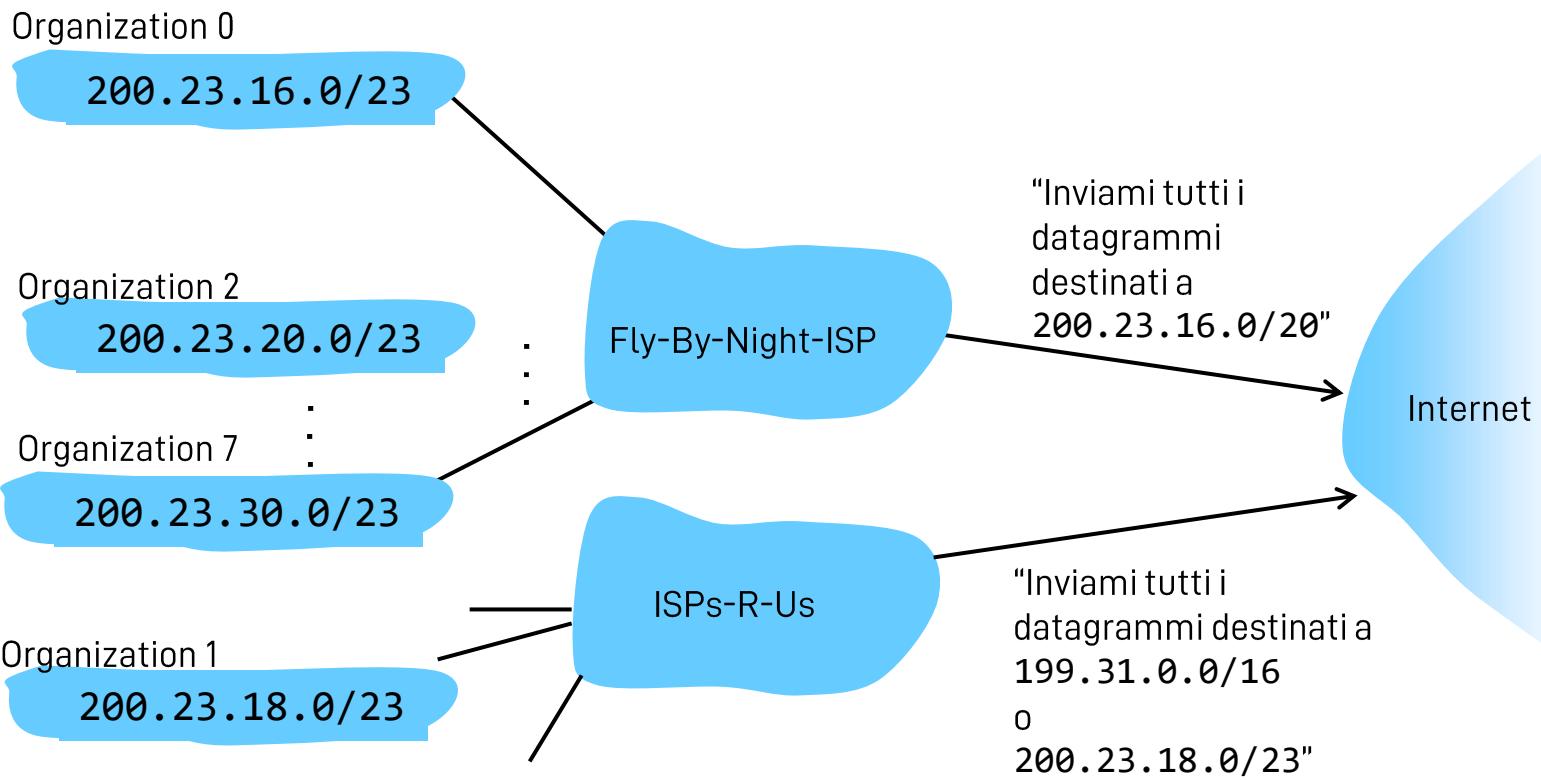


# Risolviamo ancora 200.23.19.7

- Un router da Internet deve inoltrare un datagramma per l'IP 200.23.19.7 o a "Fly-by-Night-ISP" oppure a "ISPs-R-Us"
- Qualcuno ha detto al router di Internet che esistono le due rotte
  - ❖ Vedremo più avanti come
- Matching con le rotte note
  - ❖  $200.23.19.7 \& 255.255.240.0 \text{ (/20)} = 200.23.16.0 \rightarrow \text{MATCH}$  con  $200.23.16.0/20$
  - ❖  $200.23.19.7 \& 255.255.0.0 \text{ (/16)} = 200.23.0.0 \rightarrow \text{NO MATCH}$  con  $199.31.0.0/16$
- Il router inoltra il pacchetto a Fly-by-Night-ISP
  - ❖ Sarà poi il router di Fly-by-Night-ISP a inoltrare il pacchetto alla sottorete corretta (nelle sue tabelle ci saranno maschere /23)

Destinazione	Interfaccia
200.23.16.0/20	Verso Fly-By-Night-ISP
199.31.0.0/16	Verso ISPs-R-Us

# Caso con percorsi più specifici



# Risolviamo nuovamente 200.23.19.7

- Stavolta la tabella di inoltro è diversa
- La ricerca delle corrispondenze è come segue
  - ❖ 200.23.19.7 & 255.255.240.0 (/20) = 200.23.16.0 → OK
  - ❖ 200.23.19.7 & 255.255.0.0 (/16) = 200.23.0.0 → NO MATCH
  - ❖ 200.23.19.7 & 255.255.254.0 (/23) = 200.23.18.0 → OK
- Scelgo sempre la corrispondenza con il prefisso più lungo!
  - ❖ Inoltra a ISPs-R-Us

Destinazione	Interfaccia
200.23.16.0/20	Verso Fly-By-Night-ISP
199.31.0.0/16	Verso ISPs-R-Us
200.23.18.0/23	Verso ISPs-R-Us

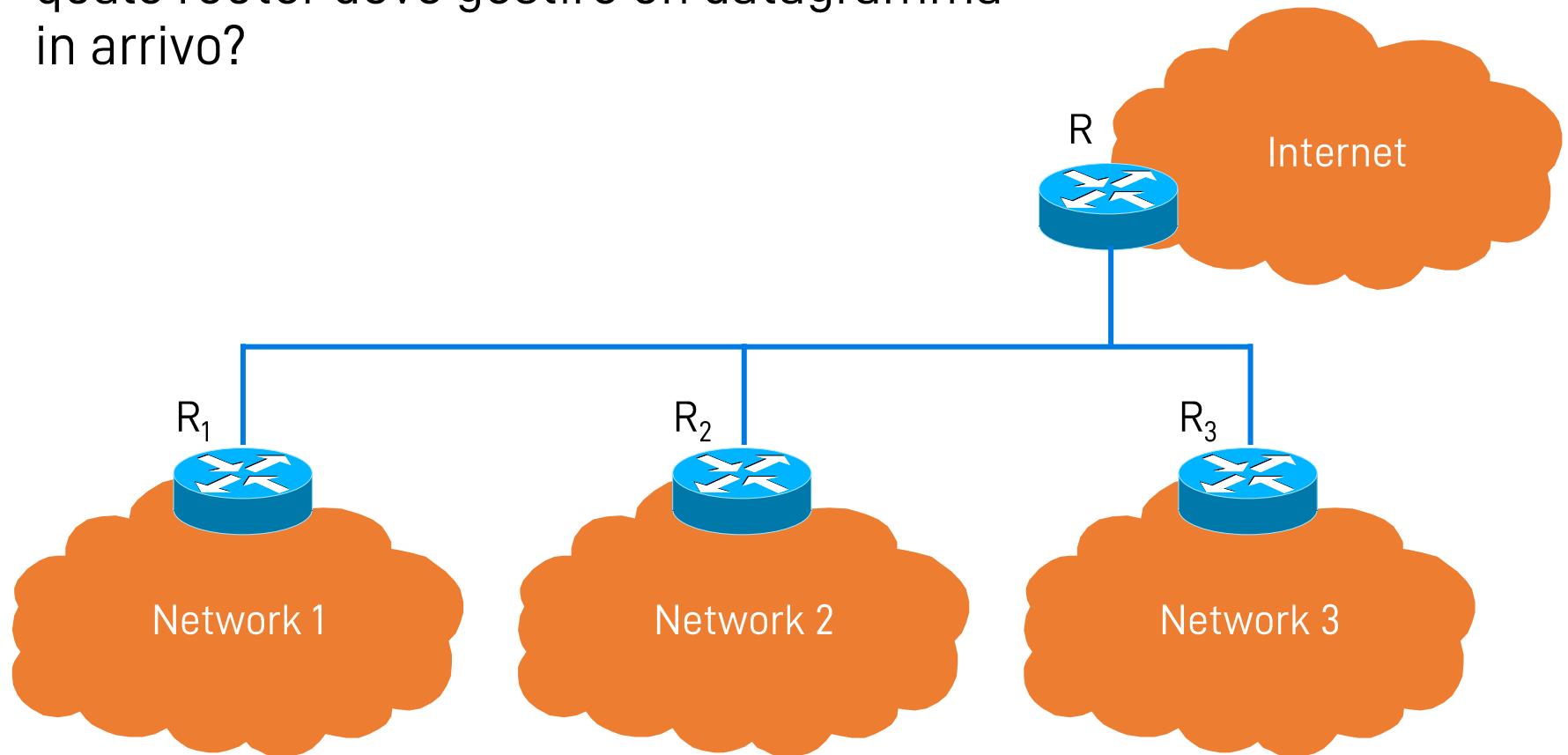
# E se non ci sono corrispondenze?

- Si usa un valore di default: il “default gateway”
  - ❖ Negli host, si tratta di solito dell'indirizzo IP del router
  - ❖ Per i router, è l'indirizzo IP di un altro router che si suppone sappia come inoltrare ulteriormente il datagramma
- Nelle tabelle, questa entry ha destinazione **0.0.0.0/0**
  - ❖ Supponiamo di voler inviare a **193.117.8.1**
    - Le prime 3 righe non corrispondono
    - Ultima riga: **193.117.8.1 & 0.0.0.0 (/0) = 0.0.0.0 → OK**

Destination	Interface
200.23.16.0/20	eth0
199.31.0.0/16	eth1
200.23.18.0/23	eth1
0.0.0.0/0	eth2

# Router multipli sulla stessa rete

- Se ci sono più router sulla stessa rete, quale router deve gestire un datagramma in arrivo?



# Soluzione: inserire l'informazione nella tabella di routing

- Le tabelle di routing includono l'IP cui inoltrare il datagramma!
  - ❖ Quello è l'indirizzo IP del router che se ne dovrà occupare
  - ❖ **NOTA:** Gli indirizzi IP sorgente e destinazione **NON CAMBIANO** quando il pacchetto viene inoltrato
    - Con l'eccezione del NAT, ne parliamo tra poco

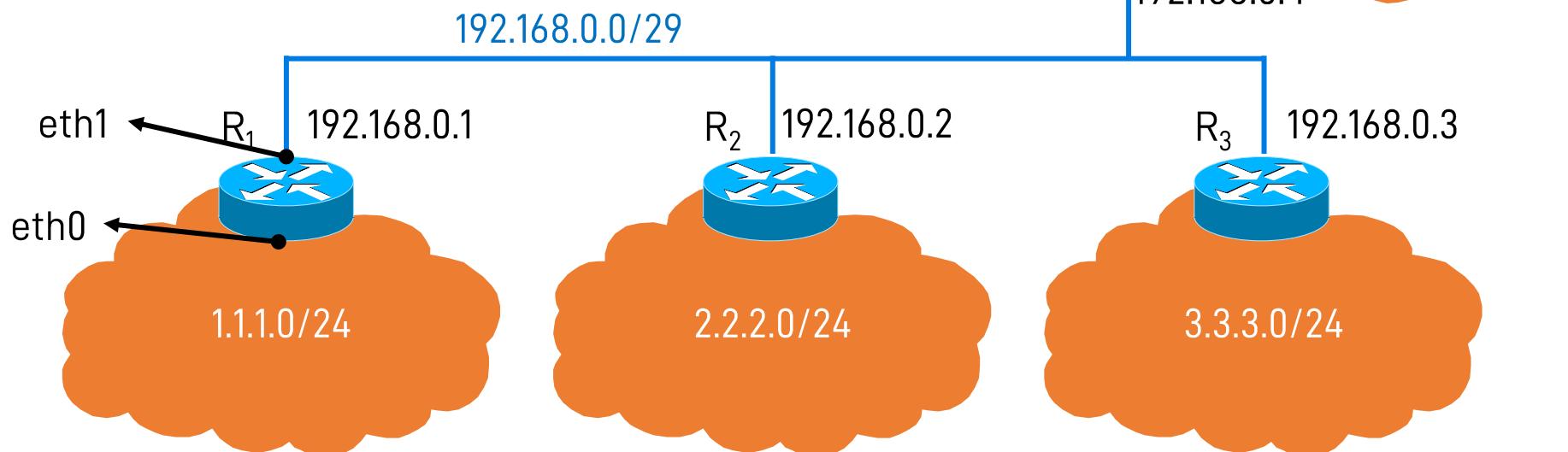


Destinazione	Interfaccia	Inoltra a
200.23.16.0/20	eth0	192.168.0.2
199.31.0.0/16	eth1	192.168.1.2
200.23.18.0/23	eth1	192.168.1.3
0.0.0.0/0	eth2	192.168.2.7

- Data plane: legge la tabella e trova interfaccia e destinatario usando longest prefix matching
- Control plane: popola la tabella

# Nel caso di prima

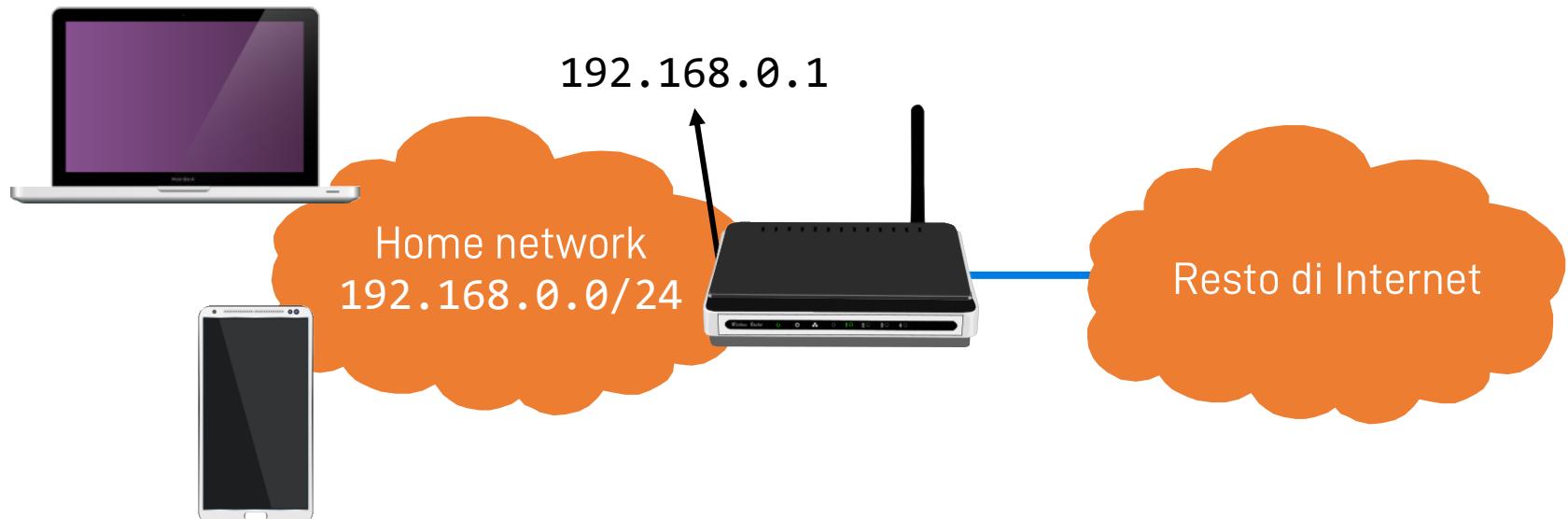
Destinazione	Interfaccia	Inoltra a
1.1.1.0/24	eth0	Diretto
2.2.2.0/24	eth1	192.168.0.2
3.3.3.0/24	eth1	192.168.0.3
192.168.0.0/29	eth1	Diretto
0.0.0.0/0	eth1	192.168.0.4



# Tipica rete casalinga

- Tabelle di routing sul laptop o smartphone

Destinazione	Interfaccia	Inoltra a
192.168.0.0/24	eth1	Diretto
0.0.0.0/0	eth1	192.168.0.1



# Sommario

- Visione d'insieme
  - ❖ Data plane e control plane
- Come è fatto un router
- Il protocollo IP
  - ❖ Formato datagrammi
  - ❖ Frammentazione
  - ❖ Indirizzamento e NAT
  - ❖ Indirizzamento classless
  - ❖ Tipi di indirizzi
  - ❖ Address Resolution Protocol (ARP)
  - ❖ Internet Control Message Protocol (ICMP)
- Il protocollo IP (segue)
  - ❖ Dynamic Host Configuration Protocol (DHCP)
  - ❖ Il viaggio di un pacchetto
  - ❖ IPv6
- Protocolli di instradamento
  - ❖ Link state: Dijkstra
  - ❖ Internet, AS, OSPF
  - ❖ Distance vector: Bellman-Ford
  - ❖ RIP
- BGP

# Indirizzi IP pubblici e privati

- La maggioranza dei  $2^{32} = 4.294.967.296$  indirizzi IP è pubblica e può essere assegnata a host e raggiunta su Internet
- Alcuni blocchi di indirizzi IP sono riservati
  - ❖ Non possono essere usati come indirizzi di destinazione in Internet
    - I router li bloccano
  - ❖ Sono definiti "**indirizzi IP privati**" e si usano per costruire reti private
  - ❖ Da 10.0.0.0 a 10.255.255.255 (10.0.0.0/8)
  - ❖ Da 172.16.0.0 a 172.31.255.255 (172.16.0.0/12)
  - ❖ Da 192.168.0.0 a 192.168.255.255 (192.168.0.0/16)
- Riutilizzabili in tutte le reti private
  - ❖ Non escono dalla rete
  - ❖ Sono sempre univoci all'interno di una rete privata
  - ❖ Di solito, sono assegnati dinamicamente

# Network Address Translation (NAT)

- Come fanno gli host con un indirizzo IP privato a mandare pacchetti verso Internet?
  - ❖ Proxy
    - Computer che ha un indirizzo pubblico e uno privato, e "media" la connessione per applicazioni specifiche (**D**: svantaggi?)
  - ❖ NAT: apparato allacciato sia alla rete privata sia ad Internet
    - Effettua la seguente mappatura

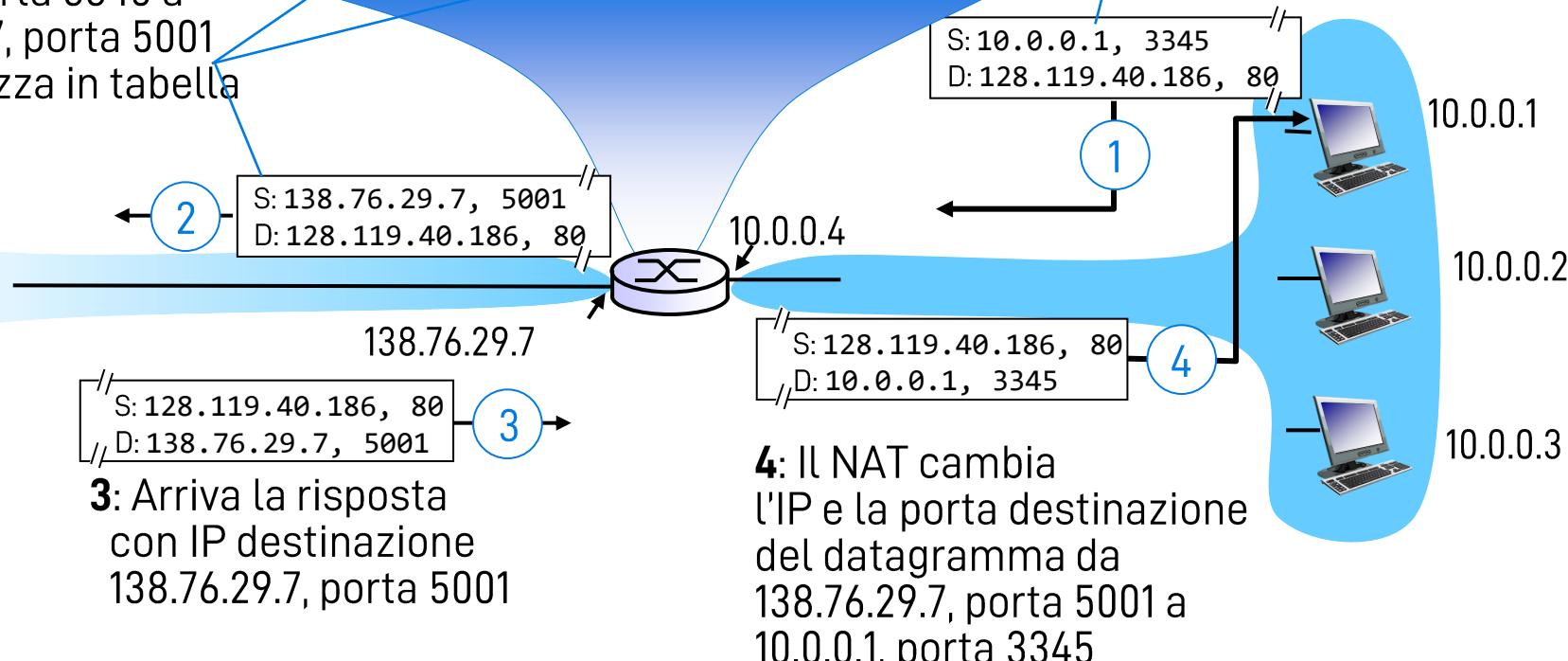
Da	A
<b>IP sorgente privato</b>	<b>IP pubblico del NAT</b>
IP destinazione pubblico	IP destinazione pubblico
<b>Porta sorgente</b>	<b>Un'altra porta sorgente</b>
Porta destinazione	Porta destinazione
Protocollo	Protocollo

- E viceversa

# NAT: network address translation

2: Il NAT cambia l'IP e la porta sorgente del datagramma da 10.0.0.1, porta 3345 a 138.76.29.7, porta 5001 e memorizza in tabella

Tabella di traduzione del NAT	
Lato esterno	Lato rete privata
138.76.29.7, 5001	10.0.0.1, 3345
.....	.....

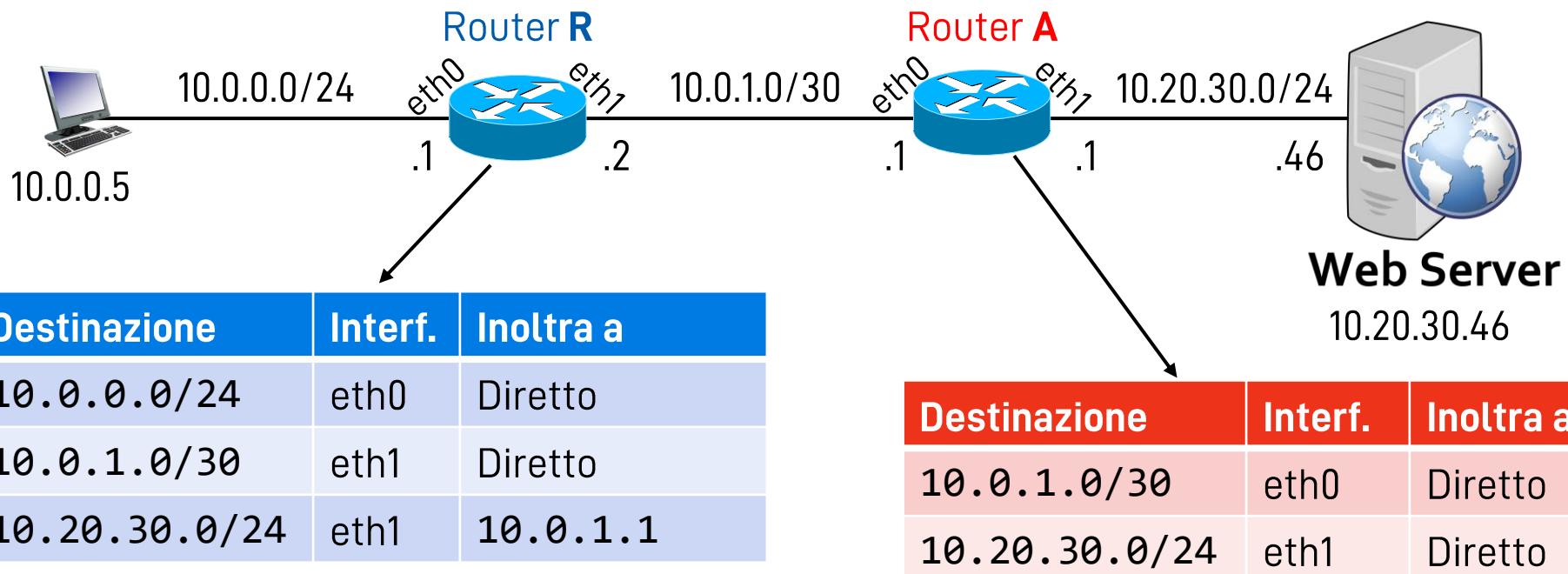


# NAT: network address translation

- Il numero di porta ha 16 bit
  - ❖ Fino a 60,000 connessioni simultanee con un solo IP pubblico
- Il NAT è stata a lungo la soluzione alla mancanza di IP pubblici
- Soluzione controversa:
  - ❖ Viola l'architettura a livelli: il NAT di fatto è un router e non dovrebbe cambiare le porte
  - ❖ In ogni caso, la mancanza di indirizzi dovrebbe essere risolta da IPv6
  - ❖ Costringe i programmati a tenere conto che potrebbe esistere (es. P2P)
  - ❖ Attraversamento del NAT: come fa un client a connettersi a un server che sta dietro un NAT?
    - Port forwarding
      - "Inoltra tutte le connessioni sulla porta 22 all'host 192.168.0.x"
    - Hole punching
    - UPnP Internet Gateway Device Protocol, STUN, ...

# NAT can be your friend

- A volte, un NAT è l'unico modo di connettere due reti se non si controllano tutti i router (es., qui supponete di non controllare A)



- Ad esempio, l'host a sinistra può aprire una pagina web sul server?
  - ❖ No, ma la cosa si può risolvere con un NAT su R!

# Indirizzi IP speciali

- Ci sono alcuni tipi di indirizzi IP “speciali” che non vengono mai assegnati agli host
- Esempi:
  - ❖ Gli indirizzi che denotano “tutta la rete”
  - ❖ L'indirizzo broadcast di una rete (“Directed Broadcast Address”)
  - ❖ L'indirizzo broadcast di rete locale (“Limited Broadcast Address”)
  - ❖ “Questo computer”
  - ❖ Indirizzi loopback
  - ❖ Gli indirizzi multicast
  - ❖ Gli indirizzi link-local

# Indirizzi di “tutta la rete”

- Sono usati per riferirsi al prefisso usato per una certa rete
- Si formano impostando l'HostID a tutti 0
- Per esempio, l'indirizzo 128.211.0.16/28 indica una rete
  - ❖ Infatti, tutti i bit oltre il 28-esimo sono a 0
  - ❖ 10000000.11010011.00000000.00010000
- Non si possono usare come indirizzo destinazione
- NOTA: 128.211.0.17/28 NON E' un indirizzo valido per denotare una rete
  - ❖ 10000000.11010011.00000000.00010001
  - ❖ Però può essere un host della rete 128.211.0.16/28

# Indirizzo broadcast di una rete

- ❑ “Directed Broadcast Address”
- ❑ Per semplificare l’invio di un pacchetto a tutti gli host di una rete, IP definisce un indirizzo broadcast specifico per quella rete
- ❑ Questo tipo di pacchetti viene inoltrato dai router
  - ❖ Una sola copia del datagramma viaggia in Internet
  - ❖ Finché non raggiunge la rete di destinazione
  - ❖ E viene consegnato a tutti gli host in quella rete
- ❑ Si genera usando un HostID di tutti 1, ad esempio
  - ❖ 10000000.11010011.00000000.00011111

# Indirizzo broadcast di rete locale

- ❑ “Limited broadcast address”
- ❑ Si riferisce al fatto che i datagrammi con questo indirizzo destinazione non escono mai dalla rete locale
  - ❖ Nessun router li inoltra
- ❑ IP riserva l'indirizzo con 32 bit tutti a 1 per questo scopo
  - ❖  $255.255.255.255 = 11111111.11111111.11111111.11111111$
- ❑ Si usa quando un host si accende e deve ancora capire su che rete si trova
  - ❖ Finche non ha un indirizzo IP valido, IP manda i datagrammi all'indirizzo broadcast di rete locale

# “Questo computer”

- Prima di inviare datagrammi verso Internet, un host deve conoscere il proprio indirizzo IP
  - La pila di protocolli TCP/IP include un protocollo per l'assegnazione automatica degli indirizzi
    - ❖ ... peccato che anche quella richieda IP per comunicare ☺
  - Soluzione: quando un host parte ("boots up") e non ha un indirizzo IP valido, imposta come IP sorgente un indirizzo di tutti 0
    - ❖ 00000000.00000000.00000000.00000000

# Indirizzi loopback

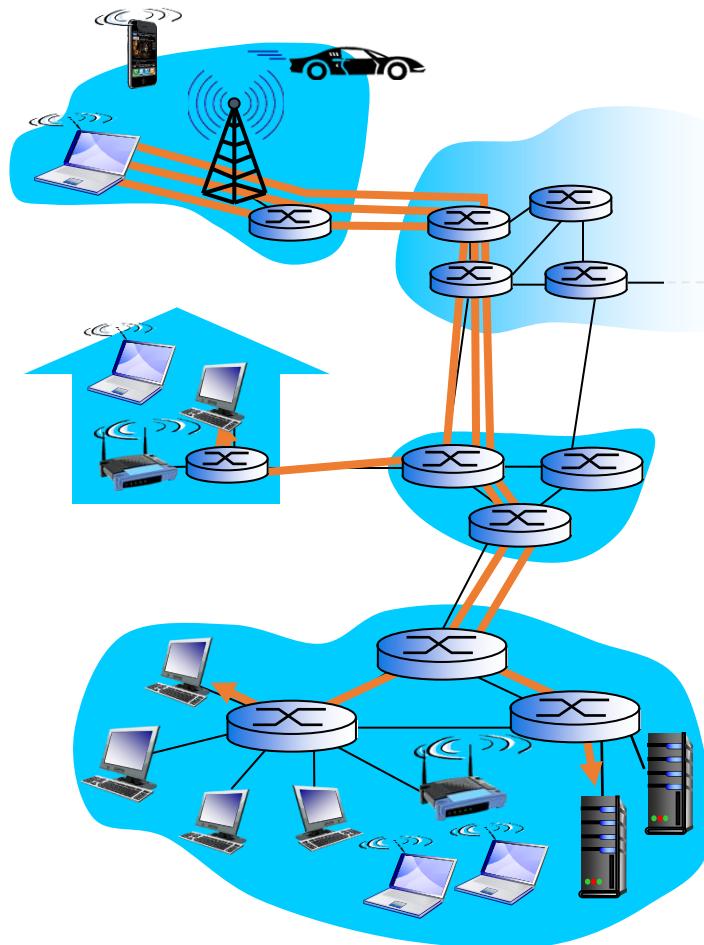
- Si usano per testare le applicazioni di rete
  - ❖ Se avete componenti che comunicano attraverso una rete, invece di farli girare su computer diversi ...
  - ❖ ... li fate girare sullo stesso computer attraverso indirizzi loopback
- Quando un componente invia un datagramma a un altro componente dell'applicazione
  - ❖ Il datagramma scende la pila protocollare fino al livello 3 (IP)
  - ❖ E poi risale verso il secondo programma
  - ❖ Nessun pacchetto lascia il computer
- IP riserva il prefisso **127.0.0.0/8** per il loopback
  - ❖ HostID irrilevante, i datagrammi sono comunque gestiti come sopra
  - ❖ Spesso si usa l'HostID 1 → **127.0.0.1 !!!**

# Indirizzi multicast

- Servono a inviare i pacchetti a un gruppo di host distribuito su Internet, anche globalmente
- I router si occupano di creare le copie necessarie
- In IPv4 gli indirizzi multicast iniziano con **1110**
  - ❖ Da **224.0.0.0** a **239.255.255.255**
- Bad news: il supporto per il multicast in Internet è minimo, e il traffico multicast viene spesso bloccato

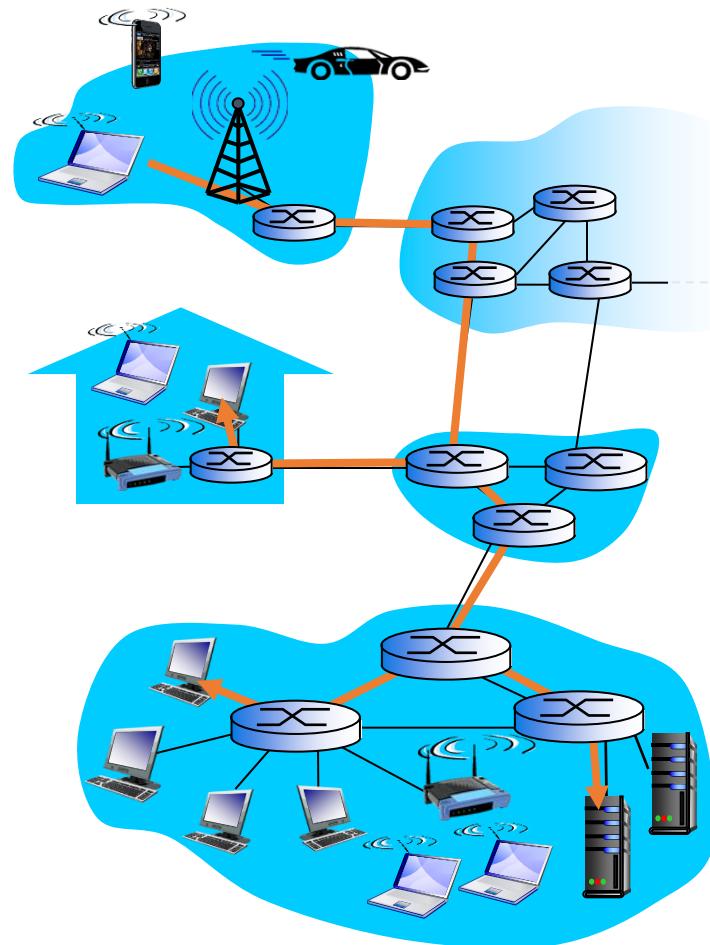
# Unicast multiplo ...

- ❑ Una copia di un pacchetto per ogni destinazione interessata
  - ❖ Può usare lo stesso link più volte per gli stessi dati



# ... confrontato con routing multicast

- Multicast: minimizza il numero di copie e reduce l'uso delle risorse



# Indirizzi IP link-local

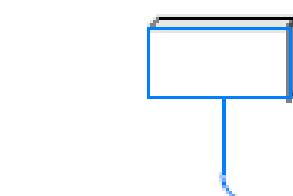
- Sottorete riservata per consentire le comunicazioni quando un host non riesce a “trovare” un indirizzo IP
- 169.254.0.0/16
- Ogni host sceglie a caso un IP da questa sottorete
- Questo consente agli host di comunicare tra loro nella sottorete, anche se li mantiene isolati dall'esterno
- Lo vedremo meglio più avanti parlando di DHCP

# Indirizzi IP per i router

- Ad ogni router si assegnano due o più indirizzi IP
  - ❖ (Almeno) un indirizzo IP per interfaccia
- Infatti
  - ❖ Un router connette multiple reti (per definizione)
  - ❖ Ogni rete ha un prefisso unico, quindi per parlare su quella rete serve un indirizzo IP in cui figura *esattamente* quel prefisso
- Concetto importante:
  - ❖ Gli indirizzi IP non contrassegnano un host ...
  - ❖ ... ma un'INTERFACCIA DI RETE, quindi una connessione logica tra un host e una rete
  - ❖ Quindi un sistema con molte interfacce (un router) avrà necessariamente diversi indirizzi IP

# Esempio

Ethernet 131.108.0.0 /16



router 1

131.108.99.5

223.240.129.2

Wi-Fi Net

223.240.129.0 /24

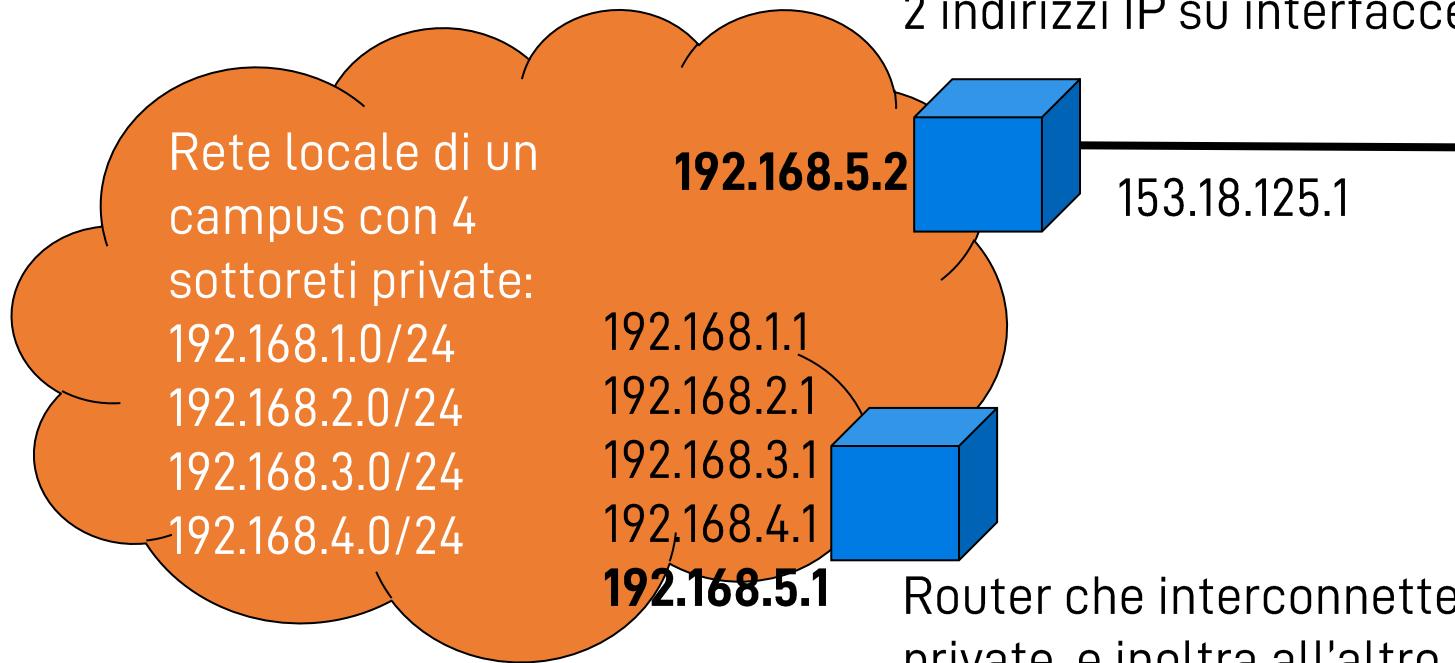
223.240.129.17

router 2

78.0.0.17

WAN 78.0.0.0 /8

# Esempio con indirizzi IP multipli

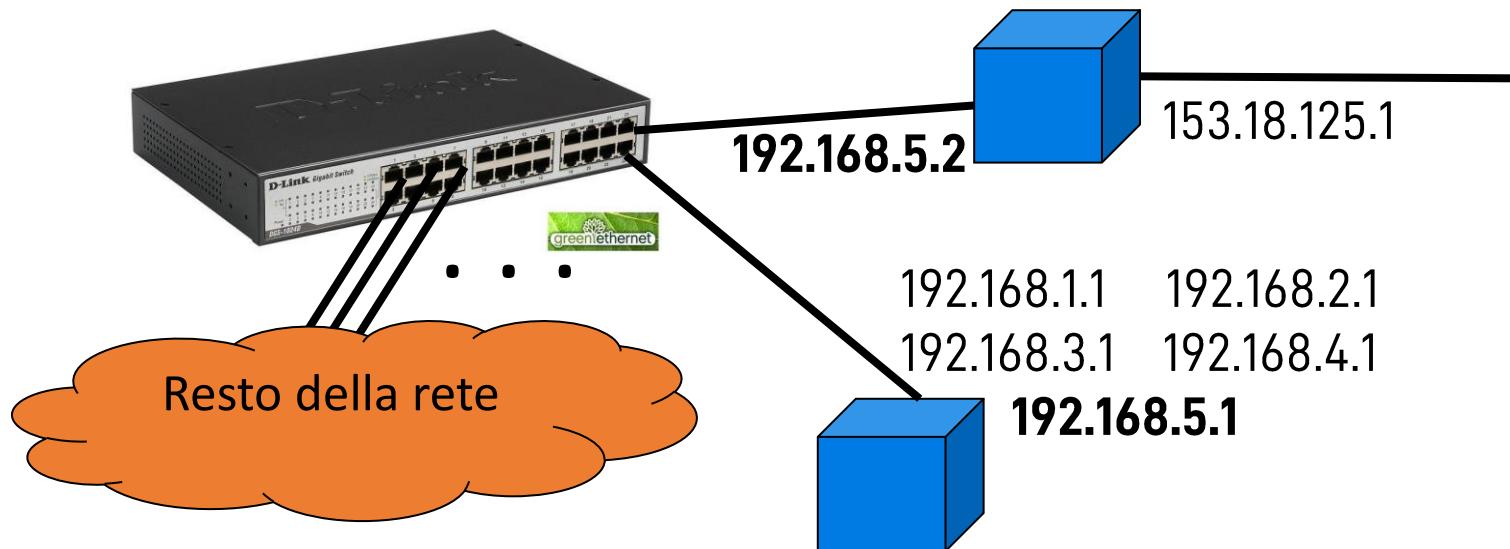


Router e NAT per l'accesso a Internet:  
2 indirizzi IP su interfacce diverse

Router che interconnette le 4 reti private, e inoltra all'altro router i datagrammi destinati a Internet : 5 IP diversi sulla stessa interfaccia

# Esempio con indirizzi IP multipli

Router e NAT per l'accesso a Internet:  
2 indirizzi IP su interfacce diverse



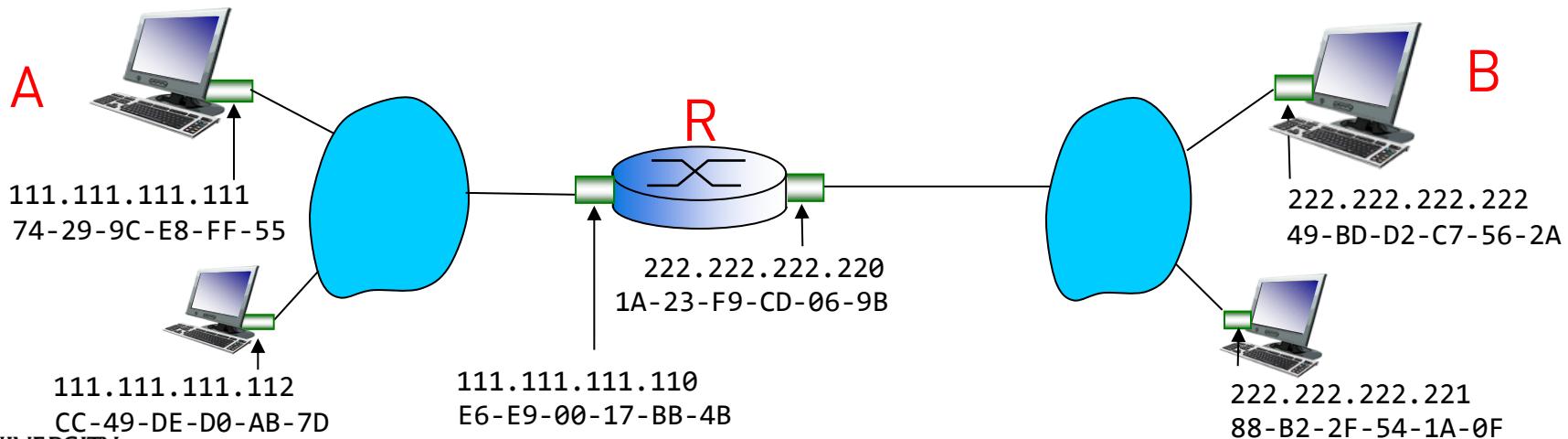
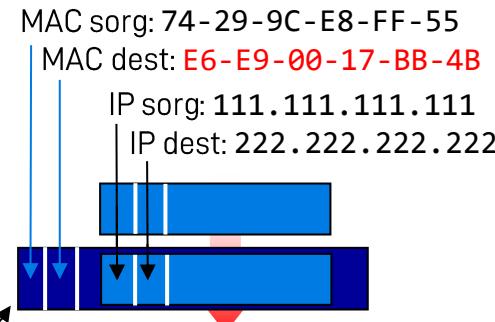
192.168.1.1    192.168.2.1  
192.168.3.1    192.168.4.1

**192.168.5.1**

Router che interconnecte le 4 reti private, e inoltra all'altro router i datagrammi destinati a Internet : 5 IP diversi sulla stessa interfaccia

# Per chiudere il cerchio: livello 2

- Per comprendere il viaggio di un pacchetto dal mittente A al destinatario B, mancano alcuni dettagli sul livello 2
- Datagramma IP → encapsulato in un frame (pacchetto di livello 2)
  - ❖ Per poter trasmettere il frame, bisogna inserire l'indirizzo di livello 2 di sorgente e destinazione (i "MAC address") nell'header di livello 2
- MAC address: stringa di 48 bit, rappresentata come 12 cifre esadecimale (ogni interfaccia ne ha uno)
- Come recuperiamo l'indirizzo MAC del destinatario?

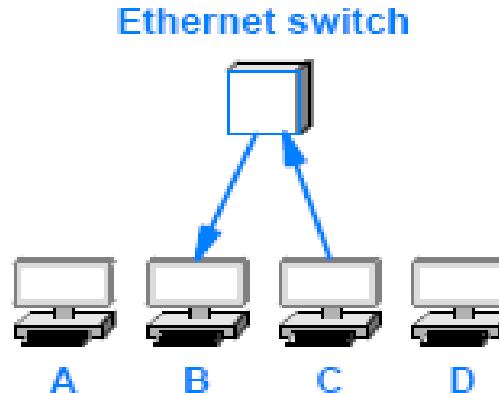
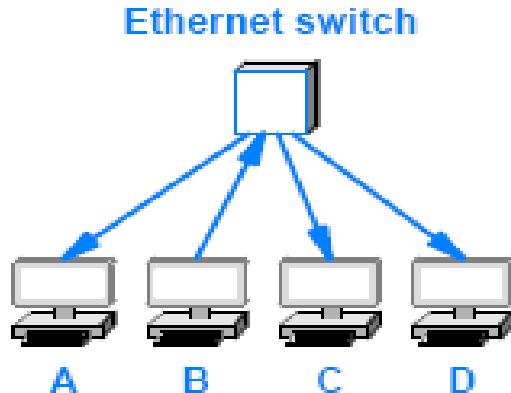


# Sommario

- Visione d'insieme
  - ❖ Data plane e control plane
- Come è fatto un router
- Il protocollo IP
  - ❖ Formato datagrammi
  - ❖ Frammentazione
  - ❖ Indirizzamento e NAT
  - ❖ Indirizzamento classless
  - ❖ Tipi di indirizzi
  - ❖ Address Resolution Protocol (ARP)
  - ❖ Internet Control Message Protocol (ICMP)
- Il protocollo IP (segue)
  - ❖ Dynamic Host Configuration Protocol (DHCP)
  - ❖ Il viaggio di un pacchetto
  - ❖ IPv6
- Protocolli di instradamento
  - ❖ Link state: Dijkstra
  - ❖ Internet, AS, OSPF
  - ❖ Distance vector: Bellman-Ford
  - ❖ RIP
- BGP

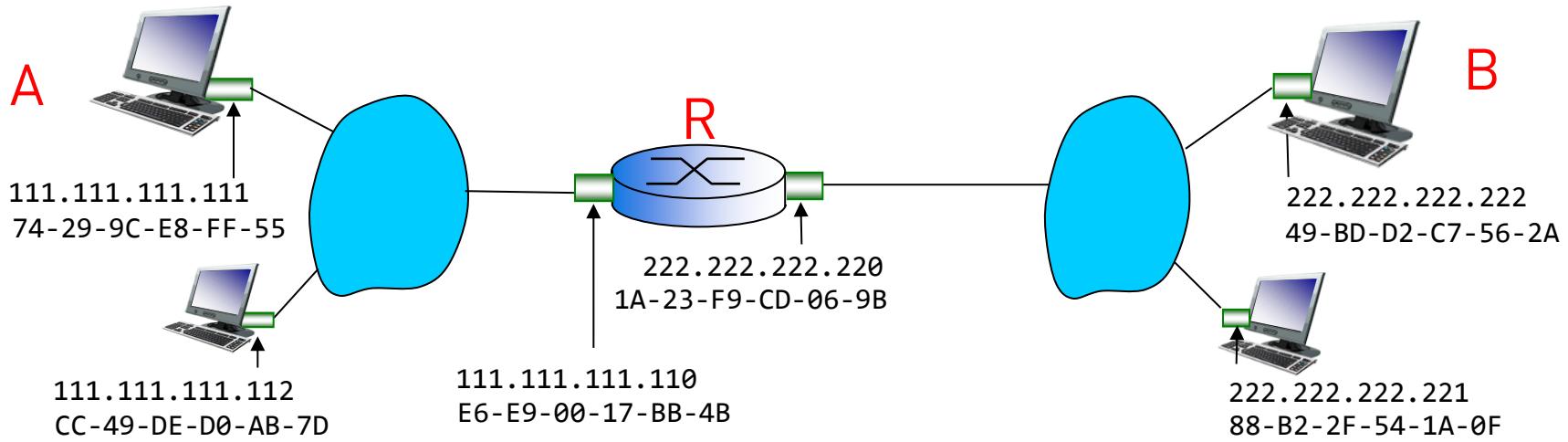
# Principio dell'ARP

- Se l'host destinatario di un datagramma IP si trova sulla mia rete, voglio inviargli il pacchetto direttamente
  - ❖ **D**: come faccio a sapere se questo è il caso?
  - ❖ **R**: perché il prefisso di rete (NetID) del destinatario è uguale al mio
- Address Resolution Protocol
  - ❖ Se l'indirizzo IP del destinatario è C, il mittente invia in broadcast una richiesta "*Mi serve il MAC address che corrisponde all'indirizzo IP C*"
  - ❖ La richiesta raggiunge tutti gli host sulla rete
  - ❖ L'host con indirizzo IP C risponde: "*Sono io e questo è il mio indirizzo MAC*"



# Principio dell'ARP

- Si fa solo per comunicazioni punto-punto sulla stessa rete
  - ❖ Anche se gli indirizzi IP sorgente e destinazione non cambiano lungo il percorso, gli indirizzi MAC cambiano ad ogni salto
  - ❖ In ogni caso, non si cerca mai l'indirizzo MAC di un host su una rete diversa dalla propria (**D**: perché?)
    - In questo caso si passa sempre il pacchetto a un router
    - Ovviamente, dovete usare ARP per trovare l'indirizzo MAC del router



# Formato del messaggio ARP

0	8	16	24	31			
HARDWARE ADDRESS TYPE		PROTOCOL ADDRESS TYPE					
HADDR LEN	PADDR LEN	OPERATION					
SENDER HADDR (first 4 octets)							
SENDER HADDR (last 2 octets)		SENDER PADDR (first 2 octets)					
SENDER PADDR (last 2 octets)		TARGET HADDR (first 2 octets)					
TARGET HADDR (last 4 octets)							
TARGET PADDR (all 4 octets)							

# Incapsulamento di ARP in un frame

- Ad esempio, frame Ethernet
- I messaggi ARP sono interpretati come dati da trasportare e incapsulati nel payload del frame



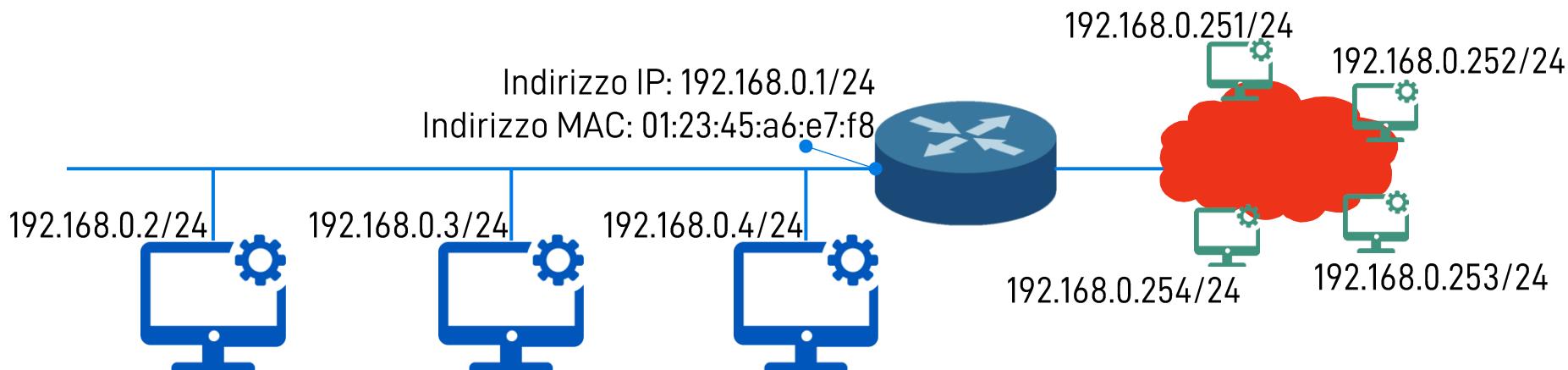
- Nell'header del frame c'è un campo "Type"
  - ❖ Identifica il tipo di dati trasportati
- In Ethernet, se il campo contiene il valore 2054 (0x806), significa che i dati sono un messaggio ARP

# ARP caching

- Per evitare di inviare un messaggio ARP per ogni datagramma, si mantengono in memoria ("cache") le risposte ARP ricevute
- Gestione della cache
  - ❖ Se si riceve una nuova risposta per una certa corrispondenza tra indirizzo IP e indirizzo MAC, questa sovrascrive la precedente
  - ❖ Le corrispondenze più vecchie vengono cancellate periodicamente (ogni 30 secondi circa)
  - ❖ La cache viene consultata sempre *prima* di inviare una richiesta

# Uso avanzato di ARP: il proxy ARP

- Un proxy ARP è una macchina che restituisce un indirizzo MAC facendo le veci di un host che si trova su una rete diversa
- Esempio: router di accesso alla rete sulla sinistra
  - ❖ Un host sulla rete di sinistra vuole comunicare con uno degli host della rete di accesso a destra
    - Il router di accesso risponde con il proprio MAC ai messaggi ARP
    - E poi si incarica di inoltrare i pacchetti
  - ❖ Così gli host sulla destra non devono stare in una rete separata

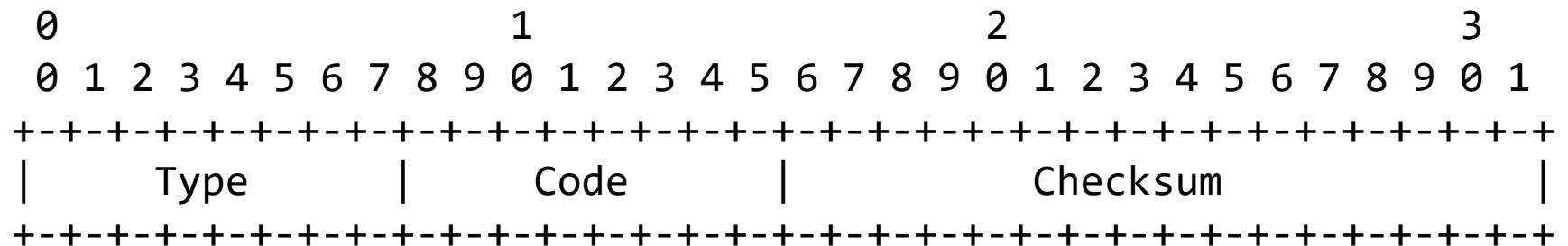


# Sommario

- Visione d'insieme
  - ❖ Data plane e control plane
- Come è fatto un router
- Il protocollo IP
  - ❖ Formato datagrammi
  - ❖ Frammentazione
  - ❖ Indirizzamento e NAT
  - ❖ Indirizzamento classless
  - ❖ Tipi di indirizzi
  - ❖ Address Resolution Protocol (ARP)
  - ❖ Internet Control Message Protocol (ICMP)
- Il protocollo IP (segue)
  - ❖ Dynamic Host Configuration Protocol (DHCP)
  - ❖ Il viaggio di un pacchetto
  - ❖ IPv6
- Protocolli di instradamento
  - ❖ Link state: Dijkstra
  - ❖ Internet, AS, OSPF
  - ❖ Distance vector: Bellman-Ford
  - ❖ RIP
- BGP

# Internet Control Message Protocol

- ❑ IP include un protocollo "partner" chiamato ICMP
    - ❖ Usato per notificare errori alla sorgente di un datagramma
    - ❖ Ottimo strumento per gestione e manutenzione delle reti
  - ❑ Interdipendenti
    - ❖ IP ha bisogno di ICMP per segnalare errori
    - ❖ ICMP si serve di IP per trasportare i messaggi
  - ❑ Header semplice: type identifica il tipo di messaggio



# Internet Control Message Protocol

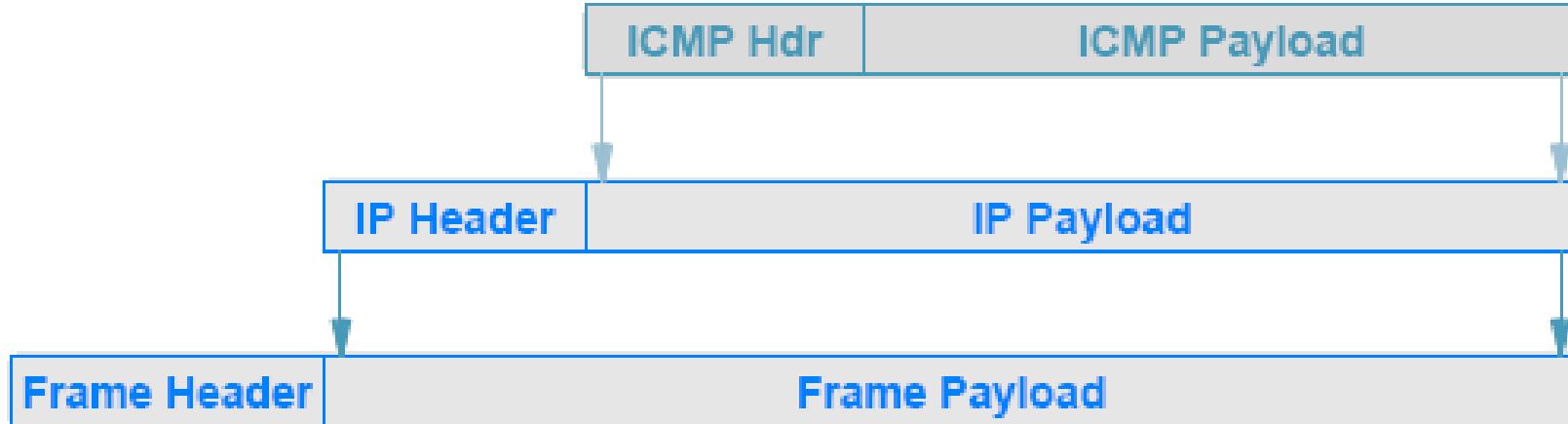
Campo Type	Descrizione
Destination unreachable	Il pacchetto non può essere consegnato
Port unreachable	Nessuno all'host destinatario ascolta sulla porta specificata
Time exceeded	Il campo TTL dell'header IP è sceso a 0 (routing loop?)
Parameter problem	Valore non valido in un campo dell'header IP
Source quench	Chiede alla sorgente di rallentare l'invio di dati
Redirect	Dice al router a quale host inoltrare il datagramma
Echo and echo reply	Usati da <b>ping</b> per capire se un host è attivo
Timestamp request/reply	Come i comandi echo, ma per l'orologio locale del router
Router advertisement/solicitation	Per proporsi come router, o per chiedere quali router ci sono nelle vicinanze
Fragmentation needed	Flag IP «non frammentare» a 1, ma datagramma eccede l'MTU

# Internet Control Message Protocol

- Due classi di messaggi
  - ❖ Quelli usati per segnalare errori
    - Es., Time Exceeded e Destination Unreachable
  - ❖ Quelli usati per recuperare informazioni
    - Es., Echo Request e Echo Reply
- Echo Request/Reply → usati dal comando ping
  - ❖ Quando un host riceve un "echo request," restituisce un messaggio ICMP con gli stessi dati presenti nella richiesta ("echo"...)
- I messaggi ICMP funzionano come qualunque pacchetto IP e sono inviati senza particolari priorità
  - ❖ Eccezione: se un messaggio ICMP genera un errore, non si invia nessun messaggio di errore
  - ❖ Evita di congestionare Internet con messaggi di errore

# Internet Control Message Protocol

- I contenuti ICMP vengono trasportati come dati in un datagramma:



# Sfruttare ICMP per ping e traceroute

## □ Ping

- ❖ Echo request + echo reply
- ❖ Si verifica l'esistenza di un percorso verso la destinazione e se ne misura l'RTT

## □ Traceroute

- ❖ Si inviano datagrammi IP con TTL sempre maggiore (1, 2, 3, ...)
- ❖ Con TTL = 1, il primo router decrementa il TTL a 0
  - Scarta il pacchetto
  - Invia un messaggio "Time Exceeded" con il proprio indirizzo IP
- ❖ Con TTL = 2, il primo router decrementa il TTL a 1, e il secondo a 0
  - Stesse conseguenze
- ❖ Si ripete finché non si è raggiunta la destinazione
- ❖ Usato per rilevare il percorso seguito dal pacchetto

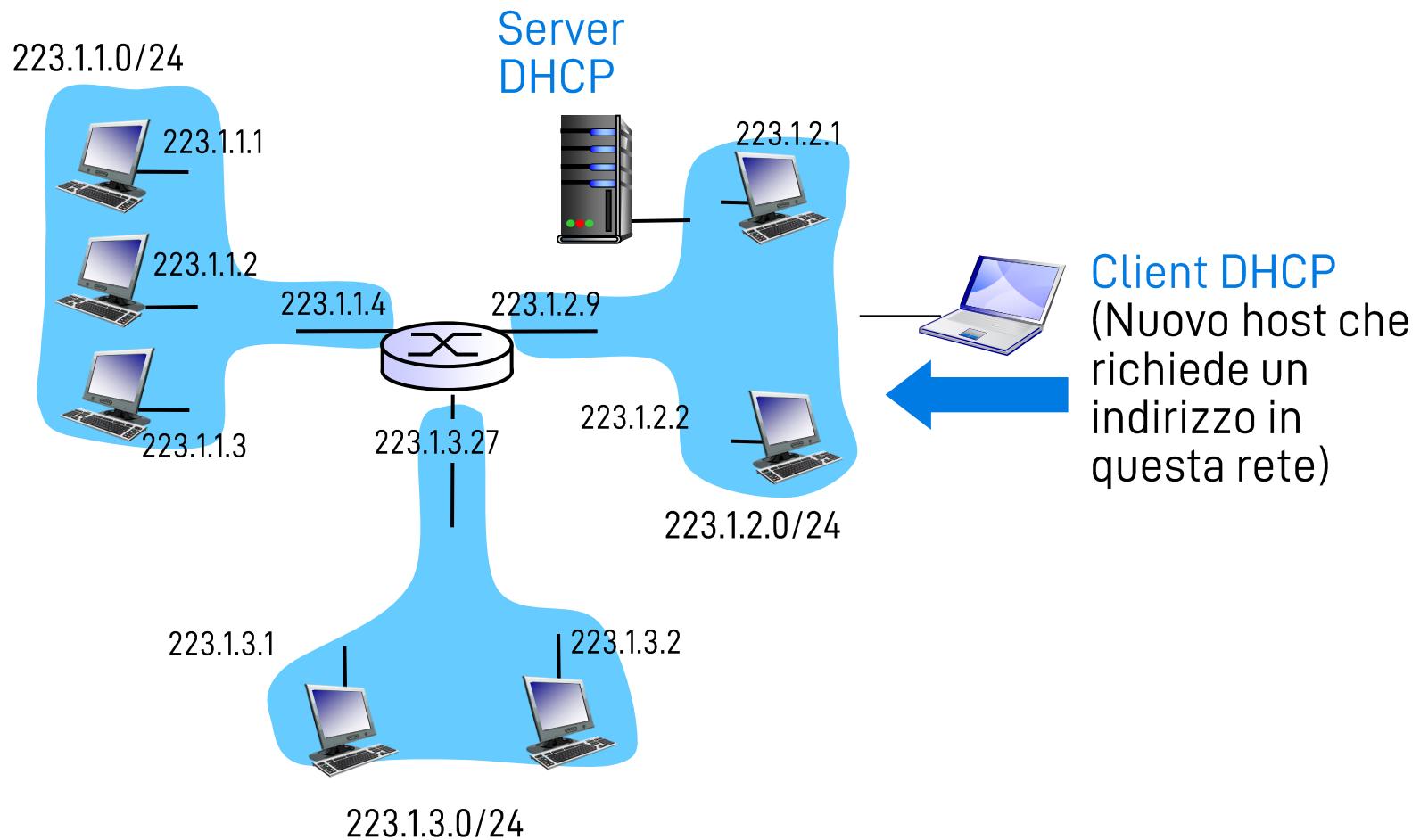
# Sommario

- Visione d'insieme
  - ❖ Data plane e control plane
- Come è fatto un router
- Il protocollo IP
  - ❖ Formato datagrammi
  - ❖ Frammentazione
  - ❖ Indirizzamento e NAT
  - ❖ Indirizzamento classless
  - ❖ Tipi di indirizzi
  - ❖ Address Resolution Protocol (ARP)
  - ❖ Internet Control Message Protocol (ICMP)
- Il protocollo IP (segue)
  - ❖ Dynamic Host Configuration Protocol (DHCP)
  - ❖ Il viaggio di un pacchetto
  - ❖ IPv6
- Protocolli di instradamento
  - ❖ Link state: Dijkstra
  - ❖ Internet, AS, OSPF
  - ❖ Distance vector: Bellman-Ford
  - ❖ RIP
- BGP

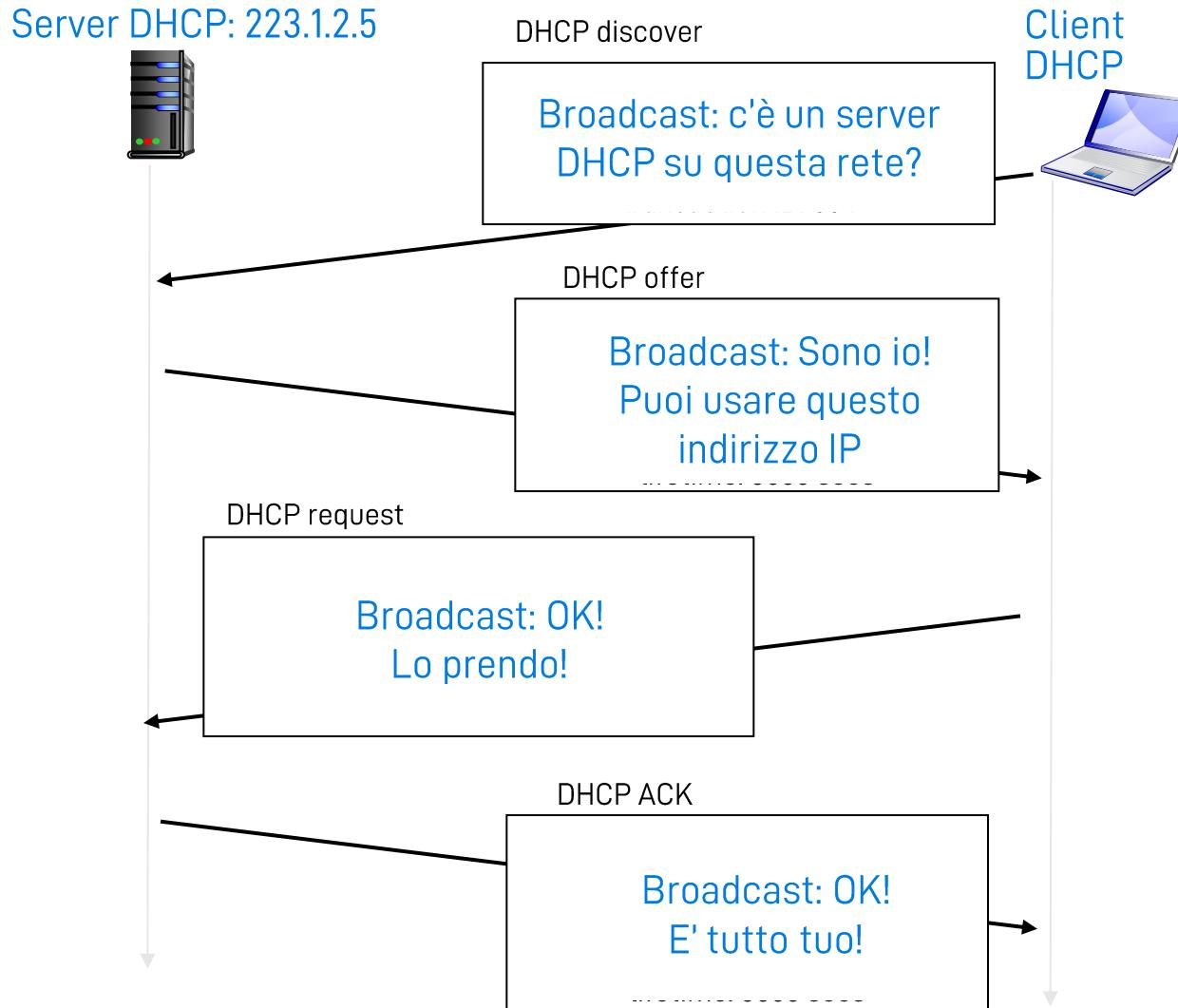
# DHCP

- Consente di assegnare un indirizzo IP a un host quando questo si accende, in maniera del tutto automatica
- Permette di liberare indirizzi quando non utilizzati, affinché possano essere assegnati ad altri host
- In sintesi la procedura è:
  - ❖ L'host invia un messaggio "DHCP discover" [opzionale]
  - ❖ Il server DHCP (di solito un software che gira in un altro host, o in un router) risponde con un messaggio "DHCP offer" [opzionale]
  - ❖ L'host richiede un indirizzo IP con un messaggio "DHCP request"
  - ❖ Il server DHCP invia un messaggio "DHCP ack"

# DHCP client-server scenario



# DHCP client-server scenario



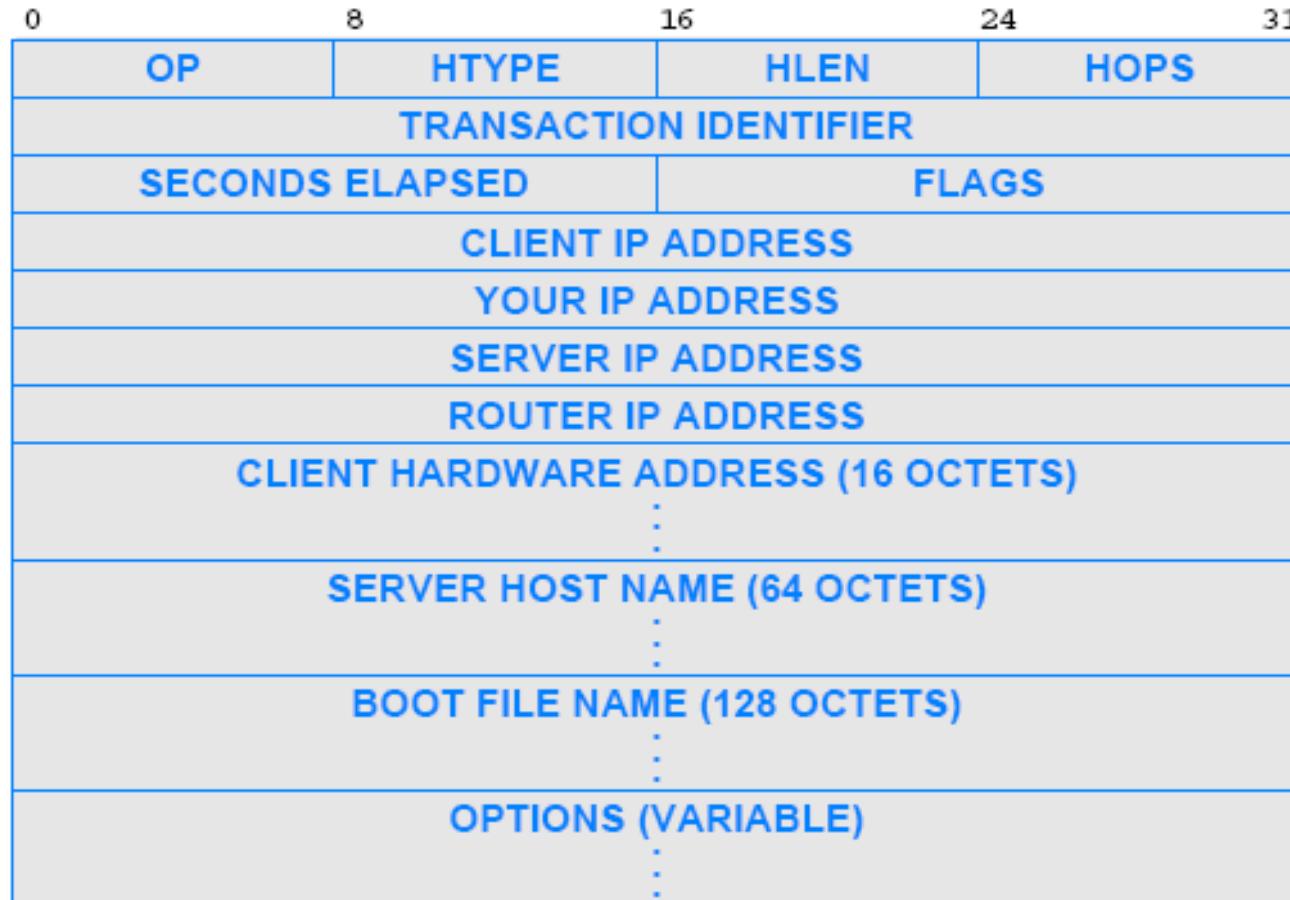
# Prestiti ("lease") DHCP

- L'indirizzo IP fornito da DHCP rimane valido per un tempo limitato
- Quando il prestito finisce, il server rimette l'indirizzo IP in un insieme di indirizzi utilizzabili
- A quel punto, il client può decider di cambiare indirizzo IP o negoziare un'estensione del prestito
- Normalmente, DHCP approva tutte le estensioni
  - ❖ Un host può continuare a funzionare senza interruzioni
  - ❖ Per varie ragioni, comunque, un server potrebbe negare l'estensione di un lease
  - ❖ In quel caso, il client DEVE smettere di usare l'IP

# Altri dettagli su DHCP

- ❑ DHCP usa UDP, che non è un protocollo di trasporto affidabile
  - ❖ DHCP è progettato per essere robusto a perdite e duplicati
  - ❖ Nessuna risposta dal server: l'host ritrasmette la richiesta
  - ❖ Se arriva una risposta duplicata: l'host ignora la copia extra
- ❑ Il client, quando trova un server DHCP, ne memorizza l'indirizzo in una cache
- ❑ Per evitare richieste simultanee
  - ❖ Ritardi casuali prima di inviare di nuovo una richiesta

# Formato dei messaggi DHCP



# Formato dei messaggi DHCP

- ❑ OP: specifica se si tratta di una risposta o una richiesta
- ❑ HTYPE e HLEN specificano il tipo di hardware e la lunghezza dell'indirizzo di livello 2
- ❑ FLAGS specifica se il mittente può ricevere broadcast o risposte dirette
- ❑ HOPS specifica quanti server hanno inoltrato la richiesta
- ❑ TRANSACTION IDENTIFIER si usa per far corrispondere le richieste alle risposte
- ❑ SECONDS ELAPSED dice quanto tempo è passato da quando l'host si è attivato
- ❑ Dimensioni fissate (eccetto per il campo OPTIONS)

# Formato dei messaggi DHCP

- I campi rimanenti trasportano le informazioni richieste
  - ❖ YOUR IP ADDRESS si usa per trasportare l'indirizzo IP offerto al client DHCP
  - ❖ Il server mette il proprio indirizzo IP e nome nei campi SERVER IP ADDRESS e SERVER HOST NAME
  - ❖ ROUTER IP ADDRESS contiene l'indirizzo IP di un router di default
    - Il DHCP lo usa per comunicare a un host il default gateway

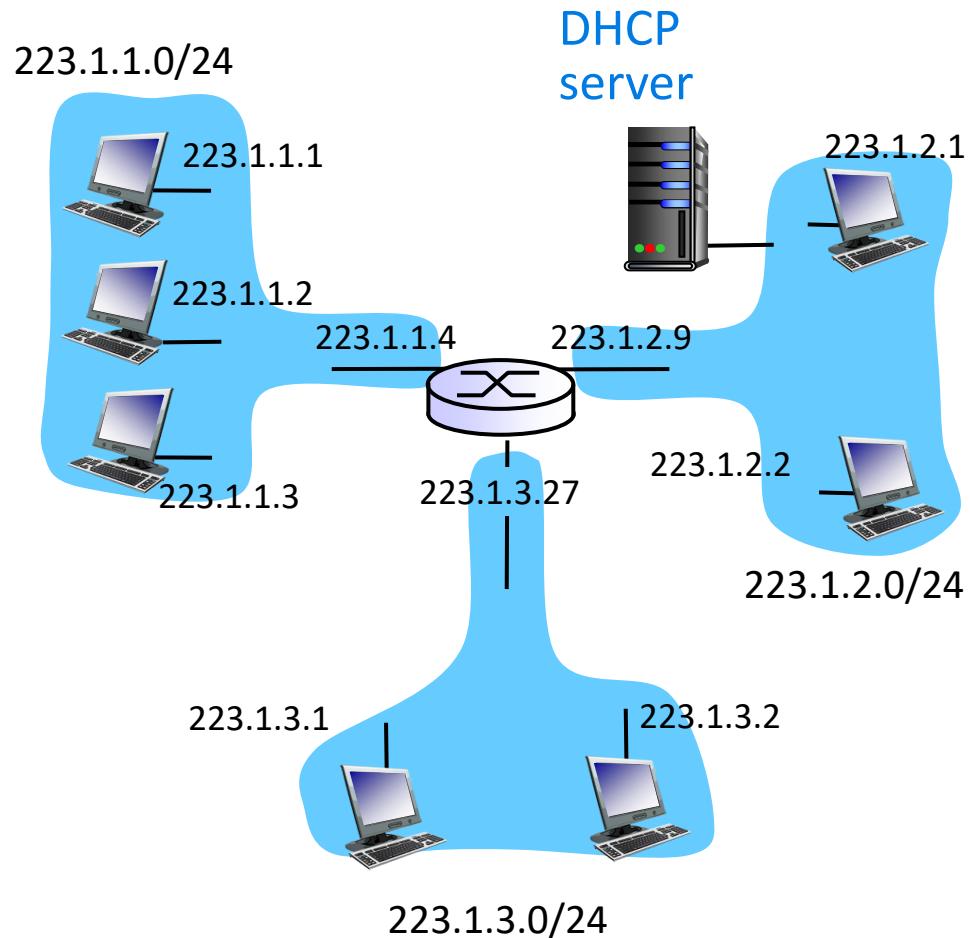
# DHCP: altre informazioni

DHCP può restituire anche

- ❑ Il nome e l'indirizzo IP di un server DNS
- ❑ La maschera di rete per suddividere il prefisso dal suffisso
- ❑ Altre informazioni (es. il percorso di un file con le istruzioni per configurare un host all'avvio)

# DHCP in una rete reale

- ❑ L'indirizzo IP dei messaggi DHCP è quello broadcast di rete locale (255.255.255.255)
  - ❑ I router non lo inoltrano
    - ❖ In ogni caso, i router possono essere configurati per inoltrare i messaggi DHCP a un server noto
    - ❖ **D:** altrimenti?
    - ❖ Altrimenti, servirebbe un server DHCP in ogni sottorete



# E se non c'è nessun DHCP?

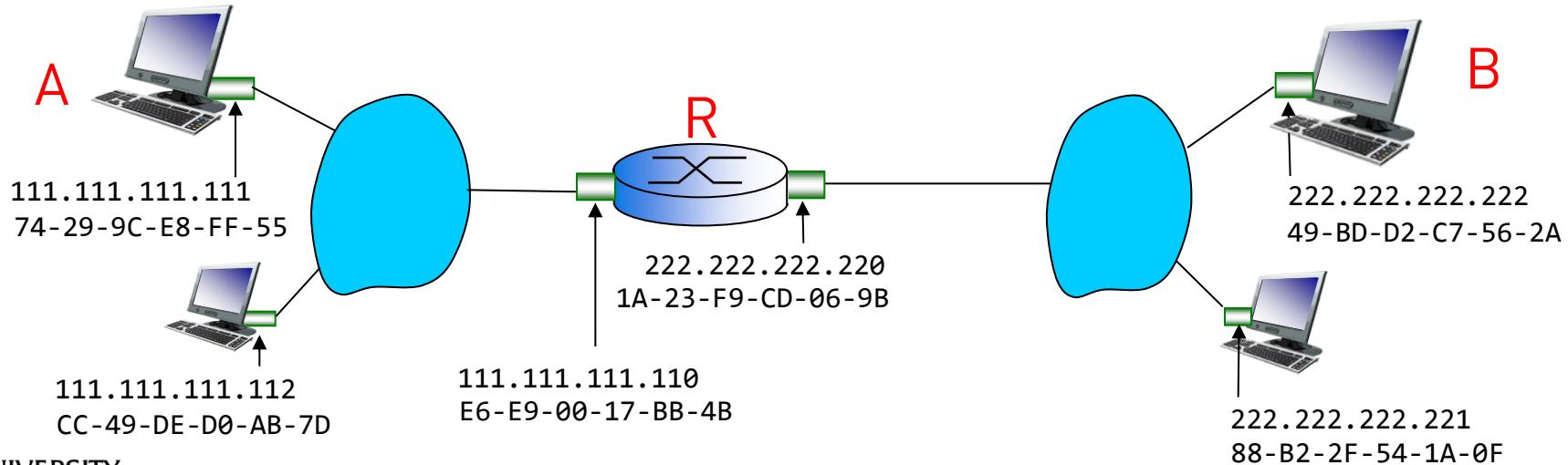
- ❑ Si disabilita il client DHCP di un host e si configura l'indirizzo IP manualmente
- ❑ Altrimenti, il client DHCP imposta un indirizzo tipo link-local
  - ❖ 169.254.0.0/16
- ❑ Passi
  1. Si sceglie casualmente un indirizzo tra 169.254.0.1 e 169.254.255.254
  2. Si cerca se esiste un interfaccia di rete con questo indirizzo
    - Come? Con una risoluzione ARP
  3. Se la si trova, si torna al punto 1
  4. Altrimenti, l'host si tiene l'indirizzo scelto al punto 1
- ❑ Consente agli host di parlarsi almeno sulla rete locale

# Sommario

- Visione d'insieme
  - ❖ Data plane e control plane
- Come è fatto un router
- Il protocollo IP
  - ❖ Formato datagrammi
  - ❖ Frammentazione
  - ❖ Indirizzamento e NAT
  - ❖ Indirizzamento classless
  - ❖ Tipi di indirizzi
  - ❖ Address Resolution Protocol (ARP)
  - ❖ Internet Control Message Protocol (ICMP)
- Il protocollo IP (segue)
  - ❖ Dynamic Host Configuration Protocol (DHCP)
  - ❖ Il viaggio di un pacchetto
  - ❖ IPv6
- Protocolli di instradamento
  - ❖ Link state: Dijkstra
  - ❖ Internet, AS, OSPF
  - ❖ Distance vector: Bellman-Ford
  - ❖ RIP
- BGP

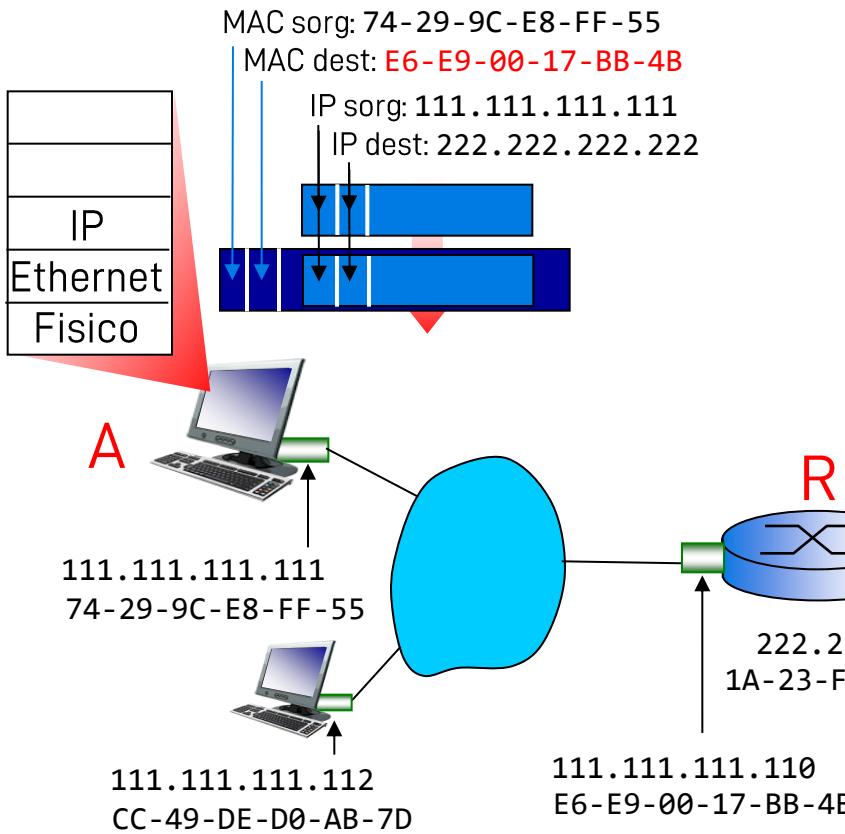
# Procedura di routing passo a passo

- Facciamo attenzione all'indirizzamento IP (datagramma) e MAC (frame)
- Assumiamo che A conosca
  - ❖ L'indirizzo IP di B
  - ❖ L'indirizzo IP del router cui inviare il pacchetto (**D**: come?)
  - ❖ L'indirizzo MAC del router R (**D**: come?)



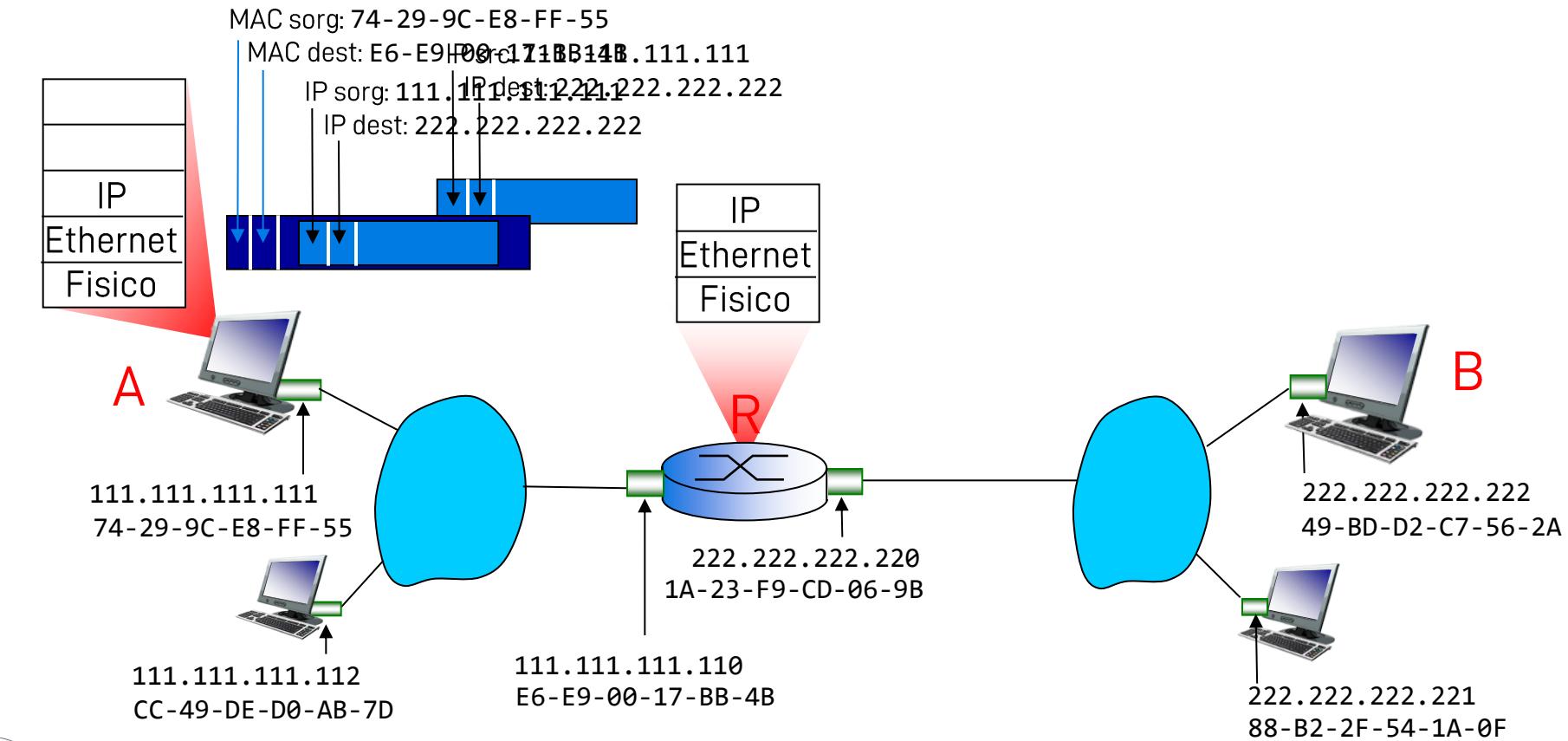
# Procedura di routing passo a passo

- A crea il datagramma IP con sorgente A e destinazione B
- Lo incapsula in un frame di livello 2, con l'indirizzo MAC di **R** come destinazione
  - ❖ Quindi il payload del frame è il datagramma IP
  - ❖ Assumiamo ARP già fatto



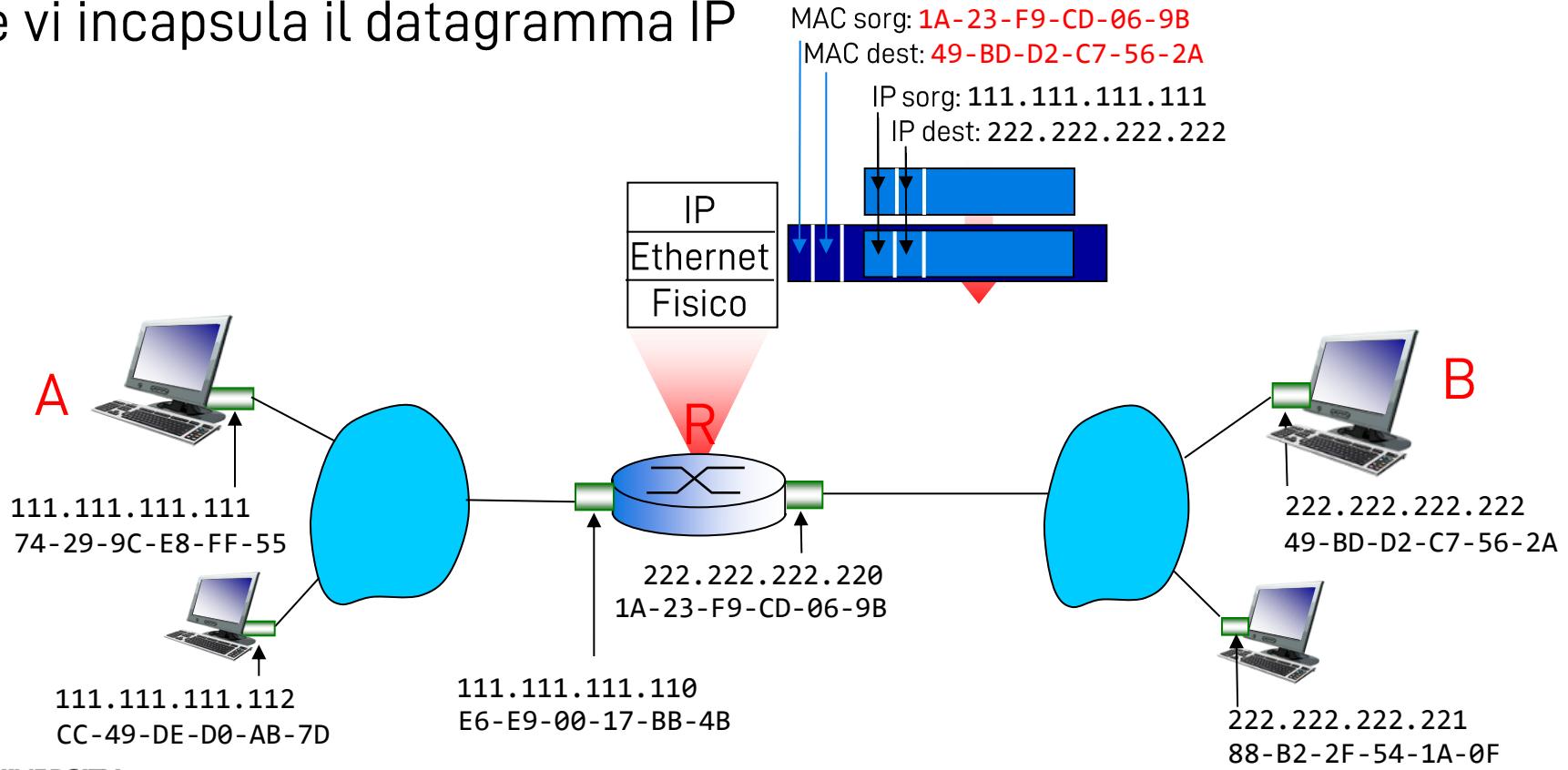
# Procedura di routing passo a passo

- Frame inviato da A a R
- Frame ricevuto ad R, datagramma estratto e passato a IP



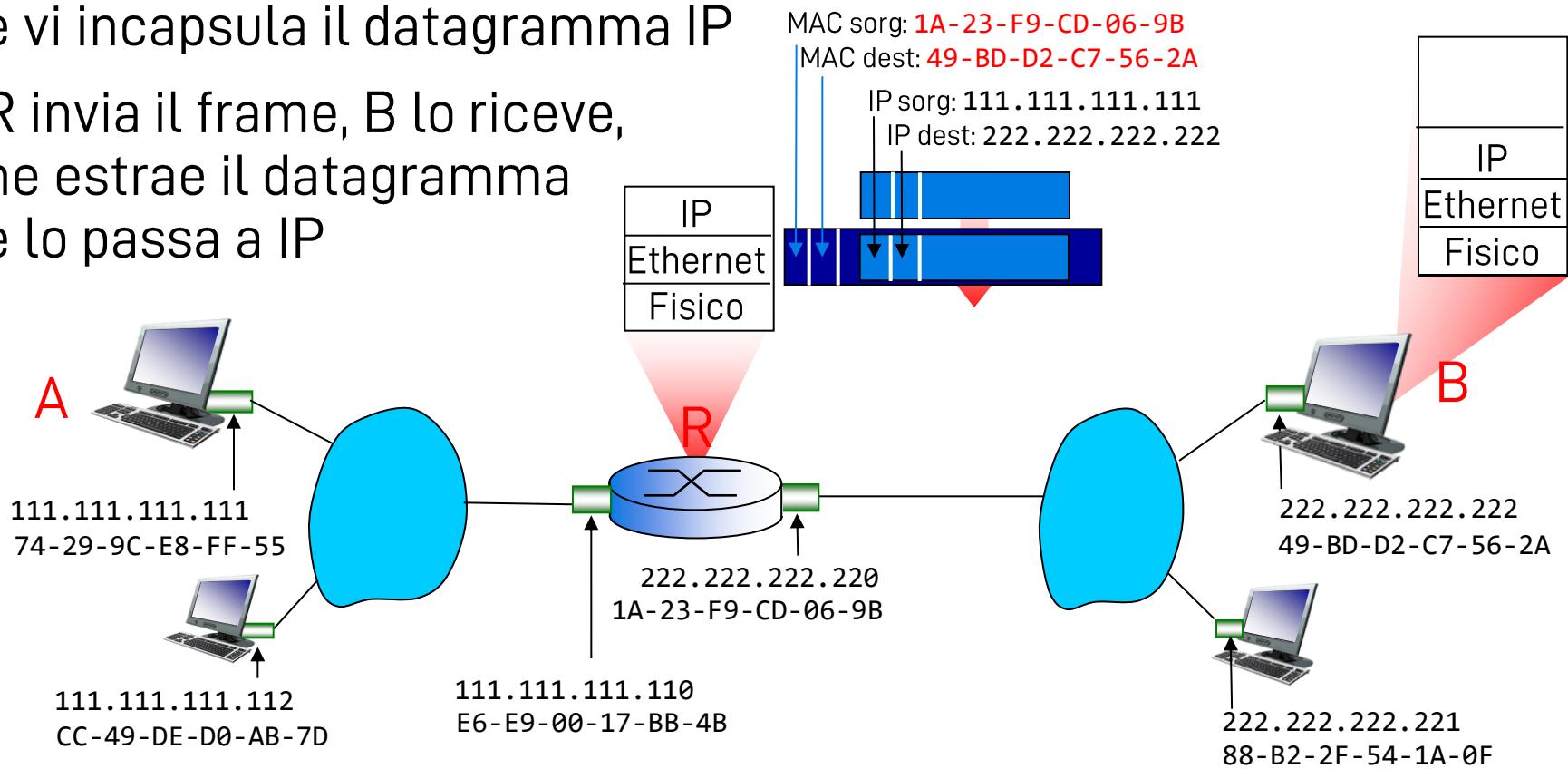
# Procedura di routing passo a passo

- R decide come inoltrare il datagramma con sorg. A e dest. B
- R crea un frame di livello 2 con destinazione l'indirizzo MAC di B e vi incapsula il datagramma IP



# Procedura di routing passo a passo

- R decide come inoltrare il datagramma con sorg. A e dest. B
- R crea un frame di livello 2 con destinazione l'indirizzo MAC di B e vi incapsula il datagramma IP
- R invia il frame, B lo riceve, ne estrae il datagramma e lo passa a IP



# Sommario

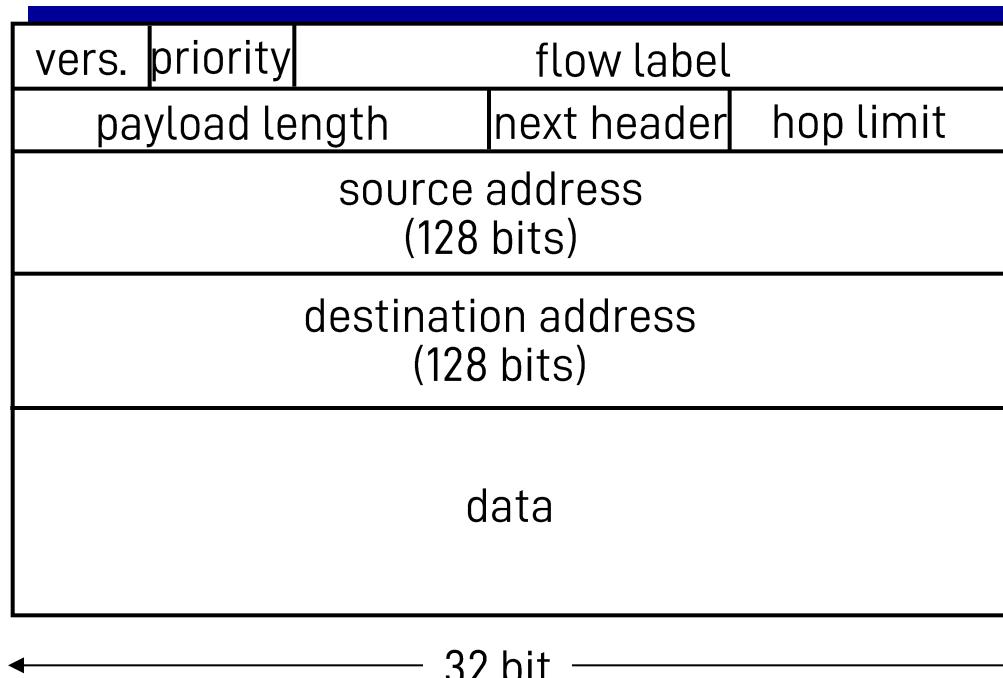
- Visione d'insieme
  - ❖ Data plane e control plane
- Come è fatto un router
- Il protocollo IP
  - ❖ Formato datagrammi
  - ❖ Frammentazione
  - ❖ Indirizzamento e NAT
  - ❖ Indirizzamento classless
  - ❖ Tipi di indirizzi
  - ❖ Address Resolution Protocol (ARP)
  - ❖ Internet Control Message Protocol (ICMP)
- Il protocollo IP (segue)
  - ❖ Dynamic Host Configuration Protocol (DHCP)
  - ❖ Il viaggio di un pacchetto
  - ❖ IPv6
- Protocolli di instradamento
  - ❖ Link state: Dijkstra
  - ❖ Internet, AS, OSPF
  - ❖ Distance vector: Bellman-Ford
  - ❖ RIP
- BGP

# Ragioni dietro al design IPv6

- Inizialmente: ampliare la quantità di indirizzi disponibili prima che quelli di IPv4 finissero
- Ragioni aggiuntive
  - ❖ Il formato dell'header velocizza l'elaborazione dei datagrammi
  - ❖ Facilitare la gestione della qualità del servizio
- Formato del datagramma IPv6:
  - ❖ Lunghezza dell'header fissata ad esattamente 40 byte
  - ❖ Frammentazione proibita

# Formato del datagramma IPv6

- Flow label: etichetta per tutti i datagrammi nello stesso “flusso”
  - ❖ Il concetto di flusso non è ben definito
- Priority: identifica la priorità dei datagrammi che fanno parte dello stesso flusso
- Next header: identifica il protocollo di livello 4 encapsulato nei dati

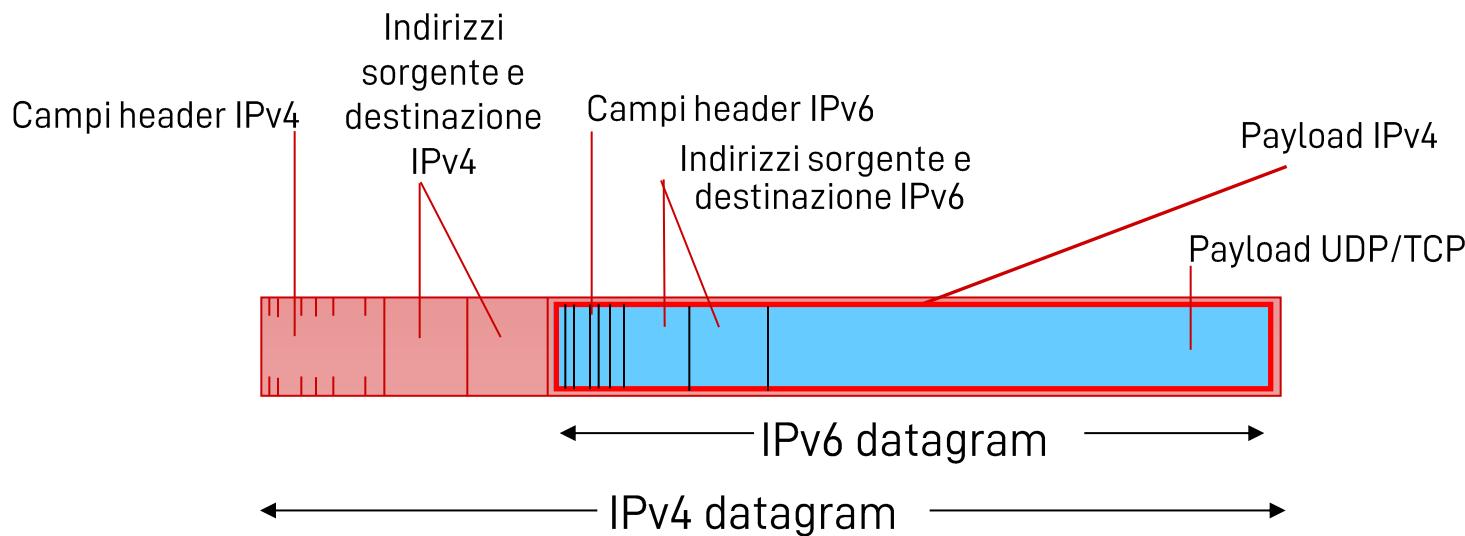


# Altri cambiamenti rispetto a IPv4

- Checksum: rimosso!
- Options: rimosse dall'header, ma consentite fuori dall'header (indicando un valore apposito nel campo "Next Header")
- ICMPv6
  - ❖ Messaggi aggiuntivi di errore, ad esempio "Packet Too Big"
  - ❖ Funzioni di gestione per i gruppi multicast

# Transizione da IPv4 a IPv6

- Non tutti i router possono essere aggiornati simultaneamente
- Come far coesistere IPv6 e IPv4 finché la transizione non è completa?
- Tunneling: i datagrammi IPv6 viaggiano come dati *incapsulati* dentro datagrammi IPv4

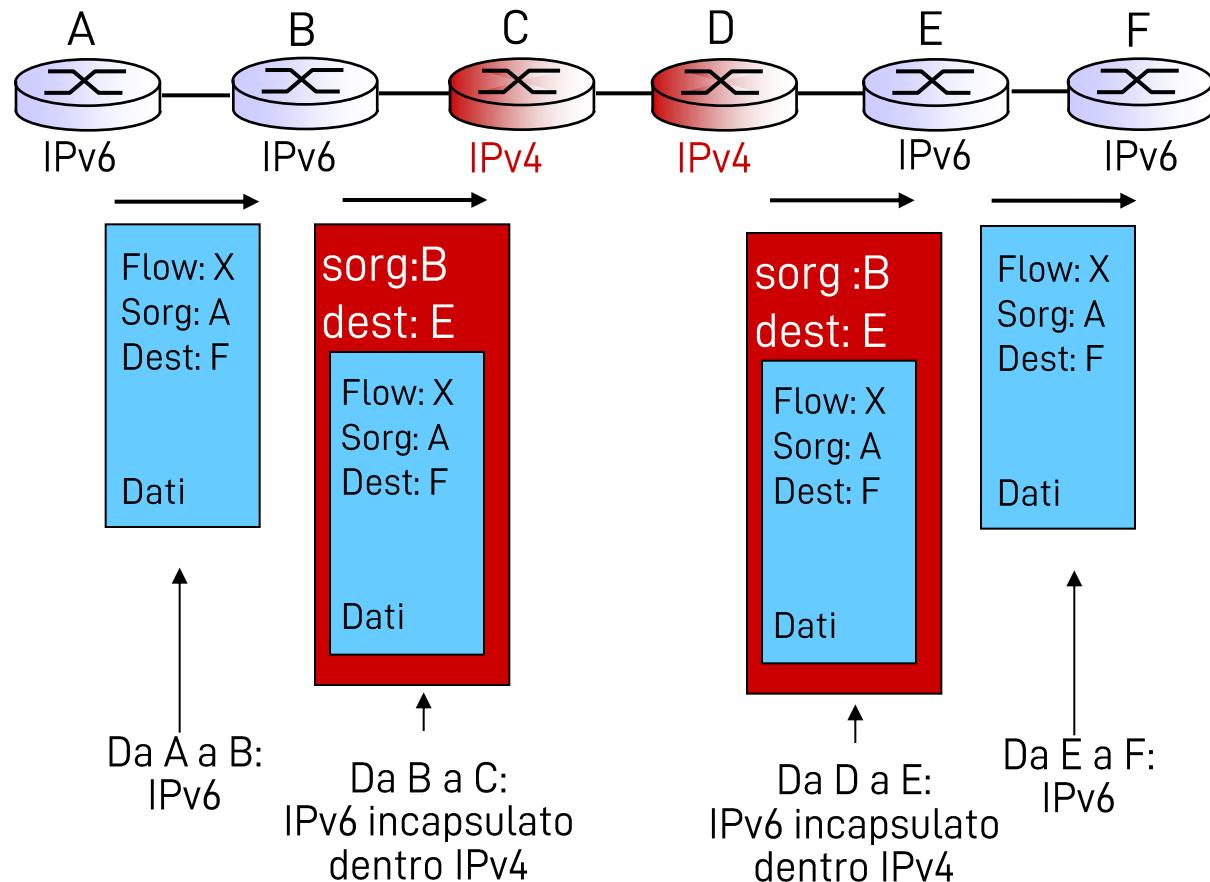


# Tunneling tramite IPv4

Vista logica:



Vista fisica:



# Indirizzi IPv6

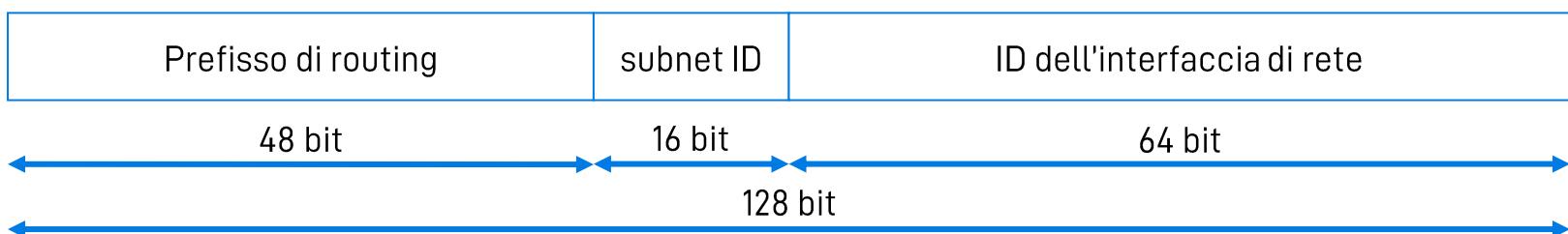
- Lunghezza = 128 bit  $\sim 3.4 \times 10^{38}$  indirizzi
- A differenza di IPv4, si scrivono in notazione esadecimale
  - ❖ Ogni gruppo di 4 bit è scritto come una cifra 0, 1, 2, ..., 9, a, b, ..., f
  - ❖ 32 cifre esadecimali per ogni indirizzo, in 8 gruppi da 4 cifre
- Esempio: 2a03:2880:f108: ...
  - ❖ 2a03:
    - 0010 (2)
    - 1010 (a)
    - 0000 (0)
    - 0011 (3)

# Indirizzi IPv6

- Un indirizzo completo può essere qualcosa del genere
  - ❖ 2a03:2880:f108:0083:face:b00c:0000:25de
- Per accorciarlo, si omettono gli zeri all'inizio di ogni campo, e si scrivono gli zeri consecutivi con un solo "0"
  - ❖ 2a03:2880:f108:0083:face:b00c:0000:25de diventa
  - ❖ 2a03:2880:f108:83:face:b00c:0:25de
- I gruppi consecutivi di zeri si rappresentano con un ::
  - ❖ 2a03:2880:f108:0000:0000:0000:0000:25de diventa
  - ❖ 2a03:2880:f108::25de
- Ma il :: si può usare solo una volta
  - ❖ Se ci sono più gruppi di zeri, lo si usa sul gruppo più lungo (RFC 5952)

# Prefissi

- I prefissi e le sottoreti funzionano esattamente come nel CIDR di IPv4, inclusa la / per indicare i bit a 1 nella maschera:
  - ❖ IPv4: 192.168.0.0/24
    - Da 192.168.0.0
    - A 192.168.0.255
  - ❖ IPv6: 2a03:2880:f108:83::/64
    - Da 2a03:2880:f108:83:0:0:0:0
    - A 2a03:2880:f108:83:ffff:ffff:ffff:ffff
- Tipica composizione di un indirizzo:



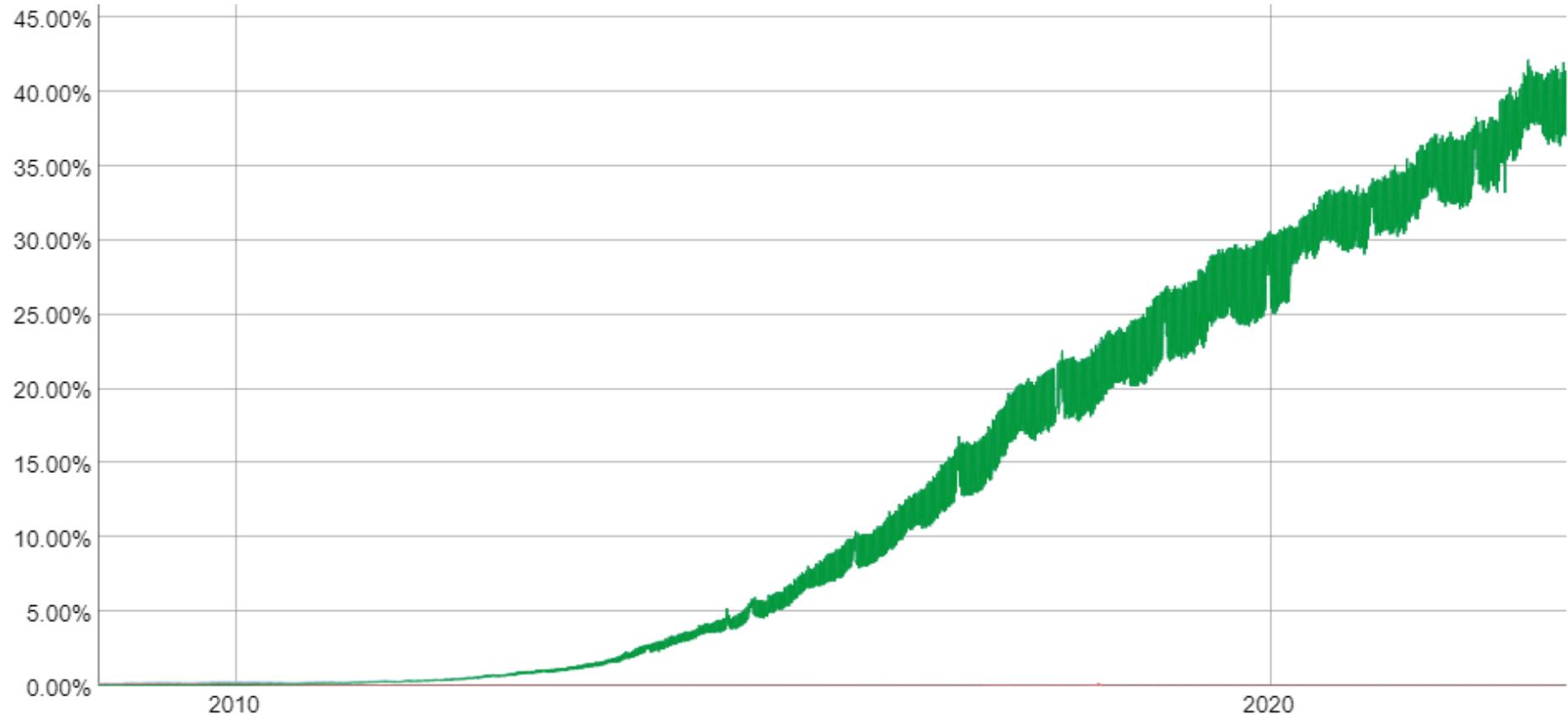
# Indirizzi speciali

- Non specificato, o “questo computer” → `::/128` (tutti 0)
- Loopback (localhost) → `::1/128` (tutti 0 con un 1 alla fine)
- Indirizzo IPv4 mappato su IPv6 → `::ffff:0:0/96`
  - ❖ Quindi `::ffff:xxyy:zzww`, dove `xx.yy.zz.ww` sono i bit dell’indirizzo IPv4, espresso in esadecimale
  - ❖ Es: IPv4 = **193.175.55.16** = **c1.af.37.10**
  - ❖ IPv6 = `::ffff:c1af:3710`
- Multicast: `ff00::/8`
- Link-local unicast: `fe80::/10`
  - ❖ **169.254.0.0/16** in IPv4
  - ❖ Per consentire il networking di rete locale senza router o DHCP

# Indirizzi “molto speciali”

- L'introduzione delle lettere esadecimali (da a ad f)
  - ❖ IPv6 di un server Facebook
    - 2a03:2880:f108:83:**face:b00c::25de**
  - ❖ E potete sbizzarrirvi con la fantasia
    - 2001:db8:a0b:12f0:0:**d0d0:i5:dead**
    - 2001:db8:a0b:12f0:0:**dead:beef:ca1f**
    - 2001:db8:a0b:12f0:0:**1ce:1ce:babe**
    - 2001:db8:a0b:12f0:**1:5ee:bad:c0de**

# Adozione di IPv6 nel mondo



# Parte 2

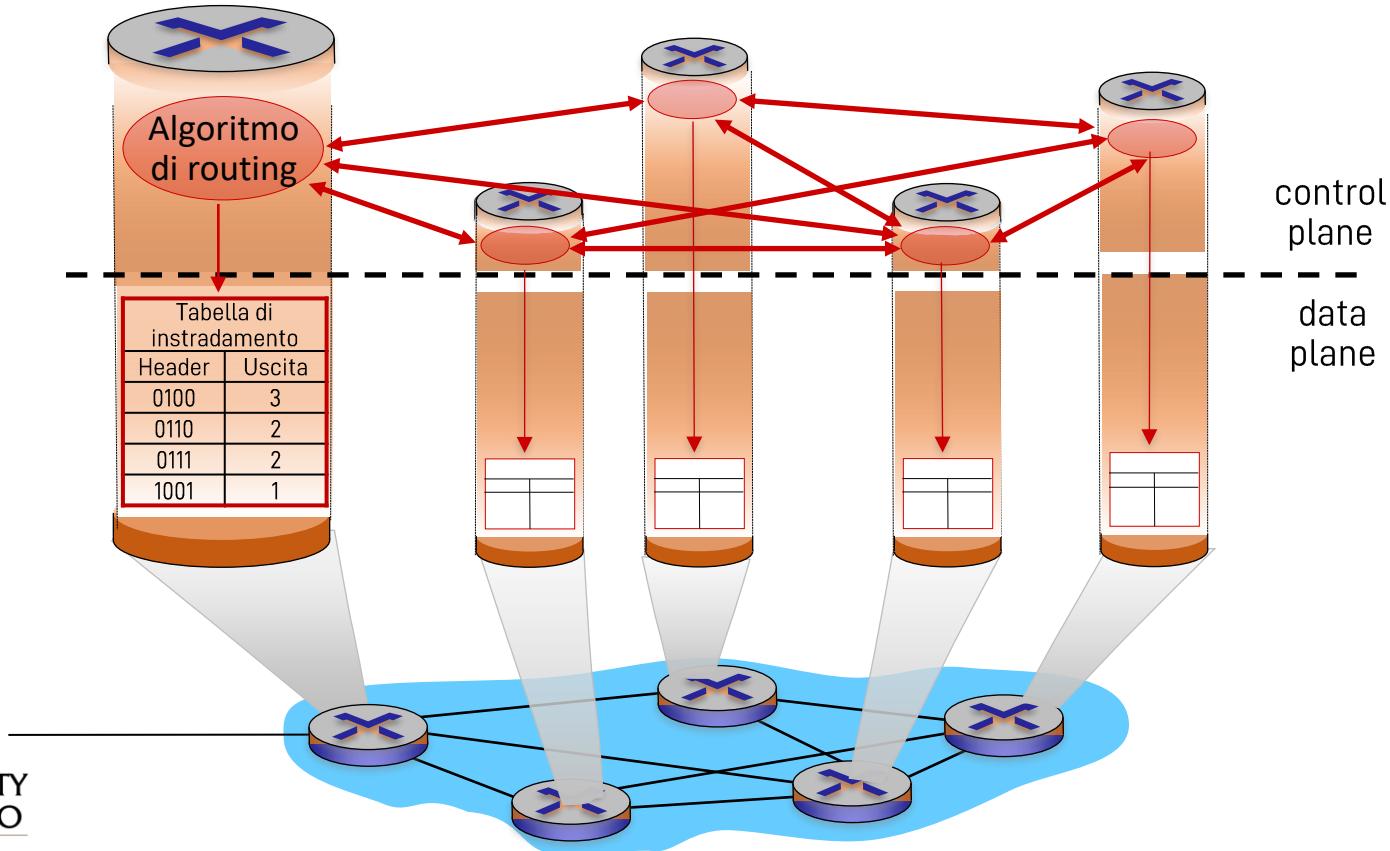
Metodi e protocolli di instradamento

# Sommario

- Visione d'insieme
  - ❖ Data plane e control plane
- Come è fatto un router
- Il protocollo IP
  - ❖ Formato datagrammi
  - ❖ Frammentazione
  - ❖ Indirizzamento e NAT
  - ❖ Indirizzamento classless
  - ❖ Tipi di indirizzi
  - ❖ Address Resolution Protocol (ARP)
  - ❖ Internet Control Message Protocol (ICMP)
- Il protocollo IP (segue)
  - ❖ Dynamic Host Configuration Protocol (DHCP)
  - ❖ Il viaggio di un pacchetto
  - ❖ IPv6
- **Protocolli di instradamento**
  - ❖ Link state: Dijkstra
  - ❖ Internet, AS, OSPF
  - ❖ Distance vector: Bellman-Ford
  - ❖ RIP
- BGP

# Finora

- Reti, indirizzi IP e router erano già pronti e configurati
  - ❖ La tabella di inoltro era fornita da “qualcosa”
- Ora vediamo come si crea una tabella di routing

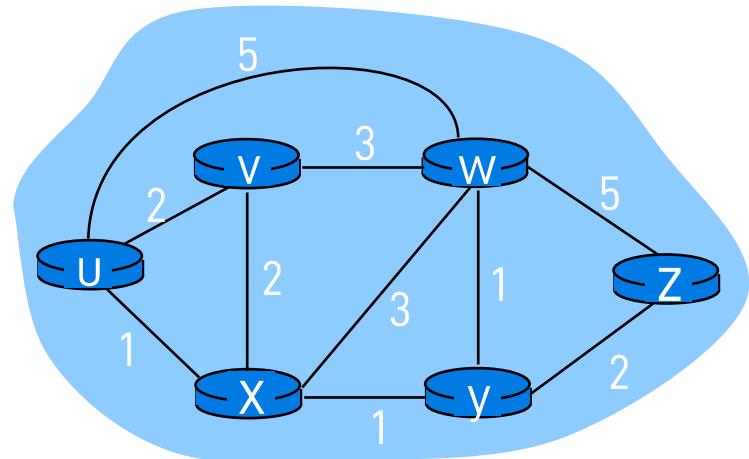


# Protocolli di instradamento ("routing", appunto)

- Obiettivo: determinare "buoni" percorsi da un mittente a un destinatario, attraverso una rete di router
  - ❖ Percorso: sequenza di router attraversati da un datagramma per giungere all'host di destinazione
- "Buon percorso": definito secondo qualche metrica
  - ❖ Costo più basso
  - ❖ Più veloce
  - ❖ Meno congestionato
- Instradamento: un'altra challenge tra le top-10!

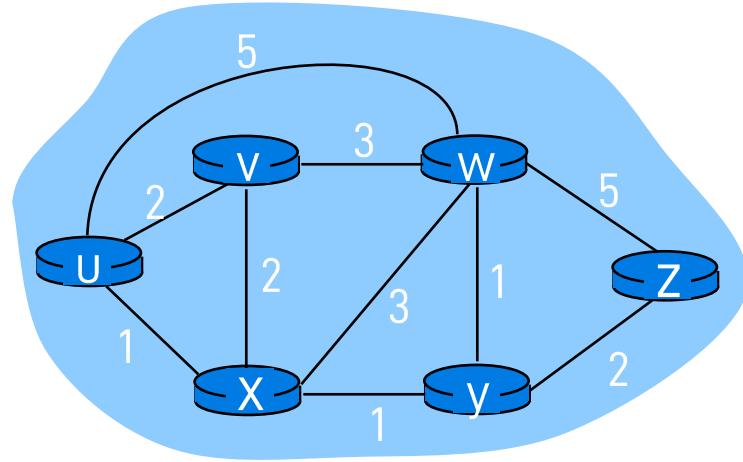
# Modello a grafo della rete

- Grafo:  $G = (N, E)$
- $N$  = insieme di nodi (router)
  - ❖  $N = \{u, v, w, x, y, z\}$
- $E$  = insieme di link
  - ❖  $E = \{(u,v), (u,x), (v,x), (v,w), (x,w), (x,y), (w,y), (w,z), (y,z)\}$



# Modello a grafo della rete: costi

- $c(x, x') = \text{costo del link } (x, x')$ 
  - ❖  $c(w, z) = 5$
- Può essere qualunque cosa
  - ❖ Numero di salti
  - ❖ Banda del link (proporzionale a \$\$)
  - ❖ Inverso della banda del link
  - ❖ Congestione
  - ❖ Dipende dall'algoritmo usato
- Costo del percorso: somma del costo di tutti i link
  - ❖  $\text{cost}(x_1, x_2, \dots, x_p) = c(x_1, x_2) + c(x_2, x_3) + \dots + c(x_{p-1}, x_p)$
- Qual è il percorso di costo minimo?
  - ❖ Compito dell'algoritmo di routing



# Tipi di algoritmi di routing

## Informazioni globali o distribuite?

### □ Globali:

- ❖ Tutti i router conoscono la topologia della rete ed i costi dei link
- ❖ Algoritmi a "link state"

### □ Distribuite:

- ❖ I router conoscono gli altri router cui sono collegati, e il costo dei link verso di essi
- ❖ Processo iterativo di calcolo percorsi e scambio informazioni con i router vicini
- ❖ Algoritmi a "distance vector"

## Statico o dinamico?

### □ Statico:

- ❖ I percorsi cambiano molto poco nel tempo

### □ Dinamico:

- ❖ Le rotte cambiano più frequentemente
  - Aggiornamenti periodici
  - Tipicamente a causa di cambiamenti nel costo dei link

# Sommario

- Visione d'insieme
  - ❖ Data plane e control plane
- Com'è fatto un router
- Il protocollo IP
  - ❖ Formato datagrammi
  - ❖ Frammentazione
  - ❖ Indirizzamento e NAT
  - ❖ Indirizzamento classless
  - ❖ Tipi di indirizzi
  - ❖ Address Resolution Protocol (ARP)
  - ❖ Internet Control Message Protocol (ICMP)
- Il protocollo IP (segue)
  - ❖ Dynamic Host Configuration Protocol (DHCP)
  - ❖ Il viaggio di un pacchetto
  - ❖ IPv6
- Protocolli di instradamento
  - ❖ Link state: Dijkstra
  - ❖ Internet, AS, OSPF
  - ❖ Distance vector: Bellman-Ford
  - ❖ RIP
- BGP

# Algoritmo di Dijkstra (link state)

- Assunzione: topologia di rete e costi dei link noti a tutti i nodi
  - ❖ Si ottiene circolando le informazioni sullo stato dei link (in broadcast a tutti i nodi)
  - ❖ Tutti i nodi hanno le stesse info
- Output: il percorso a costo minimo da un nodo a tutti gli altri nodi
  - ❖ E' la tabella di inoltro!
- Iterativo: dopo  $k$  iterazioni, si conosce il percorso a costo minimo verso almeno  $k$  destinazioni

Notazione:

- $c(x, y)$ : costo del link da  $x$  a  $y$ 
  - ❖  $= \infty$  se  $x$  e  $y$  non sono collegati
- $D(v)$ : costo del percorso verso  $v$
- $p(v)$ : predecessore di  $v$  lungo il cammino dalla sorgente a  $v$
- $N'$ : insieme di nodi per cui il cammino a costo minimo è stato già determinato

# Algoritmo di Dijkstra (link state)

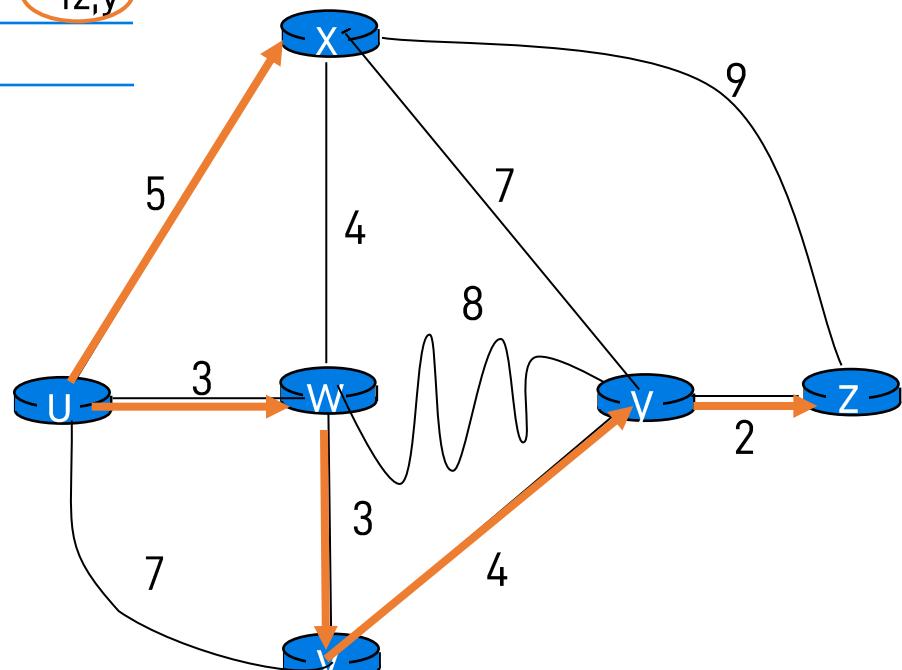
1. Inizializzazione:
2.  $N' = \{u\}$  //nodo corrente
3. for all nodes v
4.     if (v è adiacente a u)  
5.         then  $D(v) = c(u, v)$  e  $p(v) = u$
6.         else  $D(v) = \infty$
7. Loop
8. Trova w non contenuto in  $N'$  tale che  $D(w)$  è minimo
9. Aggiungi w a  $N'$
10. Aggiorna  $D(v)$  per tutti i nodi v adiacenti a w e non contenuti in  $N'$  :
  11.     if  $D(w) + c(w, v) < D(v)$  then
  12.          $D(v) = D(w) + c(w, v)$
  13.          $p(v) = w$
  14.     end if
  15.     // Il nuovo costo verso v è il costo già noto, oppure il costo minimo verso w più il costo da w a v
  16. until (Tutti i nodi sono contenuti in  $N'$ )

# Algoritmo di Dijkstra: esempio

Passo	N'	D(v)	D(w)	D(x)	D(y)	D(z)
		p(v)	p(w)	p(x)	p(y)	p(z)
0	U	7,u	3,u	5,u	$\infty$	$\infty$
1	UW	6,w	5,u	11,w	$\infty$	
2	UWX	6,w		11,w	14,x	
3	UWXV			10,v	14,x	
4	UWXVY				12,y	
5	UWXVYZ					

Note:

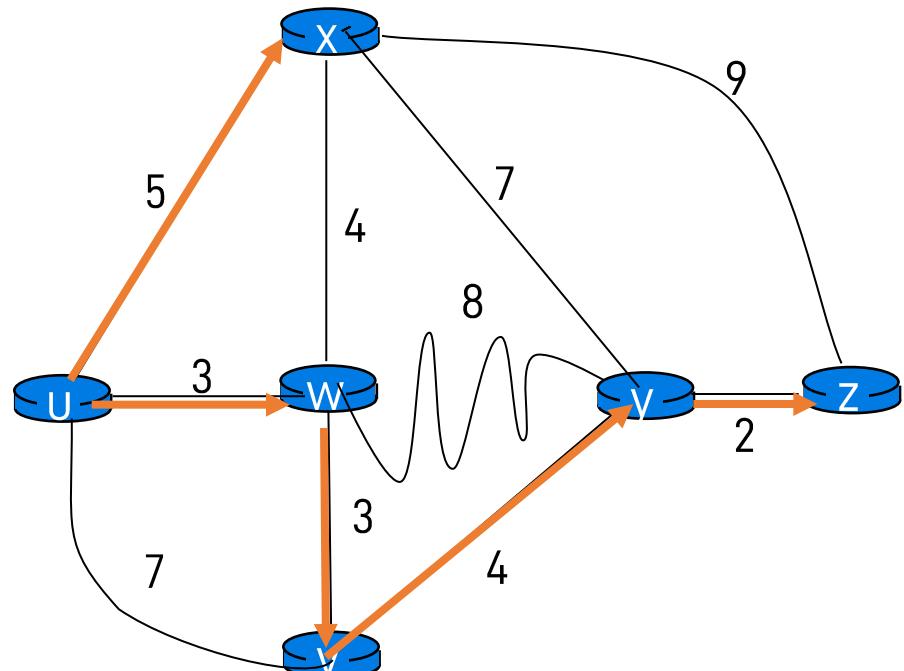
- Costruisce i percorsi minimi tracciando a ritroso i nodi predecessori
- Possono esserci percorsi dal costo uguale (si decide arbitrariamente quale tenere)



# Tabella di inoltro risultante

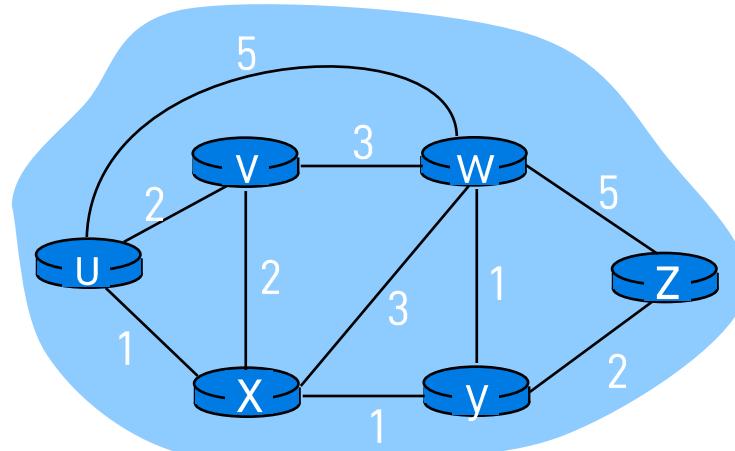
- Tabella di inoltro per il nodo u

Destinazione	Inoltra a (“next hop”)
v	w
w	w
x	x
y	w
z	w



# Algoritmo di Dijkstra: altro esempio

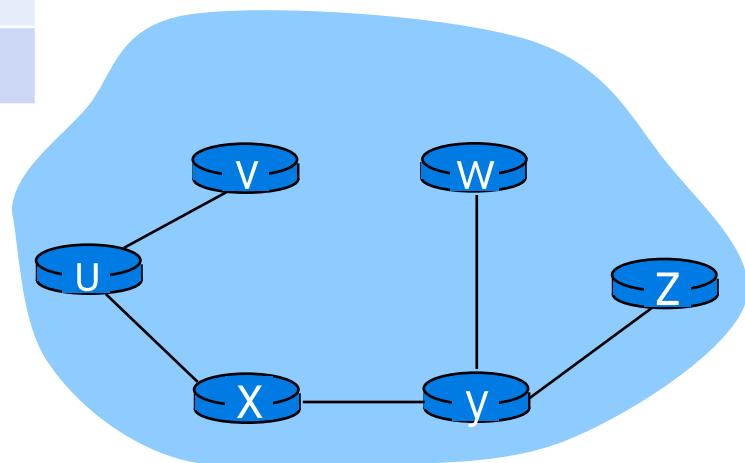
Step	$N'$	$D(v), p(v)$	$D(w), p(w)$	$D(x), p(x)$	$D(y), p(y)$	$D(z), p(z)$
0	U	2, u	5, u	1, u	$\infty$	$\infty$
1	UX	2, u	4, x		2, x	$\infty$
2	UXy	2, u	3, y			4, y
3	UXyv		3, y			4, y
4	UXyvw					4, y
5	UXyvwz					



# Tabella di inoltro risultante

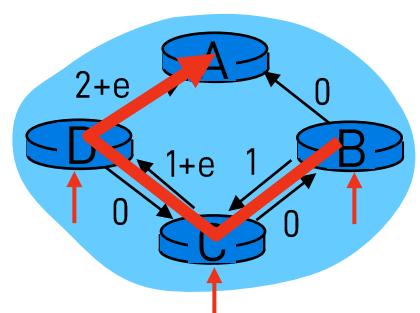
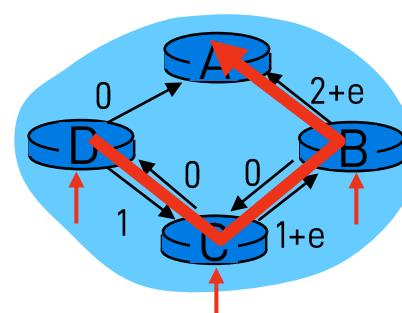
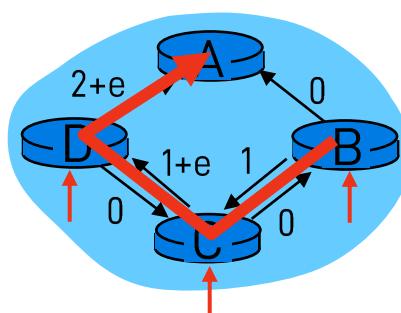
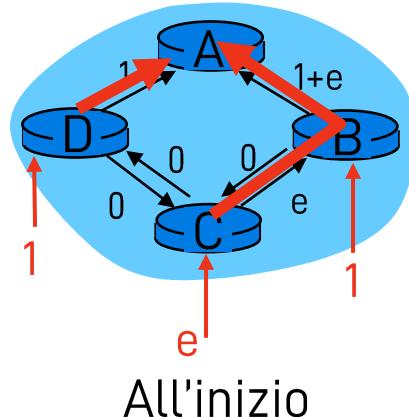
- Tabella di inoltro per il nodo u

Destinazione	Inoltra a ("next hop")
v	v
w	x
x	x
y	x
z	x



# Algoritmo di Dijkstra: discussione

- Complessità dell'algoritmo con una rete di  $n$  nodi
  - ❖ Ad ogni iterazione bisogna controllare tutti i nodi  $w \notin N'$
  - ❖  $n(n+1)/2$  confronti:  $O(n^2)$
  - ❖ Ci sono implementazioni più efficienti:  $O(n * \log(n))$
- L'algoritmo può oscillare se definite male i costi
  - ❖ Esempio: costo = quantità di traffico trasportata dal link

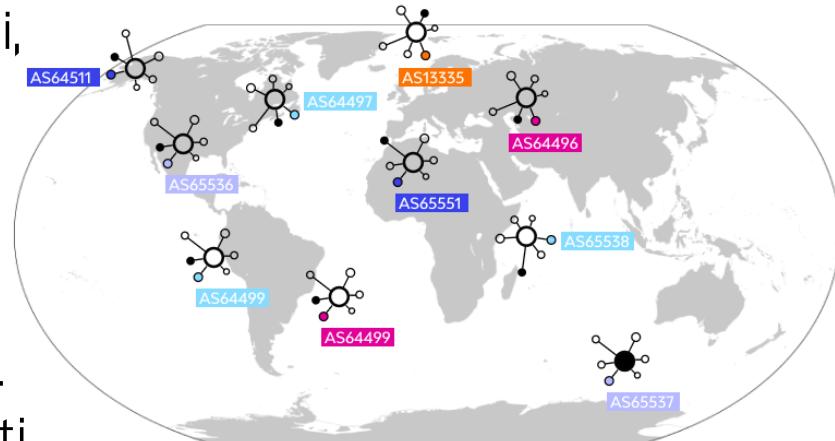


# Sommario

- Visione d'insieme
  - ❖ Data plane e control plane
- Come è fatto un router
- Il protocollo IP
  - ❖ Formato datagrammi
  - ❖ Frammentazione
  - ❖ Indirizzamento e NAT
  - ❖ Indirizzamento classless
  - ❖ Tipi di indirizzi
  - ❖ Address Resolution Protocol (ARP)
  - ❖ Internet Control Message Protocol (ICMP)
- Il protocollo IP (segue)
  - ❖ Dynamic Host Configuration Protocol (DHCP)
  - ❖ Il viaggio di un pacchetto
  - ❖ IPv6
- **Protocolli di instradamento**
  - ❖ Link state: Dijkstra
  - ❖ **Internet, AS, OSPF**
  - ❖ Distance vector: Bellman-Ford
  - ❖ RIP
- BGP

# In pratica: Internet e gli AS

- Internet è formata da moltissime sottoreti, ognuna di proprietà di qualche entità (ISP, operatori di rete, aziende, ecc.)
- Idealmente, ogni rete dovrebbe essere
  - ❖ Amministrativamente autonoma
    - Algoritmo di routing, configurazione...
  - ❖ Capace di collegarsi a tutte le altre reti
- Soluzione: organizzare i router in “autonomous systems” (ASs)
  - ❖ Definizione: un AS è un gruppo di router sotto lo stesso controllo amministrativo
  - ❖ Ogni AS è identificato da un numero (RFC 1930), ed i numeri di AS sono assegnati centralmente dai registri regionali ICANN
  - ❖ **Intra-AS** routing vs. **Inter-AS** routing
  - ❖ Nel seguito vedremo protocolli per intra-AS e inter-AS routing

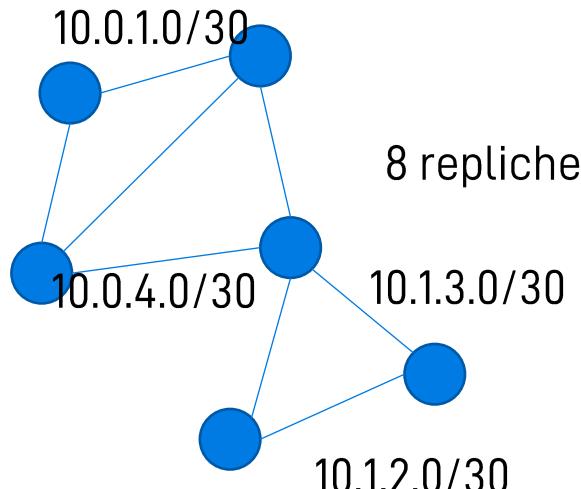


# Intra-AS: Open Shortest Path First

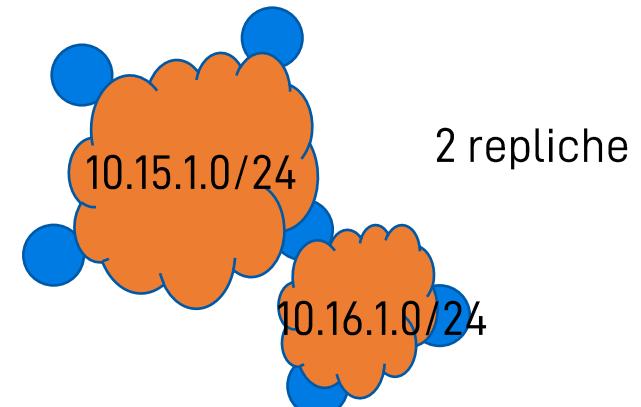
- OSPF, protocollo di routing basato su link state
  - ❖ Dissemina pacchetti con lo stato dei link
  - ❖ Assunzione: topologia nota a tutti i router
  - ❖ Calcola i percorsi usando l'algoritmo di Dijkstra
- Usa datagrammi IP
  - ❖ Non si serve di nessun livello di trasporto
  - ❖ Invia i pacchetti di aggiornamento dei link all'indirizzo multicast 224.0.0.5
- Implementa tre procedure
  - ❖ Protocollo di "Hello"
    - Messaggi di mantenimento: controllano i link funzionanti, e quindi verificano quali altri nodi sono vicini
  - ❖ Protocollo di "Exchange"
    - Usato per informare i vicini che si sono appena "conosciuti" sulla topologia della rete nota al momento
  - ❖ Protocollo di "Flooding"
    - Informa tutti i router di un cambio nello stato dei link

# Flooding controllato

- Invia messaggi ricevuti su un'interfaccia a tutte le altre interfacce
- Se esistono solo connessioni dirette (reti "/30") ad altri router
  - ❖ Un messaggio per link
- In reti con un dominio di broadcast
  - ❖ Un messaggio per dominio di broadcast

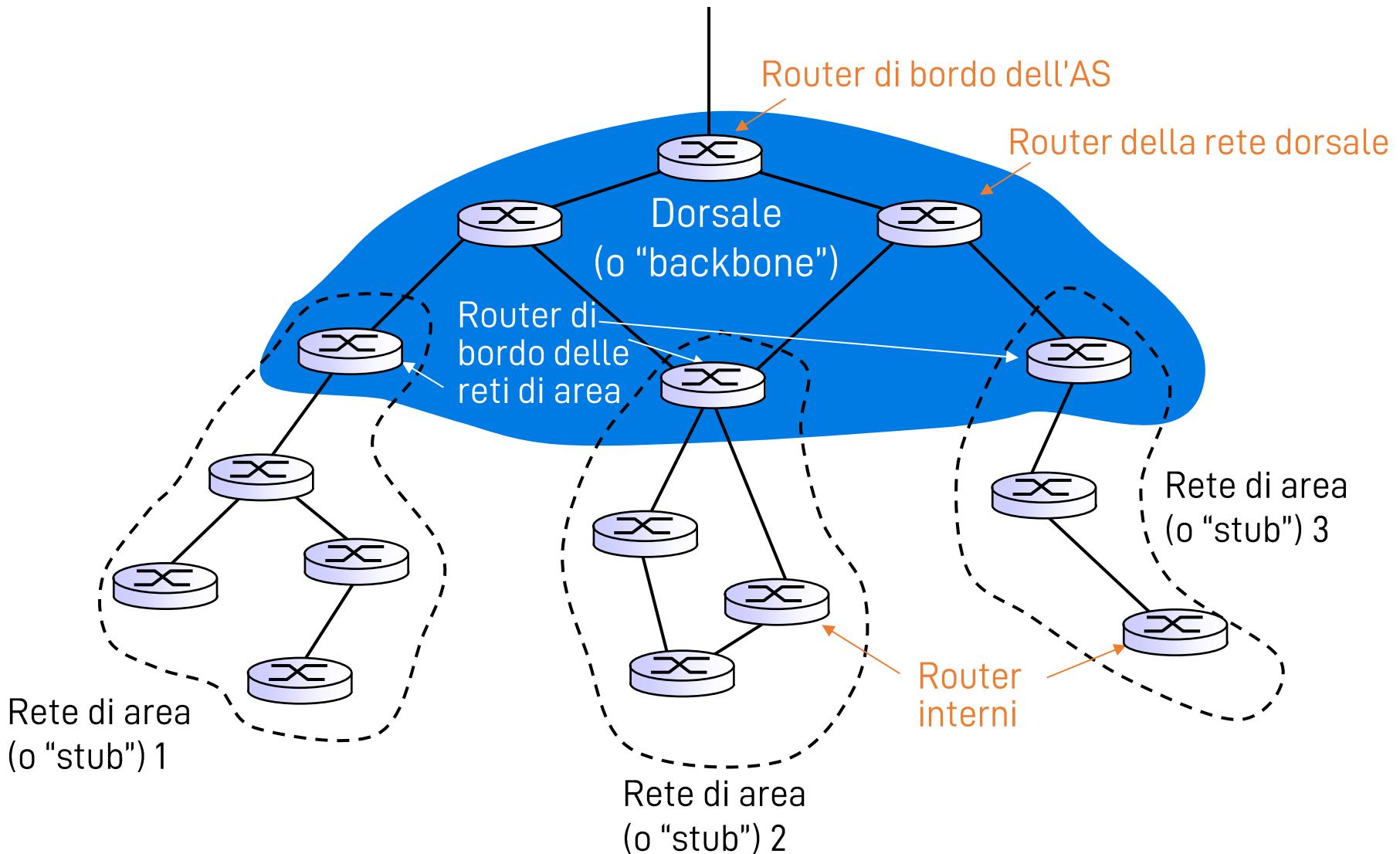


8 repliche



2 repliche

# OSPF gerarchico



# OSPF gerarchico

- ❑ Usato in reti con molti router
- ❑ Gerarchia a due livelli: dorsale ("backbone") e reti di area
  - ❖ I messaggi con i link state circolano solo nelle reti di area
  - ❖ I nodi conoscono la topologia di rete solo all'interno della propria area, e uno shortest path verso le altre aree
- ❑ I router di bordo di ciascuna area "riassumono" la distanza verso le reti che fanno parte della propria area agli altri router
- ❑ La dorsale conta come un'area
  - ❖ I router della dorsale fanno girare OSPF solo per la dorsale
- ❑ I router di bordo connettono ad altri AS
  - ❖ Usando il protocollo BGP, che vedremo

# Ma l'input sono proprio i costi?

- ❑ OSPF è un protocollo a link state
  - ❖ L'unica cosa che influenza la scelta dei percorsi è il costo dei link
- ❑ Molti operatori potrebbero "agire" sul costo dei link per ottenere un certo controllo sul traffico che passa per la loro rete
  - ❖ Esempio: se so che il traffico che entra nella mia rete da un certo router è quasi tutto destinato ad un'altra rete specifica, faccio in modo che OSPF scelga il percorso che desidero io tra i due punti
- ❑ Inverte la relazione causa-effetto del protocollo
  - ❖ Dato un obiettivo di "traffic engineering", agisco sul costo dei link per ottenerlo
  - ❖ Conseguenze →



# Sommario

- Visione d'insieme
  - ❖ Data plane e control plane
- Come è fatto un router
- Il protocollo IP
  - ❖ Formato datagrammi
  - ❖ Frammentazione
  - ❖ Indirizzamento e NAT
  - ❖ Indirizzamento classless
  - ❖ Tipi di indirizzi
  - ❖ Address Resolution Protocol (ARP)
  - ❖ Internet Control Message Protocol (ICMP)
- Il protocollo IP (segue)
  - ❖ Dynamic Host Configuration Protocol (DHCP)
  - ❖ Il viaggio di un pacchetto
  - ❖ IPv6
- **Protocolli di instradamento**
  - ❖ Link state: Dijkstra
  - ❖ Internet, AS, OSPF
  - ❖ **Distance vector: Bellman-Ford**
  - ❖ RIP
- BGP

# Routing con “distance vector”

- Algoritmo distribuito
- Non richiede conoscenza della topologia di rete
- Richiede che ogni nodo di rete conosca
  - ❖ I propri vicini
  - ❖ Il costo dei link verso questi vicini
- La presenza di altri nodi posizionati oltre i propri vicini viene notificata con dei messaggi
- Algoritmo di base usato per questo tipo di routing:  
Bellman-Ford

# Algoritmo di Bellman-Ford

- Consideriamo i router
  - ❖ Sono loro a far girare l'algoritmo
  - ❖ Ogni router conosce la rete connessa alle proprie interface e i NetID
  - ❖ Nel seguito, usiamo gli ID dei router per indicare le destinazioni, ma in realtà le tabelle riportano indirizzi IP
- Notazione
  - ❖ N: insieme di vicini  $\rightarrow N_x$  è l'insieme di vicini del router x
  - ❖ R: tabella di inoltro  $\rightarrow R_x$  è la tabella di inoltro del router x
    - $R[d]$ : riga della tabella di inoltro per la destinazione d
      - $R[d].cost$ : costo per raggiungere d
      - $R[d].nexthop$ : nodo cui inoltrare il pacchetto per procedere verso d
      - $R[d].time$ : time al quale il percorso è stato impostato, usato per invalidare percorsi troppo vecchi
  - ❖ D: vettore con tutte le distanze  $\rightarrow D_x$  è il distance vector del router x
    - $D_x = [(d, R_x[d].cost) \text{ tale che } d \in R_x]$
  - ❖ I costi dei link si chiamano anche "distanze" o "metriche"

# Algoritmo di Bellman-Ford (nodo x)

1. Inizializzazione:
2. Per tutti i vicini  $n \in N_x$ :  
 $R_x[n].cost = c(x, n)$ ,  $R_x[n].nexthop = n$ ,  $R_x[n].time = \text{adesso}$
4. Ogni T secondi:
  5. invia  $D_x = [(d, R_x[d].cost) \text{ tale che } d \in R_x]$  a tutti i vicini in  $N_x$
  6. Quando si riceve un vettore  $D_y$  dal vicino  $y$ :
    7. Per ogni  $(d, c) \in D_y$ :
      8. if  $d \notin R$  or  $c + c(x, y) < R[d].cost$  or  $y = R[d].nexthop$ :
        9.  $R[d].cost = c + c(x, y)$
        10.  $R[d].nexthop = y$
        11.  $R[d].time = \text{adesso}$

# Distance vector in the book

- Versione reattiva ai cambiamenti, tiene una copia di tutti i DV ricevuti dai vicini

1. Inizializzazione:

2. Per tutte le destinazioni  $y \rightarrow D_x(y) = c(x, y)$

// $c(x, y) = \infty$  se  $x, y$  non sono vicini ,  $c(x, y) = 0$  se  $x = y$

3. Per tutti i vicini  $w$ , e tutte le destinazioni  $y \rightarrow D_w(y) = ?$

4. Per ogni vicino  $w \rightarrow$  invia  $D_x = [D_x(y) : y \text{ in } N_x]$  a  $w$

5. Loop:

6. Aspetto finche il costo verso un vicino non cambia o non ricevo  $D_w$  da  $w$

7. Per ogni destinazione  $y$ :

8.  $D_x(y) = \min_v \{ c(x, v) + D_v(y) \}$  // $v$  è nell'insieme dei vicini

9.  $\text{next-hop}_x(y) = \operatorname{argmin}_v \{ c(x, v) + D_v(y) \}$

10. if  $D_x(y)$  è cambiato per una qualsiasi destinazione  $y$ :

11. invia  $D_x = [D_x(y) : y \text{ destinazione nota a } x]$  a tutti i vicini

12. forever

# Bellman-Ford: esempio

Router X		
Dest	Next	Costo
Y	Y	2
Z	Z	7

Router X		
Dest	Next	Costo
Y	Y	2
Z	Y	3

Router Y		
Dest	Next	Costo
X	X	2
Z	Z	1

$$R_x[Y].cost + D_y[Z] (3) < R_x[Z].cost (7)$$

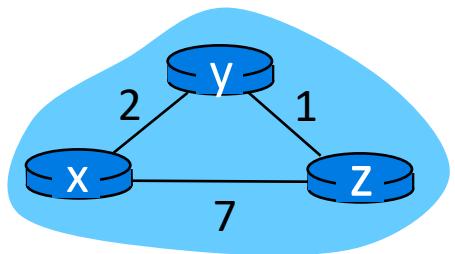
$$D_y = [(X, 2), (Z, 1)]$$

Router Z		
Dest	Next	Costo
X	X	7
Y	Y	1

Router Z		
Dest	Next	Costo
X	Y	3
Y	Y	1

$$R_z[Y].cost + D_y[X] (3) < R_z[X].cost (7)$$

time



# Bellman-Ford: esempio

Router X		
Dest	Next	Costo
Y	Y	2
Z	Z	7

Router X		
Dest	Next	Costo
Y	Y	2
Z	Y	3

Router Y		
Dest	Next	Costo
X	X	2
Z	Z	1

$$D_x = [(Y, 2), (Z, 3)]$$

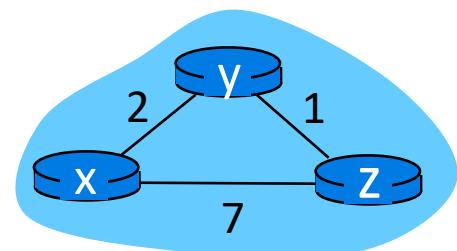
Router Y		
Dest	Next	Costo
X	X	2
Z	Z	1

Router Z		
Dest	Next	Costo
X	X	7
Y	Y	1

Router Z		
Dest	Next	Costo
X	Y	3
Y	Y	1

Router Z		
Dest	Next	Costo
X	Y	3
Y	Y	1

time



# Bellman-Ford: esempio

Router X		
Dest	Next	Costo
Y	Y	2
Z	Z	7

Router X		
Dest	Next	Costo
Y	Y	2
Z	Y	3

Router X		
Dest	Next	Costo
Y	Y	2
Z	Y	3

Router Y		
Dest	Next	Costo
X	X	2
Z	Z	1

Router Y		
Dest	Next	Costo
X	X	2
Z	Z	1

Router Y		
Dest	Next	Costo
X	X	2
Z	Z	1

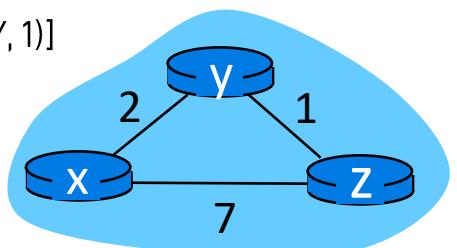
Router Z		
Dest	Next	Costo
X	X	7
Y	Y	1

Router Z		
Dest	Next	Costo
X	Y	3
Y	Y	1

Router Z		
Dest	Next	Costo
X	Y	3
Y	Y	1

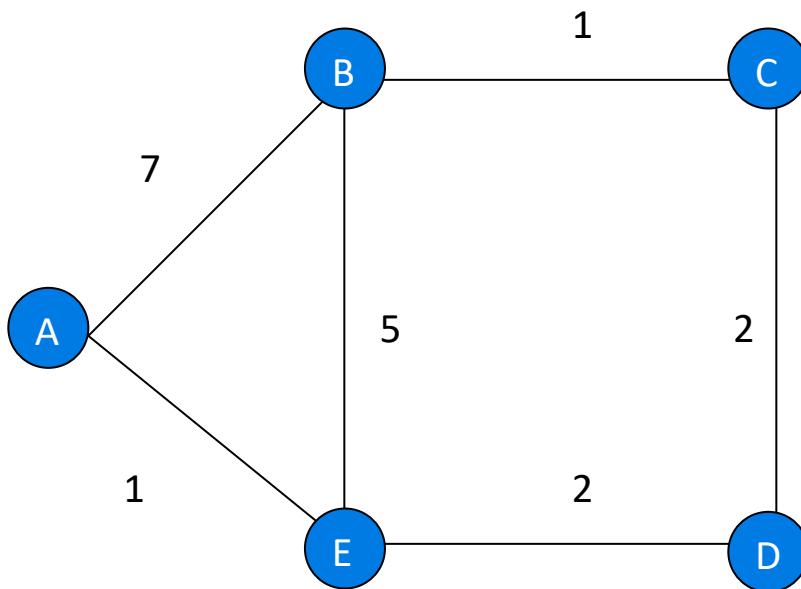
time

$$D_z = [(X, 3), (Y, 1)]$$



# Un altro esempio di DV

B	dist	NH
A	7	A
B	0	-
C	1	C
D	-	-
E	5	E



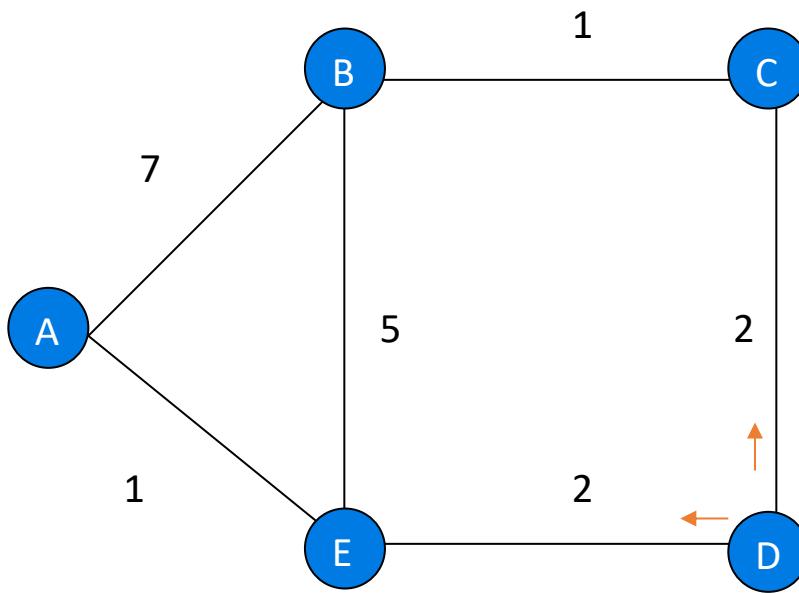
C	dist	NH
A	-	-
B	1	B
C	0	-
D	2	D
E	-	-

A	dist	NH
A	0	-
B	7	B
C	-	-
D	-	-
E	1	E

E	dist	NH
A	1	A
B	5	B
C	-	-
D	2	D
E	0	-

D	dist	NH
A	-	-
B	-	-
C	2	C
D	0	-
E	2	E

# Un altro esempio di DV

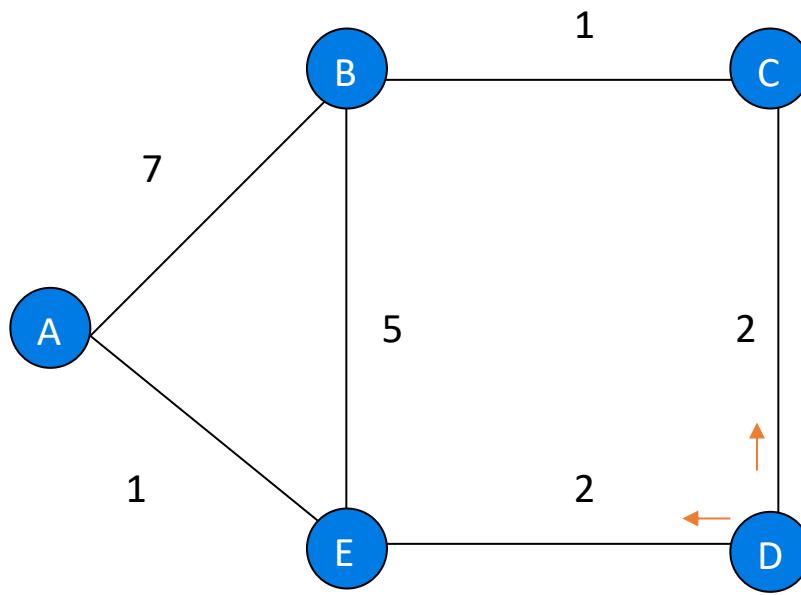


C	dist	NH
A	-	-
B	1	B
C	0	-
D	2	D
E	-	-

E	dist	NH
A	1	A
B	5	B
C	-	-
D	2	D
E	0	-

D	dist	NH
A	-	-
B	-	-
C	2	C
D	0	-
E	2	E

# Un altro esempio di DV



C	dist	NH
A	-	-
B	1	B
C	0	-
D	2	D
E	4	D

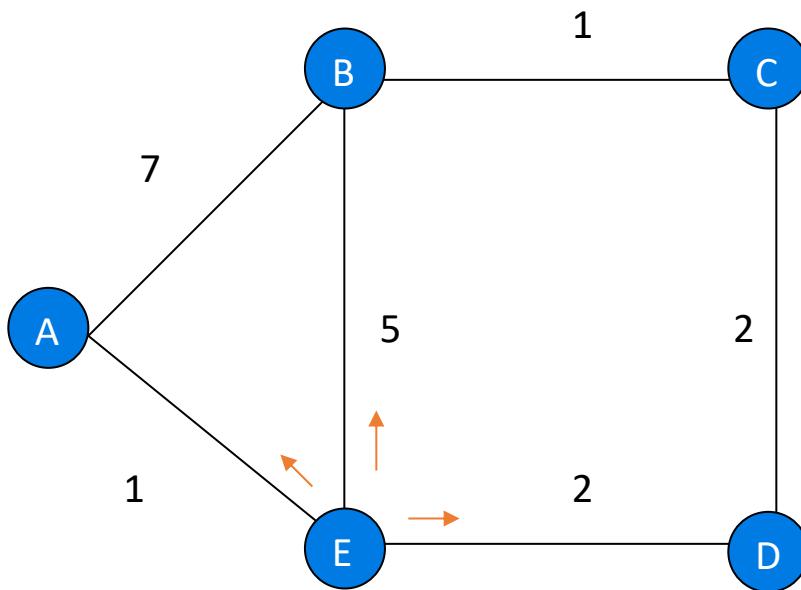
E	dist	NH
A	1	A
B	5	B
C	4	D
D	2	D
E	0	-

D	dist	NH
A	-	-
B	-	-
C	2	C
D	0	-
E	2	E

# Un altro esempio di DV

B	dist	NH
A	7	A
B	0	-
C	1	C
D	-	-
E	5	E

A	dist	NH
A	0	-
B	7	B
C	-	-
D	-	-
E	1	E



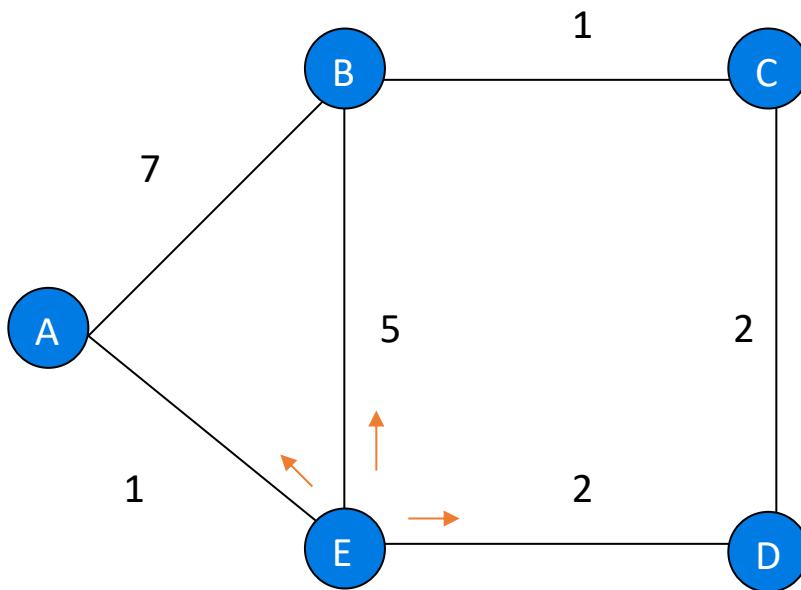
E	dist	NH
A	1	A
B	5	B
C	4	D
D	2	D
E	0	-

D	dist	NH
A	-	-
B	-	-
C	2	C
D	0	-
E	2	E

# Un altro esempio di DV

B	dist	NH
A	6	E
B	0	-
C	1	C
D	7	E
E	5	E

A	dist	NH
A	0	-
B	6	E
C	5	E
D	3	E
E	1	E

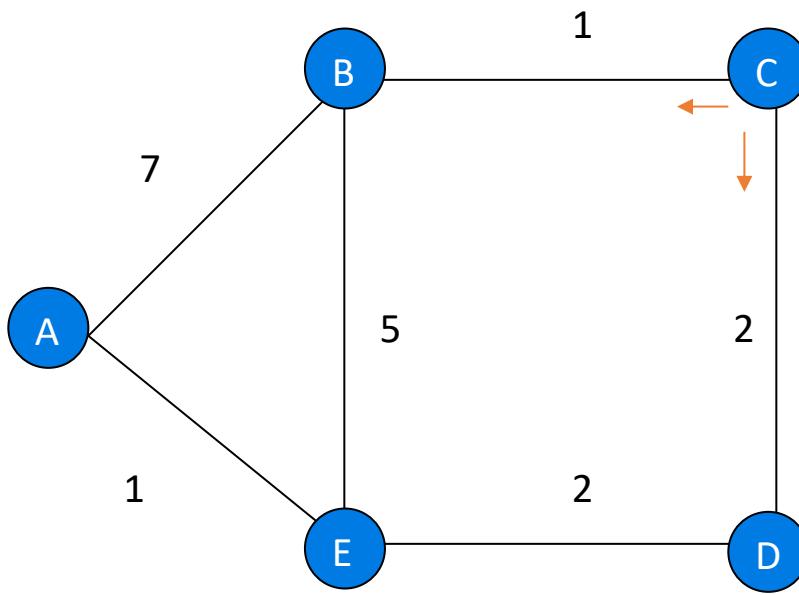


E	dist	NH
A	1	A
B	5	B
C	4	D
D	2	D
E	0	-

D	dist	NH
A	3	E
B	7	E
C	2	C
D	0	-
E	2	E

# Un altro esempio di DV

B	dist	NH
A	6	E
B	0	-
C	1	C
D	7	E
E	5	E

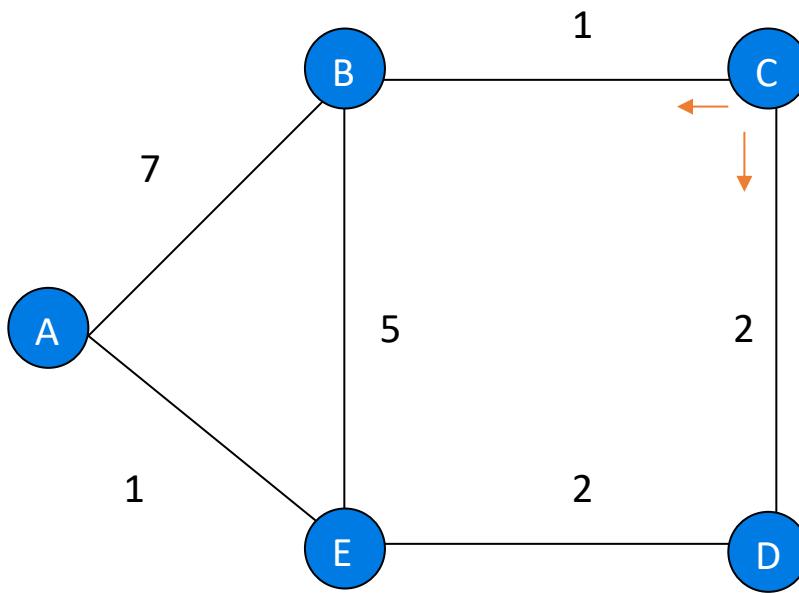


C	dist	NH
A	-	-
B	1	B
C	0	-
D	2	D
E	4	D

D	dist	NH
A	3	E
B	7	E
C	2	C
D	0	-
E	2	E

# Un altro esempio di DV

B	dist	NH
A	6	E
B	0	-
C	1	C
D	3	C
E	5	E

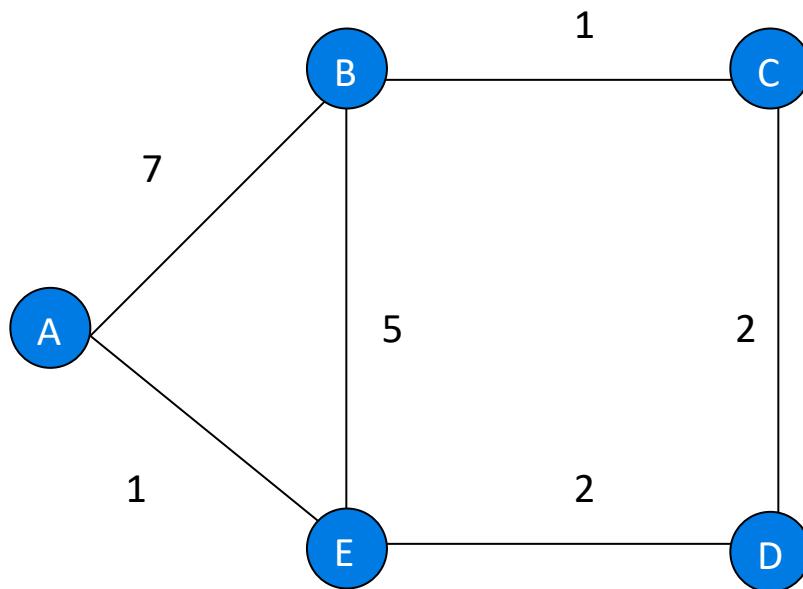


C	dist	NH
A	-	-
B	1	B
C	0	-
D	2	D
E	4	D

D	dist	NH
A	3	E
B	3	C
C	2	C
D	0	-
E	2	E

# Un altro esempio di DV (alla fine...)

B	dist	NH
A	6	E
B	0	-
C	1	C
D	3	C
E	5	E



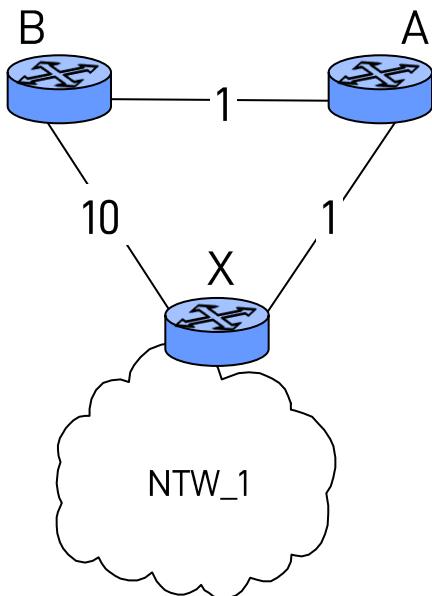
C	dist	NH
A	5	D
B	1	B
C	0	-
D	2	D
E	4	D

A	dist	NH
A	0	-
B	6	E
C	5	E
D	3	E
E	1	E

E	dist	NH
A	1	A
B	5	B
C	4	D
D	2	D
E	0	-

D	dist	NH
A	3	E
B	3	C
C	2	C
D	0	-
E	2	E

# Count-to-infinity



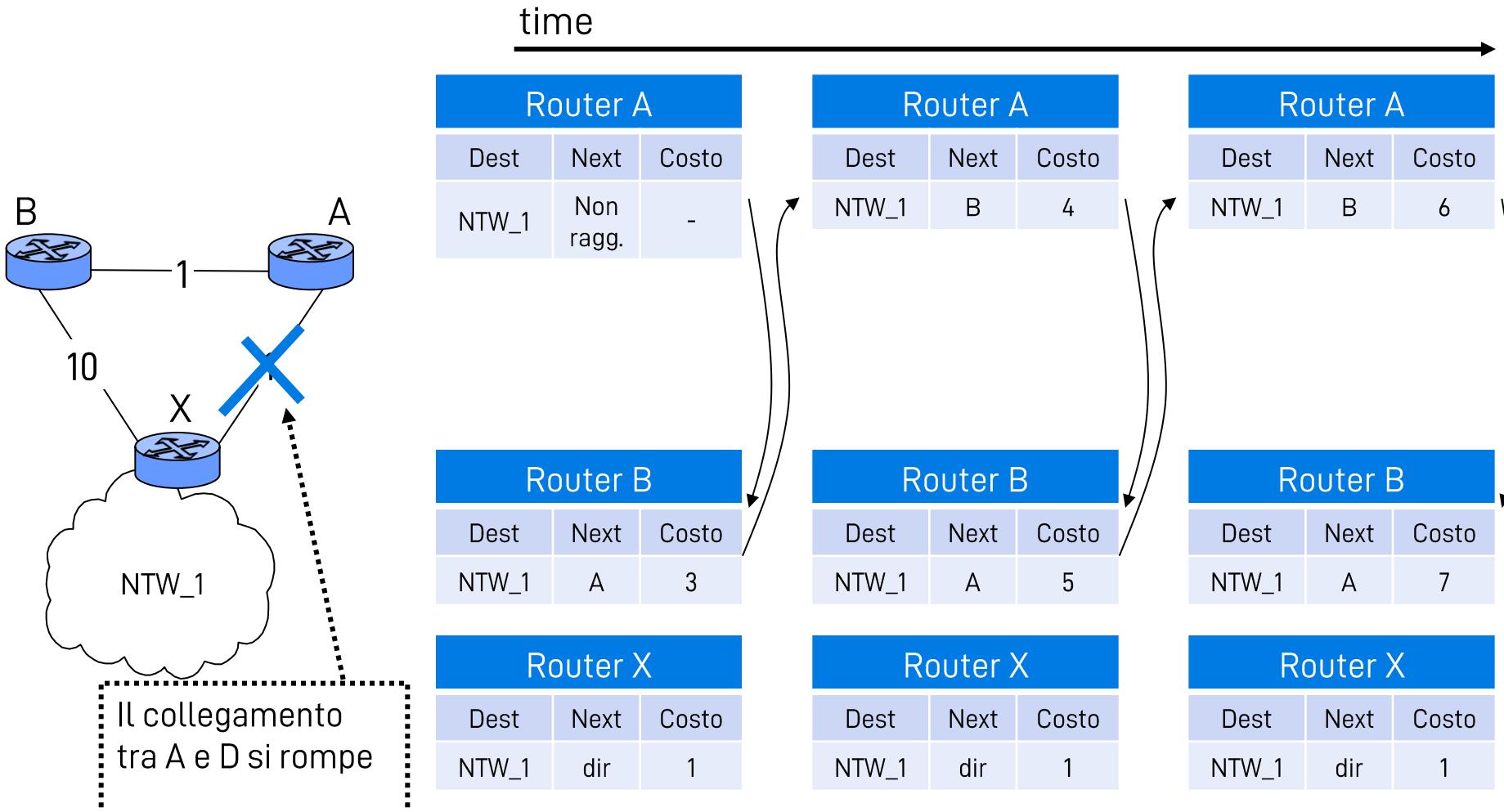
Router A		
Dest	Next	Costo
NTW_1	X	2

Router B		
Dest	Next	Costo
NTW_1	A	3

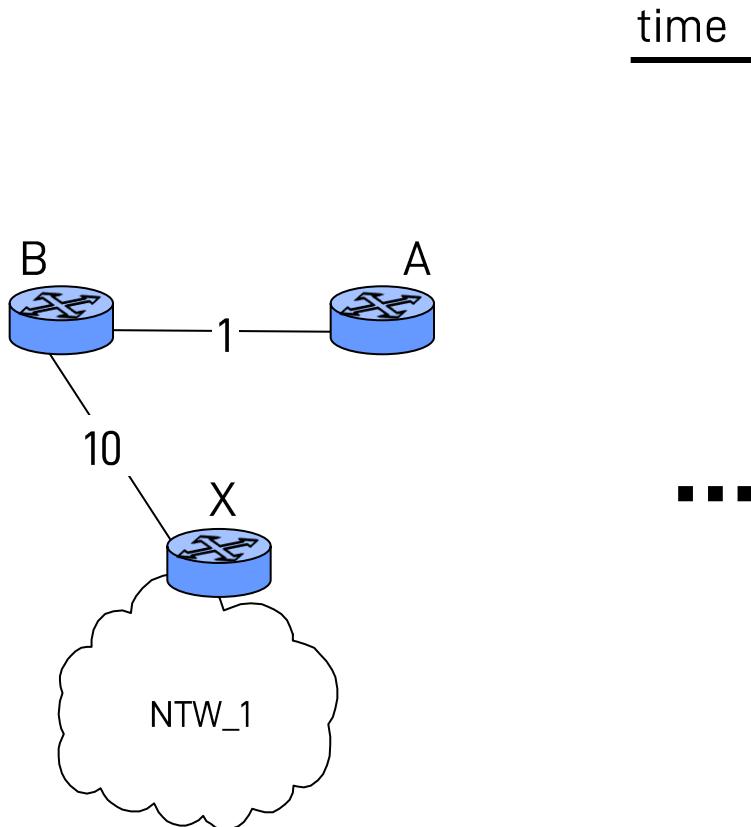
Router X		
Dest	Next	Costo
NTW_1	dir	1

- Consideriamo le righe corrispondenti alla rete NTW\_1 nelle tabelle a fianco
- Il router X è direttamente connesso a NTW\_1

# Count-to-infinity



# Count-to-infinity



time →

Router A

Dest	Next	Costo
NTW_1	B	11

Router A

Dest	Next	Costo
NTW_1	B	12

Router B

Dest	Next	Costo
NTW_1	X	11

Router B

Dest	Next	Costo
NTW_1	X	11

Router X

Dest	Next	Costo
NTW_1	dir	1

Router X

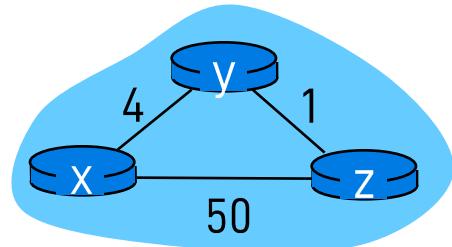
Dest	Next	Costo
NTW_1	dir	1

# Soluzioni per il count-to-infinity

- **Massimo numero di hop** per la propagazione dei DV: 15 hop
  - ❖ Abbassa il tempo di convergenza
- **Split Horizon**
  - ❖ Semplice
  - ❖ Quando un nodo manda aggiornamenti a un vicino, *omette le rotte apprese da quel vicino*
  - ❖ Nell'esempio precedente, B non avrebbe incluso (NTW\_1, 3) nel DV inviato ad A, perché lo aveva appreso proprio da A
- **Poisoned reverse**
  - ❖ Finché un nodo x raggiunge un nodo z attraverso y, x comunica ad y che  $D_x(z) = \infty$ 
    - Nell'esempio precedente, B includerebbe la riga (NTW\_1,  $\infty$ ) nel DV inviato ad A, perché A è il nodo attraverso cui B raggiunge NTW\_1

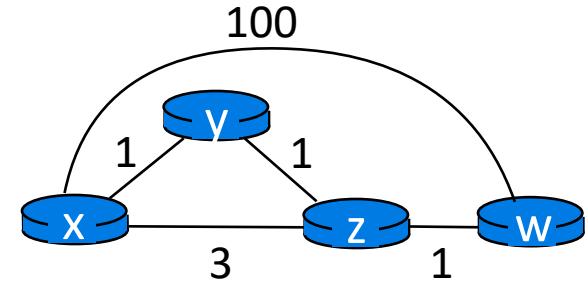
# Distance vector: cambio dei costi

- Se i costi dei link cambiano
  - ❖ Le buone notizie (costi minori) viaggiano in fretta
  - ❖ Le cattive notizie (costi più alti) viaggiano lentamente
  
- Provate a far girare l'algoritmo DV nella topologia sotto quando
  - ❖ Il costo del link x-y diminuisce da 4 a 1
  - ❖ Il costo del link x-y aumenta da 4 a 60



# Split horizon e poisoned reverse: funzionano sempre?

- Solo un esempio
  - ❖ Dipende sempre dall'ordine dei messaggi



Router X		
Dest	NH	C
W	Y	3

Router Y		
Dest	NH	C
W	Z	2

Router Z		
Dest	NH	C
W	Z	1

time

# Split horizon e poisoned reverse: funzionano sempre?

- Si rompe il link Z-W
  - ❖ Siccome i nodi operano in modo distribuito, X potrebbe inviare il proprio messaggio in qualunque momento

Router X		
Dest	NH	C
W	Y	3

$$D_X = [(Y,1), (Z,3)] \text{ per } Y$$

$$D_X = [(W,3), (Y,1), (Z,3)] \text{ per } Z$$

**Si crea un ciclo!**

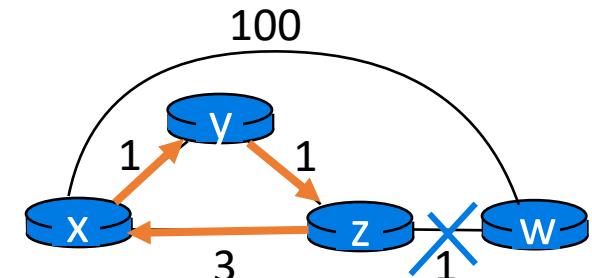
Router Y		
Dest	NH	C
W	Z	2

Router Y		
Dest	NH	C
W	Z	2

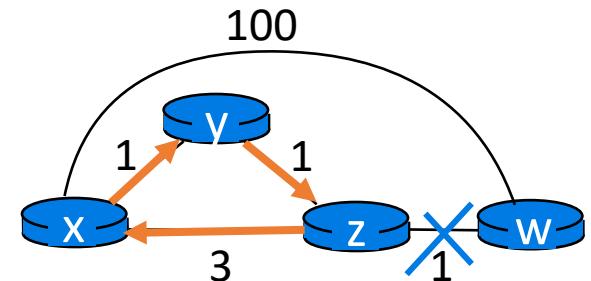
Router Z		
Dest	NH	C

Router Z		
Dest	NH	C
W	X	6

time



# Split horizon e poisoned reverse: funzionano sempre?



Router X		
Dest	NH	C
W	Y	3

Router X		
Dest	NH	C
W	Y	3

Router Y		
Dest	NH	C
W	Z	2

Router Y		
Dest	NH	C
W	Z	2

Router Y		
Dest	NH	C
W	Z	7

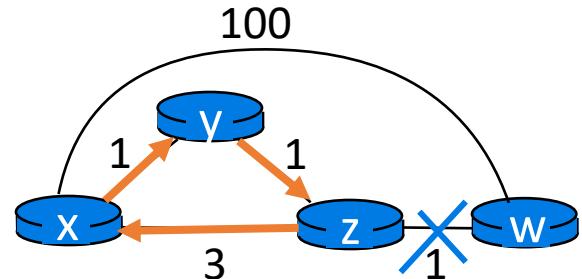
Router Z		
Dest	NH	C

Router Z		
Dest	NH	C
W	X	6

time

$$\begin{aligned}D_Z &= [(X,3), (Y,1)] \text{ per } X \\D_Z &= [(W,6), (X,3), (Y,1)] \text{ per } Y\end{aligned}$$

# Split horizon e poisoned reverse: funzionano sempre?



Router X		
Dest	NH	C
W	Y	3

Router X		
Dest	NH	C
W	Y	3

Router X		
Dest	NH	C
W	Y	8

Router Y		
Dest	NH	C
W	Z	2

Router Y		
Dest	NH	C
W	Z	2

Router Y		
Dest	NH	C
W	Z	7

Router Z		
Dest	NH	C

Router Z		
Dest	NH	C
W	X	6

time

Router Z		
Dest	NH	C
W	X	6

$$\begin{aligned}D_Y &= [(W,7), (X,3), (Y,1)] \text{ per } X \\D_Y &= [(X,3), (Y,1)] \text{ per } Z\end{aligned}$$

# Split horizon e poisoned reverse: funzionano sempre?

Router X		
Dest	NH	C
W	Y	3

Router X		
Dest	NH	C
W	Y	3

Router X		
Dest	NH	C
W	Y	8

Router Y		
Dest	NH	C
W	Z	2

Router Y		
Dest	NH	C
W	Z	2

Router Y		
Dest	NH	C
W	Z	7

Router Z		
Dest	NH	C

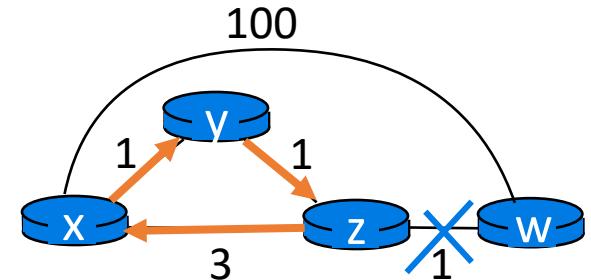
Router Z		
Dest	NH	C
W	X	6

Router Z		
Dest	NH	C
W	X	6

Router Y		
Dest	NH	C
W	Z	7

Router Z		
Dest	NH	C
W	X	11

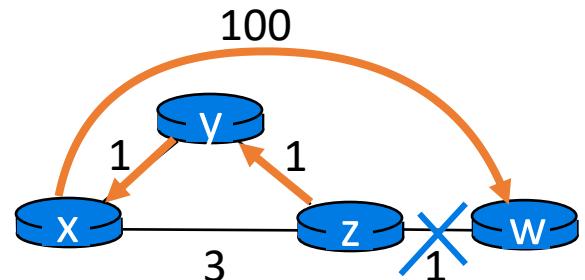
time



$$D_X = [(Y, 1), (Z, 3)] \text{ per } Y$$

$$D_X = [(W, 8), (Y, 1), (Z, 3)] \text{ per } Z$$

# Dopo molti cicli...



Router X		
Dest	NH	C
W	W	100

■ ■ ■

Router Y		
Dest	NH	C
W	X	101

Router Z		
Dest	NH	C
W	Y	102

time

# Sommario

- Visione d'insieme
  - ❖ Data plane e control plane
- Come è fatto un router
- Il protocollo IP
  - ❖ Formato datagrammi
  - ❖ Frammentazione
  - ❖ Indirizzamento e NAT
  - ❖ Indirizzamento classless
  - ❖ Tipi di indirizzi
  - ❖ Address Resolution Protocol (ARP)
  - ❖ Internet Control Message Protocol (ICMP)
- Il protocollo IP (segue)
  - ❖ Dynamic Host Configuration Protocol (DHCP)
  - ❖ Il viaggio di un pacchetto
  - ❖ IPv6
- **Protocolli di instradamento**
  - ❖ Link state: Dijkstra
  - ❖ Internet, AS, OSPF
  - ❖ Distance vector: Bellman-Ford
  - ❖ RIP
- BGP

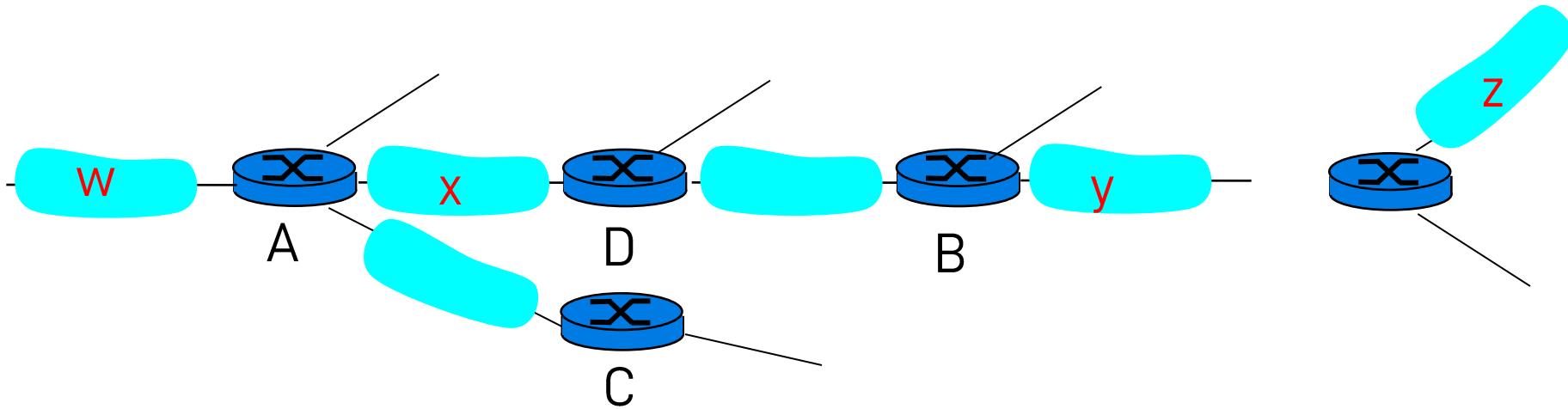
# Intra-AS: Routing Information Protocol

- Protocollo per routing Intra-AS
  - ❖ Implementa distance vector
  - ❖ Vantaggi
    - Semplice da implementare e gestire
  - ❖ Svantaggi
    - Convergenza lenta
    - Dimensione della rete limitata
- È tipicamente incluso in UNIX BSD dal 1982

# Intra-AS: Routing Information Protocol

- Il costo dei link è il numero di hop
  - ❖ Al massimo = 15, mentre  $16 = \infty$  (limita il tempo di convergenza)
  - ❖ Un singolo hop potrebbe avere comunque un costo  $> 1$
- Ogni 30 secondi (o se cambiano le tabelle di routing) RIP invia i distance vector (DV)
  - ❖ “RIP advertisement”
  - ❖ Ogni messaggio contiene un elenco comprendente
    - Fino a 25 sottoreti di destinazione all'interno dell'AS
    - La distanza del mittente rispetto a ciascuna di tali sottoreti
  - ❖ Usa UDP, porta 520, indirizzo multicast 224.0.0.9

# RIP: esempio



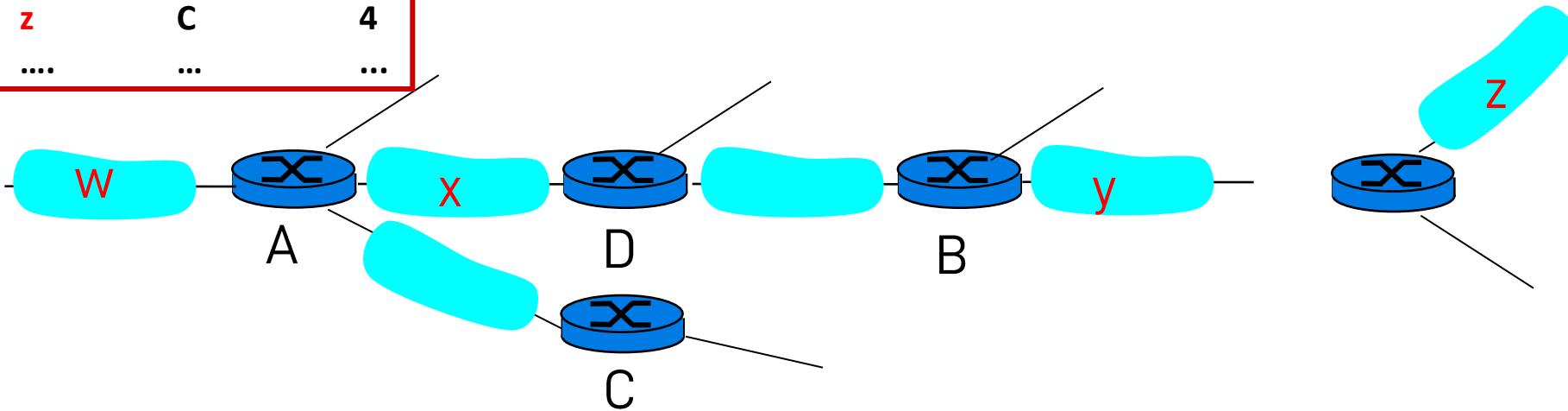
Sottorete destin.	Router successivo	Numero di hop verso la dest.
W	A	2
y	B	2
z	B	7
x	--	1
....	....	....

Tabella d'instradamento nel router D

# RIP: esempio

Dest	Next	#hop
w	-	1
x	-	1
z	c	4
....	...	...

Notifica dal  
router A a D



Sottorete destin.      Router successivo      Numero di hop verso la dest.

w	A	2
y	B	2
z	<del>B A</del>	<del>7</del> 5
x	--	1
....	....	....

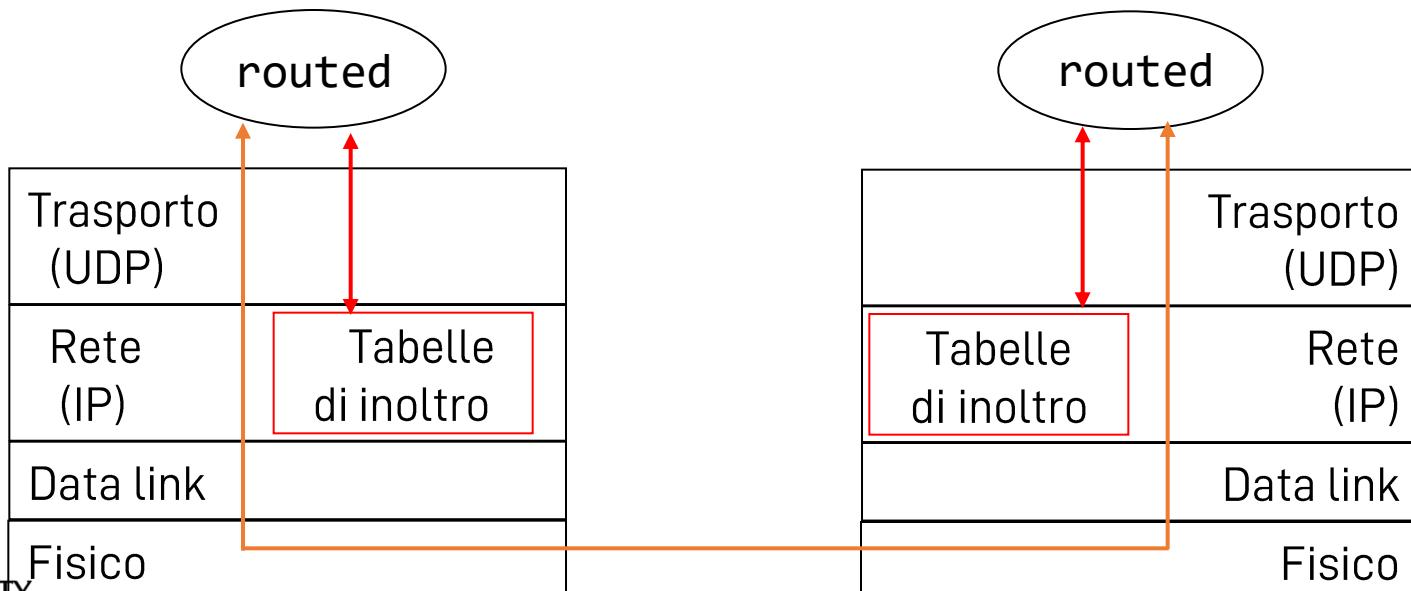
Tabella d'instradamento nel router D

# RIP: guasto sul collegamento e recupero

- Se un router non riceve notizie dal suo vicino per 180 s
  - ❖ Il nodo adiacente/il collegamento viene considerato guasto
  - ❖ RIP modifica la tabella d'instradamento locale
  - ❖ Propaga l'informazione mandando annunci ai router vicini
  - ❖ I vicini inviano nuovi messaggi (se la loro tabella d'instradamento è cambiata)
  - ❖ L'informazione che il collegamento è fallito si propaga su tutta la rete
  - ❖ Poisoned reverse per evitare i loop (distanza infinita = 16 hop)

# Tabella d'instradamento RIP

- Un processo chiamato **routed** esegue RIP, ossia mantiene le informazioni d'instradamento e scambia messaggi con i processi **routed** nei router vicini
- Poiché RIP viene implementato come un processo a livello di applicazione, può inviare e ricevere messaggi su una socket standard e utilizzare un protocollo di trasporto standard



# Confronto tra algoritmi link state e distance vector

## Complessità dello scambio mess.

- LS: con  $n$  nodi ed  $E$  link, vengono inviati  $O(nE)$  messaggi
- DV: messaggi inviati solo ai vicini
  - ❖ Tempo di convergenza: dipende

## Velocità di convergenza

- LS: L'algoritmo di complessità  $O(n^2)$  richiede  $O(nE)$  messaggi
  - ❖ Ci possono essere oscillazioni
- DV: dipende
  - ❖ Si potrebbero generare cicli
  - ❖ Problema del count-to-infinity

## Robustezza: che succede se un router si rompe?

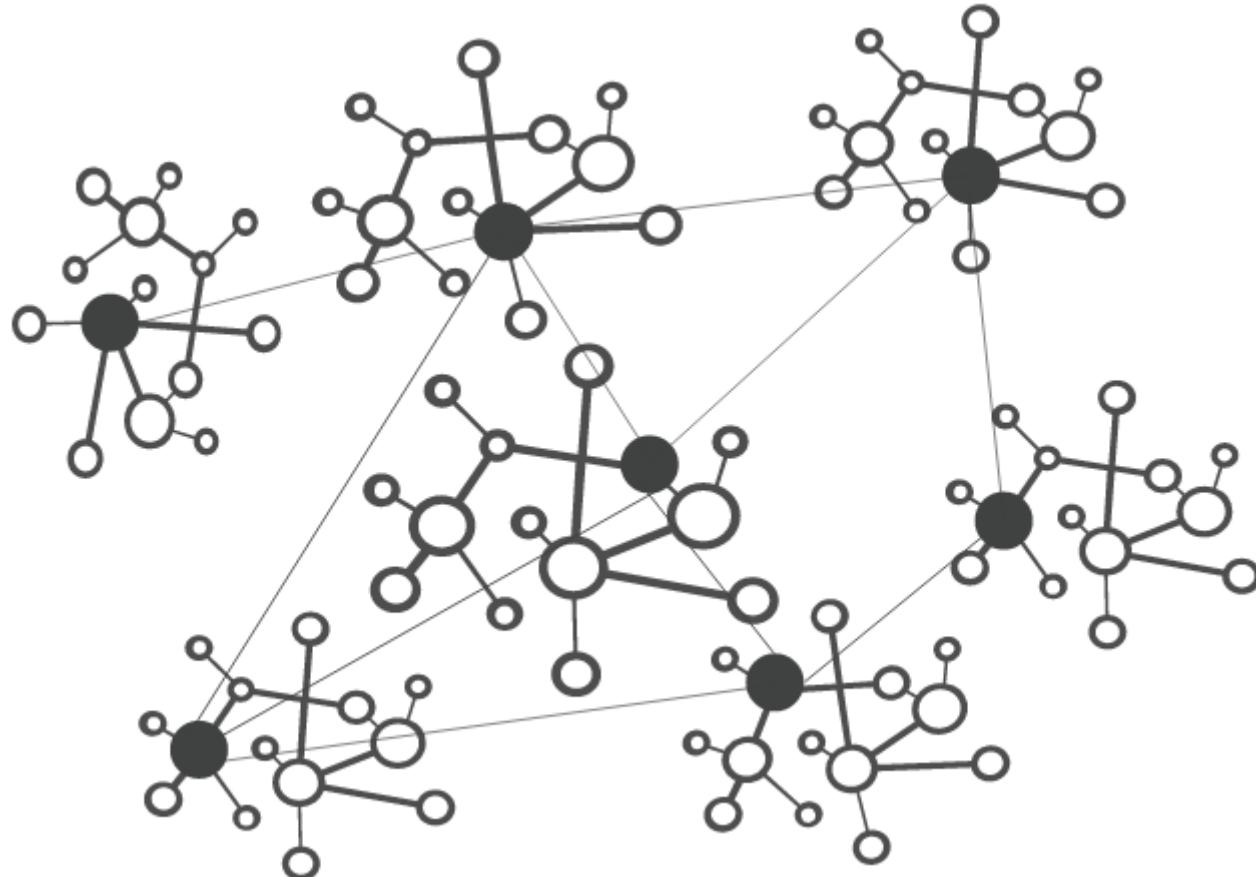
- LS:
  - ❖ I nodi potrebbero comunicare costi sbagliati per i link
  - ❖ Ogni nodo gestisce la propria tabella indipendentemente
- DV:
  - ❖ I nodi potrebbero comunicare costi sbagliati per i percorsi
  - ❖ La tabella di ogni nodo è usata anche dagli altri
    - Gli errori si propagano attraverso la rete

# Sommario

- Visione d'insieme
  - ❖ Data plane e control plane
- Come è fatto un router
- Il protocollo IP
  - ❖ Formato datagrammi
  - ❖ Frammentazione
  - ❖ Indirizzamento e NAT
  - ❖ Indirizzamento classless
  - ❖ Tipi di indirizzi
  - ❖ Address Resolution Protocol (ARP)
  - ❖ Internet Control Message Protocol (ICMP)
- Il protocollo IP (segue)
  - ❖ Dynamic Host Configuration Protocol (DHCP)
  - ❖ Il viaggio di un pacchetto
  - ❖ IPv6
- Protocolli di instradamento
  - ❖ Link state: Dijkstra
  - ❖ Internet, AS, OSPF
  - ❖ Distance vector: Bellman-Ford
  - ❖ RIP
- BGP
  - ❖ Prossima lezione

# Border Gateway Protocol (BGP)

- Basata sulle slide di Mattia Milani (2020, 2021)

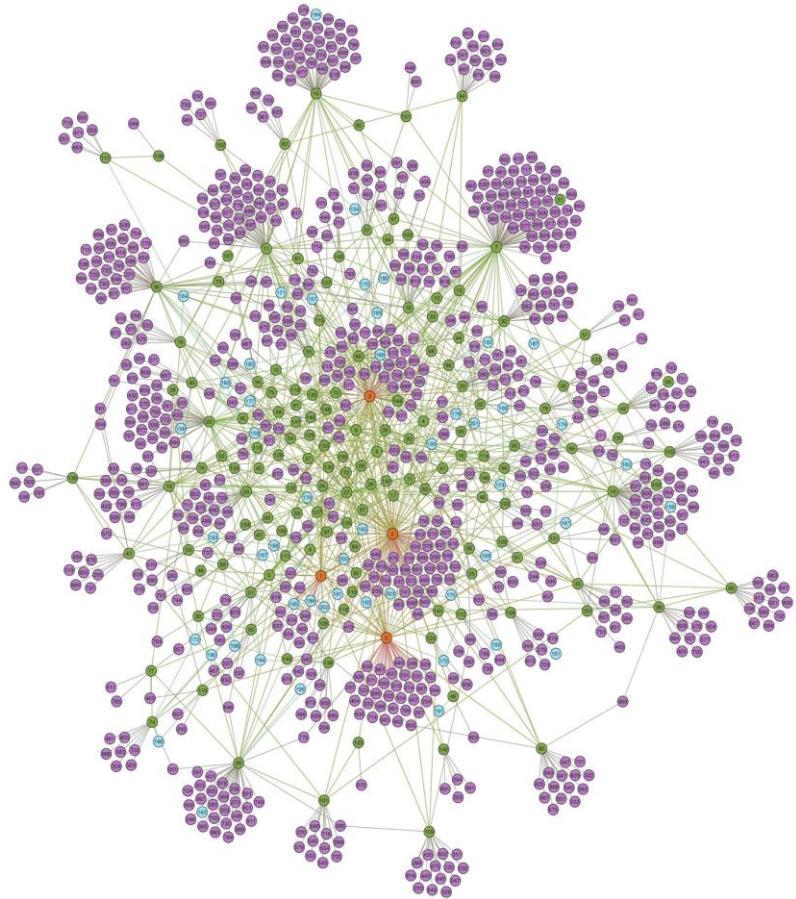
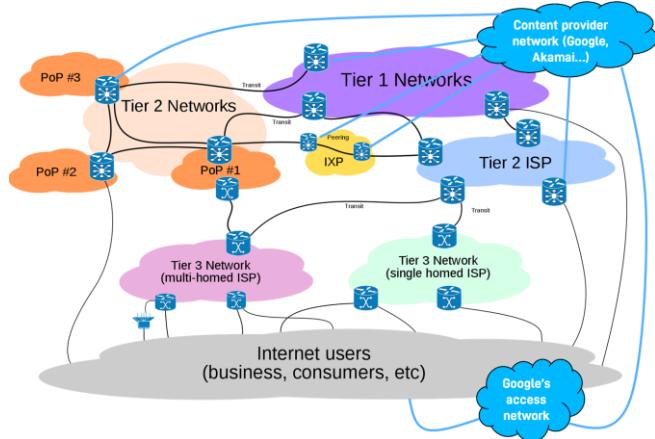


# Perché BGP?

- ❑ Perché manca l'ultimo ingrediente per interconnettere tutto
- ❑ Cosa sappiamo finora
  - ❖ Come stabilire regole di inoltro usando protocolli di routing internamente ad un AS (es. OSPF, RIP)
  - ❖ Come gestire gli indirizzi IP ed eseguire gli inoltri (es. CIDR, longest prefix matching, ...)
  - ❖ Che in un AS esistono router di bordo, che connettono l'AS con il resto di Internet
- ❑ Cosa manca?
  - ❖ Far parlare i router di bordo tra loro per realizzare questa connessione tra diversi AS

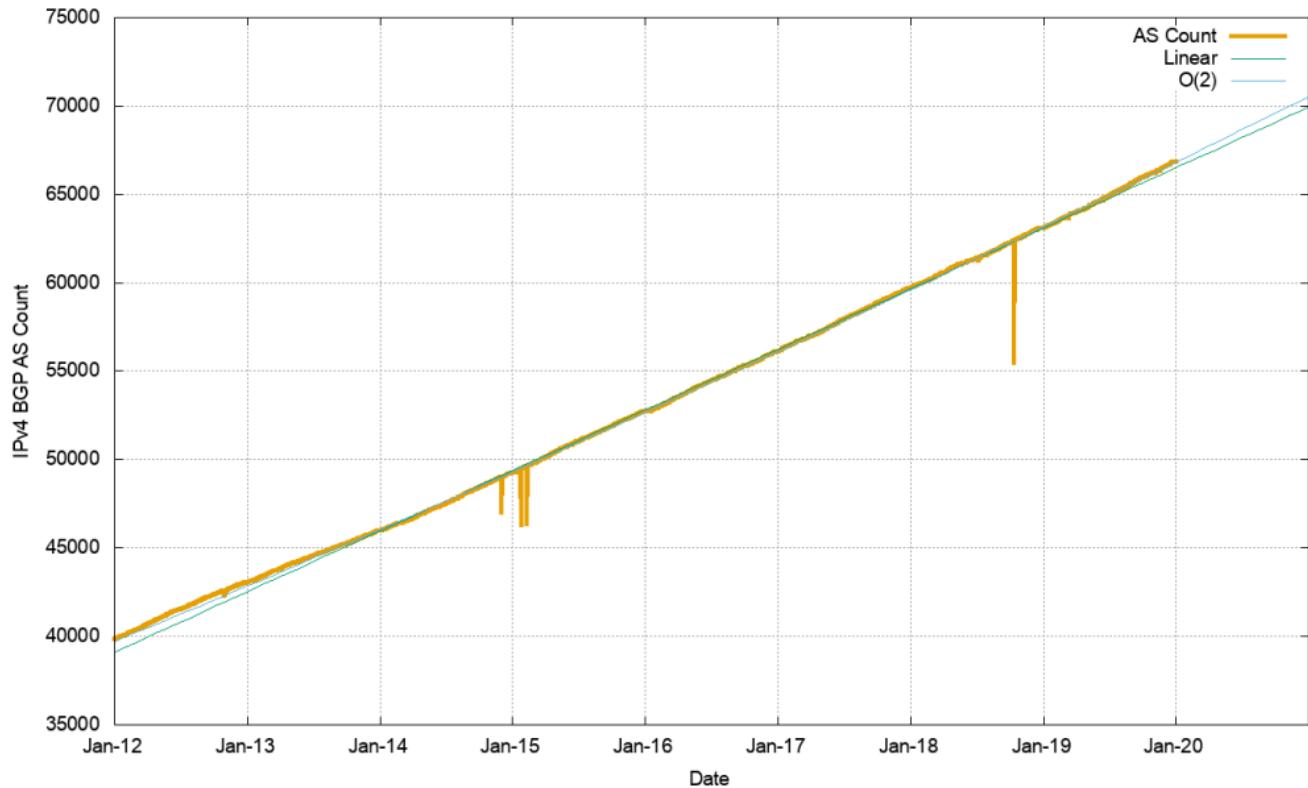
# Ricordiamo cos'è un AS

- ❑ Un insieme di router ed altri sistemi di rete che si trovano sotto la stessa amministrazione
    - ❖ Es.: un network operator che controlla una porzione di indirizzi/traffico/flussi di Internet
  - ❑ Molteplici livelli di AS
    - ❖ Tier one, ..., customers
  - ❑ Interconnessioni tra gli AS per lo scambio di informazioni



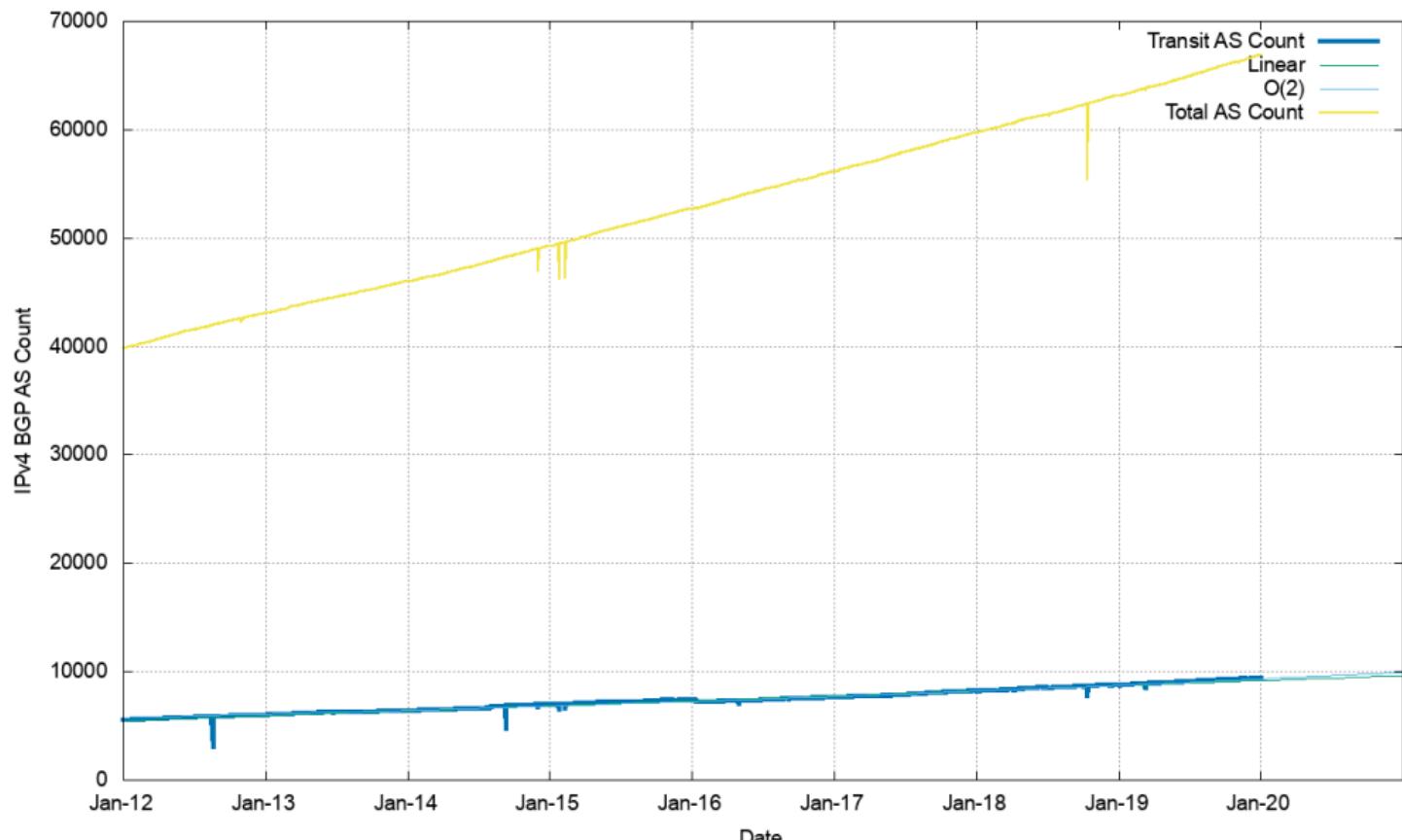
# BGP: inter-AS networking

- Gli AS comunicano tra loro per condividere info di raggiungibilità
- Ogni AS decide autonomamente i propri punti di ingresso e di uscita
- Ogni AS può decidere di condividere informazioni con alcuni vicini ma non con altri
- Al mondo vi sono più di 60.000 AS interconnessi



# Transit AS

- AS che non condividono conoscenze sulla propria rete, ma fanno solo da tramite



# Border Gateway Protocol (BGP)

- Di fatto, è l'unico protocollo standard per comunicare informazioni di raggiungibilità per le reti (= prefissi)
- BGP v4: RFC 4271 del 2006 e modifiche/miglioramenti negli anni successivi
- BGP è un *Path Vector Protocol*
- Le relazioni tra AS sono complicate, e anche BPG lo è per potersi adattare a tutti i casi possibili
- Principio chiave: libertà amministrativa per ogni AS, che:
  - ❖ Può decidere se pubblicizzare info di raggiungibilità per le proprie reti interne oppure no
  - ❖ Può decidere di ritrarre la raggiungibilità di qualunque propria rete
  - ❖ Può decidere offrire o meno transito verso altri AS in ogni momento

# Border Gateway Protocol (BGP)

- Ogni AS ha un certo numero di router detti «BGP speaker»
- BGP possono parlare attraverso una connessione TCP tra loro
- Una volta connessi, due BGP speaker possono scambiarsi informazioni di raggiungibilità
- Le connessioni instaurate fanno affidamento su TCP
  
- Path vector protocol
  - ❖ Le informazioni condivise includono non solo la raggiungibilità di un certo prefisso, ma anche il **percorso** per raggiungerlo

Destination	Next Hop	Path (ID degli AS)
14.8.0.0/16	AS 15	{15, 32, 571, 11}

# Interconnessioni tra AS

## □ Client – Provider

- ❖ Il client paga il provider per ottenere raggiungibilità

## □ Provider – Client

- ❖ Il provider viene pagato dal client per fornirgli raggiungibilità

## □ Peer

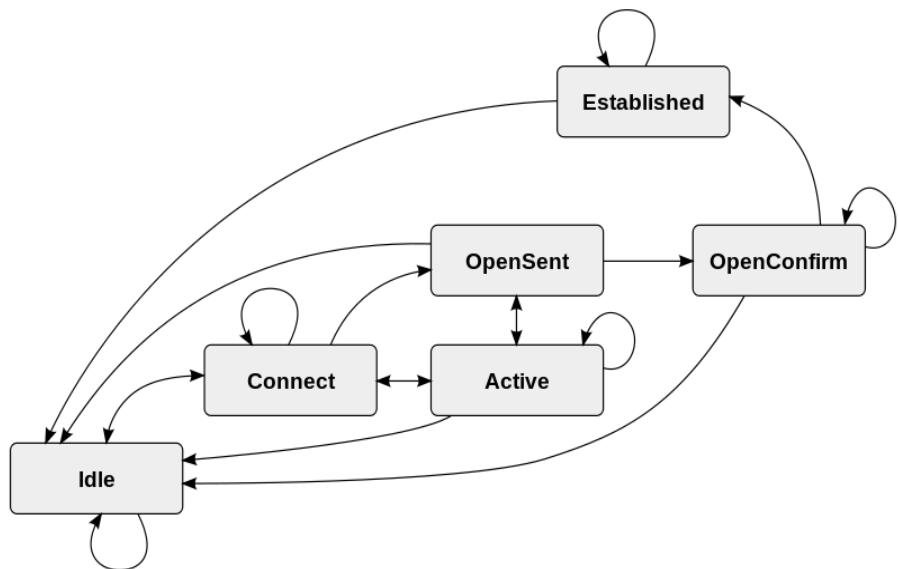
- ❖ Due nodi hanno una relazione di peering nel momento in cui condividono tutte le proprie conoscenze di raggiungibilità

# Policies

- Le interconnessioni tra AS sono controllate da contratti
- Le policies controllano ciò che sono abilitato a condividere con un certo vicino e cosa no
- Ingress Policies
  - ❖ Politiche applicate alle rotte che voglio importare nel mio sistema
- Egress Policies
  - ❖ Politiche applicate alle rotte prima che vengano esportate

# RFC 4271 BGP v4

- ❑ BGP ha raggiunto la sua quarta versione nel 2006
- ❑ La struttura base di BGP è data da una macchina a stati finiti ed il funzionamento del protocollo è dettato dallo stato in cui ci si trova
- ❑ Già nel 1994 BGP prevedeva l'utilizzo di policies e filtri
- ❑ L'RFC 4271 dà solamente delle indicazioni generali su come il protocollo deve funzionare
  - ❖ Ogni sviluppatore ha poi creato la propria versione, in grado di interoperate con le altre



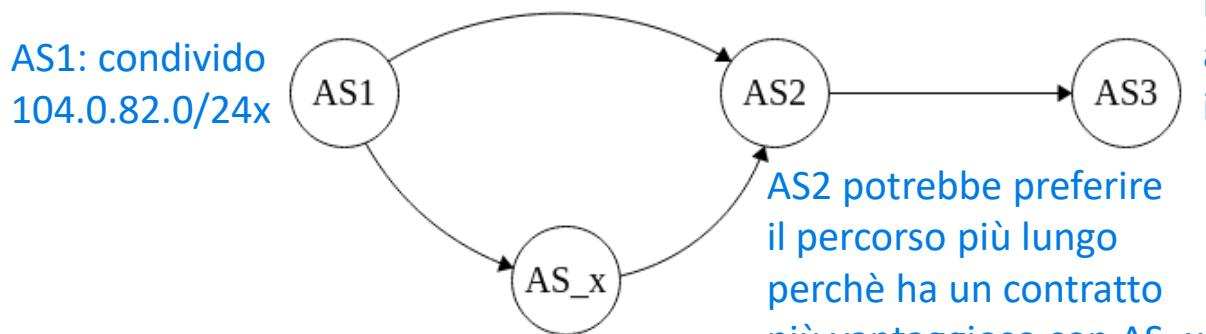
# RFC ?

□ Solo per citarne alcuni:

- [RFC 1997](#) - BGP Communities Attribute
- [RFC 2385](#) - Protection of BGP Sessions via TCP MD5 Signature
- [RFC 2545](#) - Use of BGP Multiprotocol Extensions for IPv6
- [RFC 2918](#) - Route Refresh Capability
- [RFC 3107](#) - Carrying Label Information in BGP
- [RFC 4360](#) - BGP Extended Communities Attribute
- [RFC 4364](#) - BGP/MPLS IPv4 Virtual Private Networks
- [RFC 1997](#) - BGP Communities Attribute
- [RFC 2385](#) - Protection of BGP Sessions via TCP MD5 Signature
- [RFC 2545](#) - Use of BGP Multiprotocol Extensions for IPv6
- [RFC 2918](#) - Route Refresh Capability
- [RFC 3107](#) - Carrying Label Information in BGP
- [RFC 4360](#) - BGP Extended Communities Attribute
- [RFC 4364](#) - BGP/MPLS IPv4 Virtual Private Networks

# Best Path BGP

- Il best path in BGP è diverso dal Best Path di OSPF o RIP, in cui si preferiva il percorso più breve, o comunque con costo minore
- Il concetto di "preferito" in BGP è vago e dipende principalmente dalle policy degli AS
- Un Best Path può variare in base anche ad un solo cambiamento nelle policy di un singolo AS che si trova sul percorso verso la destinazione



AS2 potrebbe preferire il percorso più lungo perché ha un contratto più vantaggioso con AS\_x o perchè il link con AS1 non è molto stabile

AS3 potrebbe decidere che non gli piace raggiungere AS1 attraverso AS2 perciò ignorerà entrambe le rotte

# Best Path BGP

- ❑ Ogni speaker BGP condivide solamente il proprio best path per raggiungere una destinazione
- ❑ Se decide di ignorare certi percorsi (come AS3 poco fa), questi non potranno mai diventare best path per quella destinazione, e non verranno mai condivisi
- ❑ I best path vengono solitamente installati nella routing table del router che fa da speaker BGP

**Non è detto che il best path BGP  
sia il percorso più veloce**

# Messaggi

## □ BGP usa 4 messaggi

### □ **Open**

❖ Usato per aprire una nuova connessione con un altro nodo BGP

### □ **Notification**

❖ Usato per condividere errori

### □ **KeepAlive**

❖ Usato per mantenere attiva la connessione

### □ **Update**

❖ Usato per inoltrare conoscenza

# OPEN

0	1	2	3
0 1 2 3 4 5 6 7 8 9 0	1 2 3 4 5 6 7 8 9 0	1 2 3 4 5 6 7 8 9 0 1	
+++-----+   Version	+++-----+   My Autonomous System	+++-----+   Hold Time	
+++-----+   Opt Parm Len	+++-----+   BGP Identifier	+++-----+   Optional Parameters (variable)	
+++-----+	+++-----+	+++-----+	

# OPEN

- ❑ Usato per aprire una connessione BGP
- ❑ Se l'OPEN message viene accettato, viene inviato un messaggio KEEPALIVE di conferma,
  - ❖ Dopo la ricezione del KEEPALIVE, la connessione viene considerata aperta
- ❑ L'hold time è una proposta che si scambiano i due BGP speaker riguardo al tempo di validità della rotta
  - ❖ Viene utilizzato il minore
  - ❖ Una connessione BGP può essere rifiutata a causa dell'hold timer
- ❑ BGP identifier: indirizzo IP identificativo dello speaker, uguale per tutte le interfacce BGP dello speaker

# NOTIFICATION

0	1	2	3
0 1 2 3 4 5 6 7 8 9 0	1 2 3 4 5 6 7 8 9 0	1 2 3 4 5 6 7 8 9 0	1
Error code   Error subcode   Data (variable)			
+-----+-----+-----+-----+			

- 6 Error codes in base al tipo di errore
- 20 Error subcodes in base a dove avviene l'errore

# KEEPALIVE

- ❑ Devono essere inviati in modo tale da non lasciar scadere l'hold timer che è stato scelto per la connessione
- ❑ Possono essere disattivati impostando un Hold Timer di 0s
- ❑ Consistono nell'header BGP senza contenuti, 19 Byte

# UPDATE

- ❑ Contengono informazioni di raggiungibilità ed attributi correlati
- ❑ I messaggi di update sono i responsabili per la disseminazione delle informazioni
- ❑ Le informazioni possono essere di due tipi:
  - ❖ Additive
    - Portano informazioni di raggiungibilità riguardanti nuovi percorsi
  - ❖ Sottrattive
    - Rimuovono percorsi per raggiungere una destinazione

# WITHDRAW

- Un withdraw è l'azione di rimozione di una rotta
  - ❖ All'interno di un pacchetto di update vi è una sezione apposita per le rotte che vengono cancellate
- Si effettua un withdraw quando non vi è più nessun percorso disponibile verso un certo AS



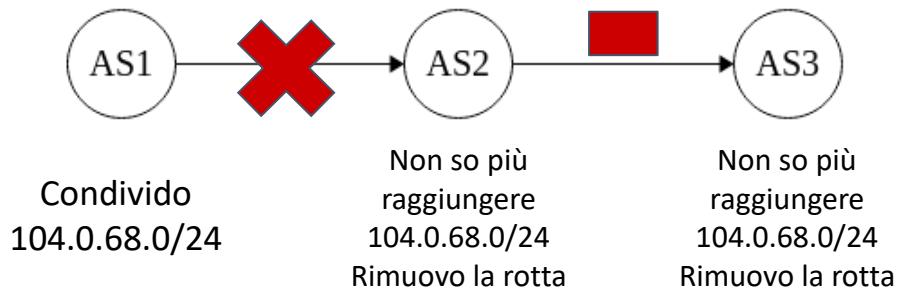
Condivido  
104.0.68.0/24

Raggiungo  
104.0.68.0/24  
Attraverso {AS1}

Raggiungo  
104.0.68.0/24  
Attraverso  
{AS2,AS1}

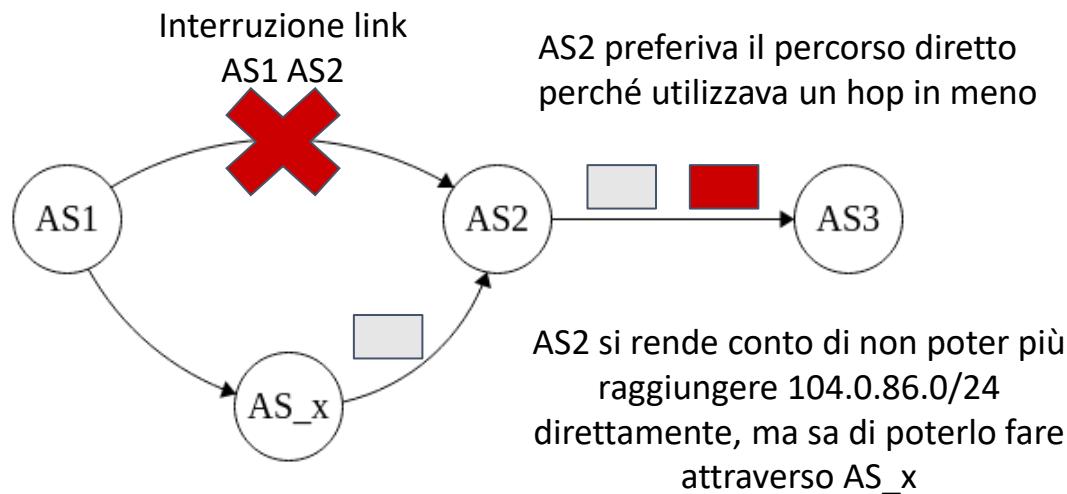
# WITHDRAW

- Un withdraw è l'azione di rimozione di una rotta
  - ❖ All'interno di un pacchetto di update vi è una sezione apposita per le rotte che vengono cancellate
- Si effettua un withdraw quando non vi è più nessun percorso disponibile verso un certo AS



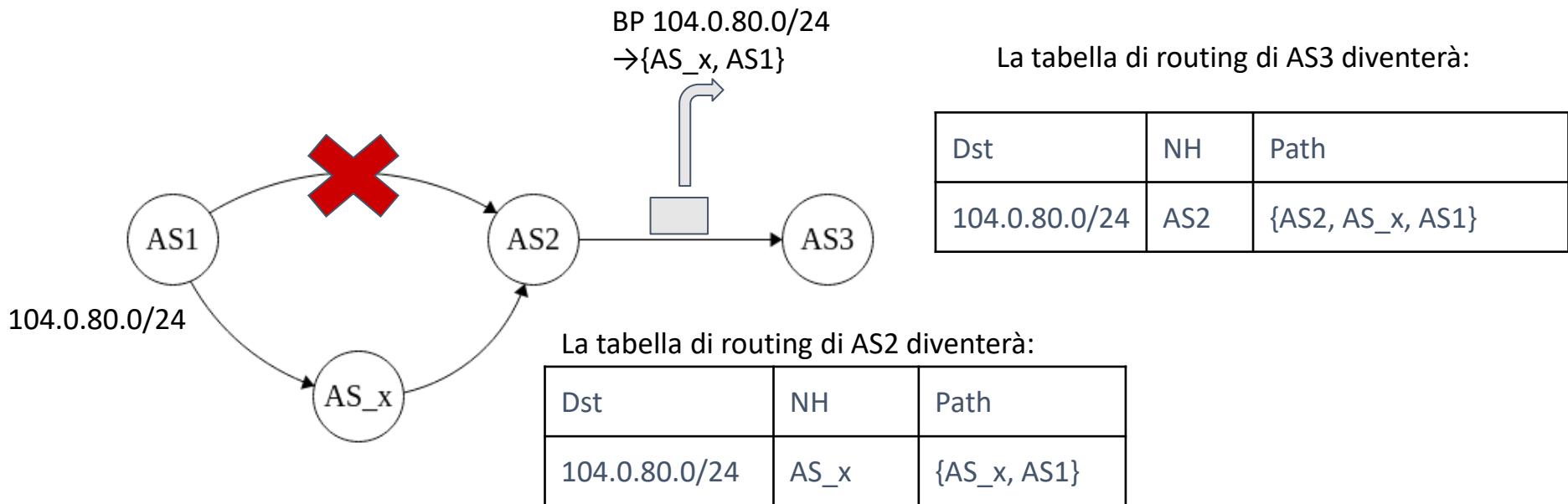
# UPDATE + WITHDRAW

- Nel caso in cui un nodo BGP conoscesse un percorso alternativo per raggiungere una rottta, potrebbe decidere di
  - ❖ Rimuovere la rottta precedente
  - ❖ Utilizzare il nuovo percorso e condividerlo



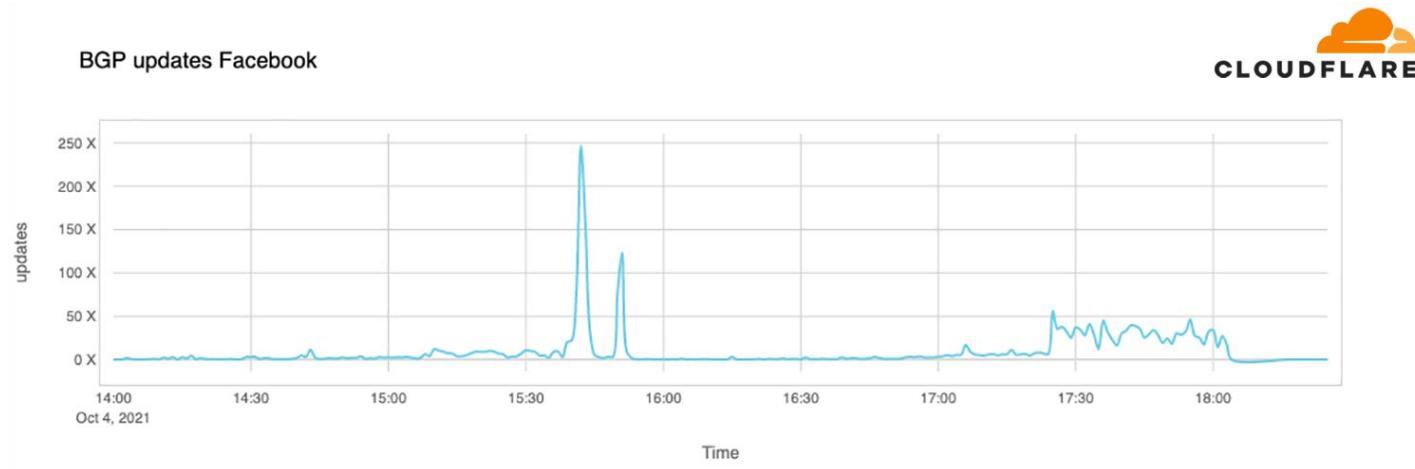
# Implicit WITHDRAW

- Permette di non inviare un pacchetto di withdraw seguito da uno di update, ma solamente uno di "aggiornamento" del percorso migliore



# Facebook disaster

- 4 Ottobre 2021, Facebook non è raggiungibile
  - ❖ DNS lookup error?
- 15:40 UTC Picco di aggiornamenti nelle rotte di facebook

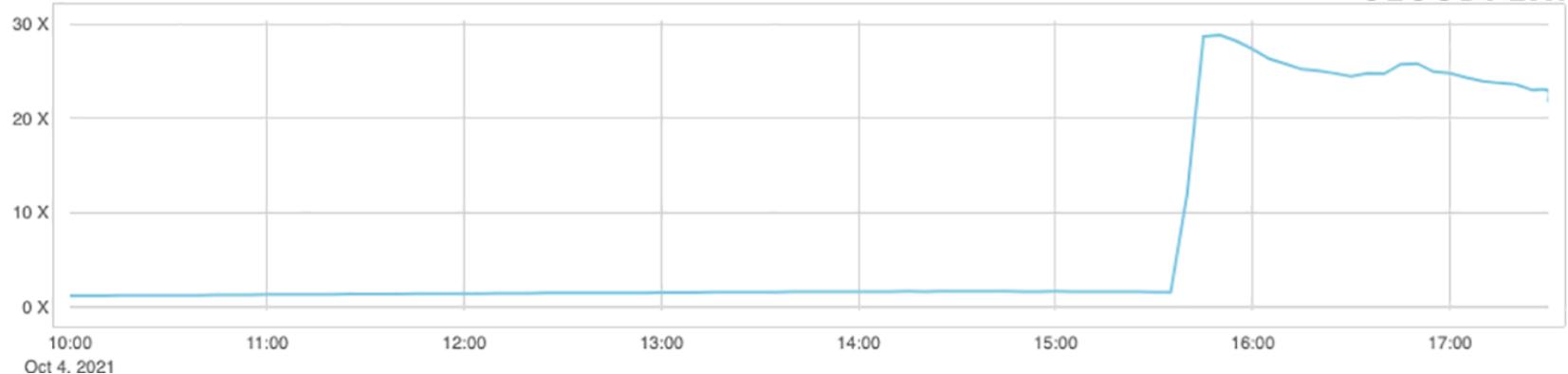


<https://blog.cloudflare.com/october-2021-facebook-outage/>

# Facebook disaster

- CloudFlare ha ricevuto un picco di WITHDRAW da parte di facebook che ritiravano le rotte associate
- E a quel punto tutti hanno cercato di usare i servizi di facebook.
- Alle 21:28 UTC facebook torna online

Queries for websites: facebook, whatsapp, messenger, instagram



<https://blog.cloudflare.com/october-2021-facebook-outage/>

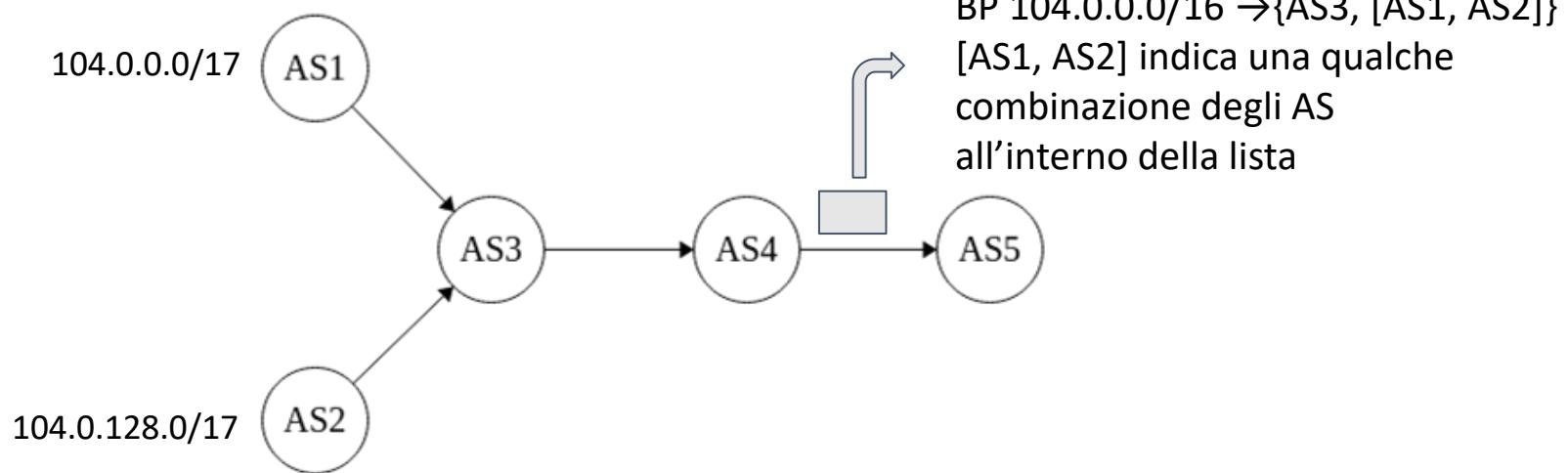
# Facebook disaster – Inside view

- Un comando di routine per controllare lo stato della dorsale di trasporto dati di Facebook innesca dei WITHDRAW
- Il sistema, per un bug, non ha fermato il WITHDRAW delle rotte
- I server DNS disabilitano i messaggi BGP se non sono in grado di comunicare, il che ha portato alla completa disconnessione
- I tool di diagnostica erano parzialmente compromessi ed un collegamento remoto non era disponibile
  
- Non era semplicemente possibile spegnere e riaccendere tutti i server contemporaneamente!

<https://engineering.fb.com/2021/10/05/networking-traffic/outage-details/>

# Aggregazione delle rotte

- Per poter risparmiare informazioni, all'interno dei pacchetti generalmente le rotte vengono aggregate
  - ❖ Si perde però precisione nel percorso



# Filtr

- ❑ I filtri controllano ciò che entra e ciò che esce
- ❑ Possono esserci filtri specifici per ogni connessione BGP
  - ❖ Ingress filters
  - ❖ Egress filters
- ❑ Possono essere generici
  - ❖ Es. «Non lasciar passare nulla che contenga AS129»
- ❑ Possono essere molto precisi
  - ❖ Si può arrivare a filtrare su qualsiasi campo dei singoli pacchetti
- ❑ Un pacchetto che non supera i filtri viene scartato

# Routing information Base (RIB)

- BGP sostanzialmente utilizza 3 tabelle per le rotte
- ADJ\_RIB\_IN
  - ❖ Contiene le rotte che state accettate in ingresso
  - ❖ Usato per valutare percorsi alternativi
- Routing Table
  - ❖ Contiene i best path attuali
- ADJ\_RIB\_OUT
  - ❖ Contiene le rotte che hanno superato i filtri in uscita e che devono essere condivise

# RIB, ricezione di un UPDATE costruttivo

- ❑ Il pacchetto viene valutato dai filtri in ingresso
- ❑ Superati i filtri in ingresso, le rotte ivi contenute vengono **aggiunte** alla ADJ\_RIB\_IN
- ❑ Il best path per la destinazione viene rivalutato in base alla ADJ\_RIB\_IN aggiornata
- ❑ Nel caso in cui il best path fosse variato:
  - ❖ Aggiorna Routing Table
  - ❖ Aggiorna ADJ\_RIB\_OUT
  - ❖ Inoltra i cambiamenti ai vicini

# RIB, ricezione di un UPDATE con WITHDRAW

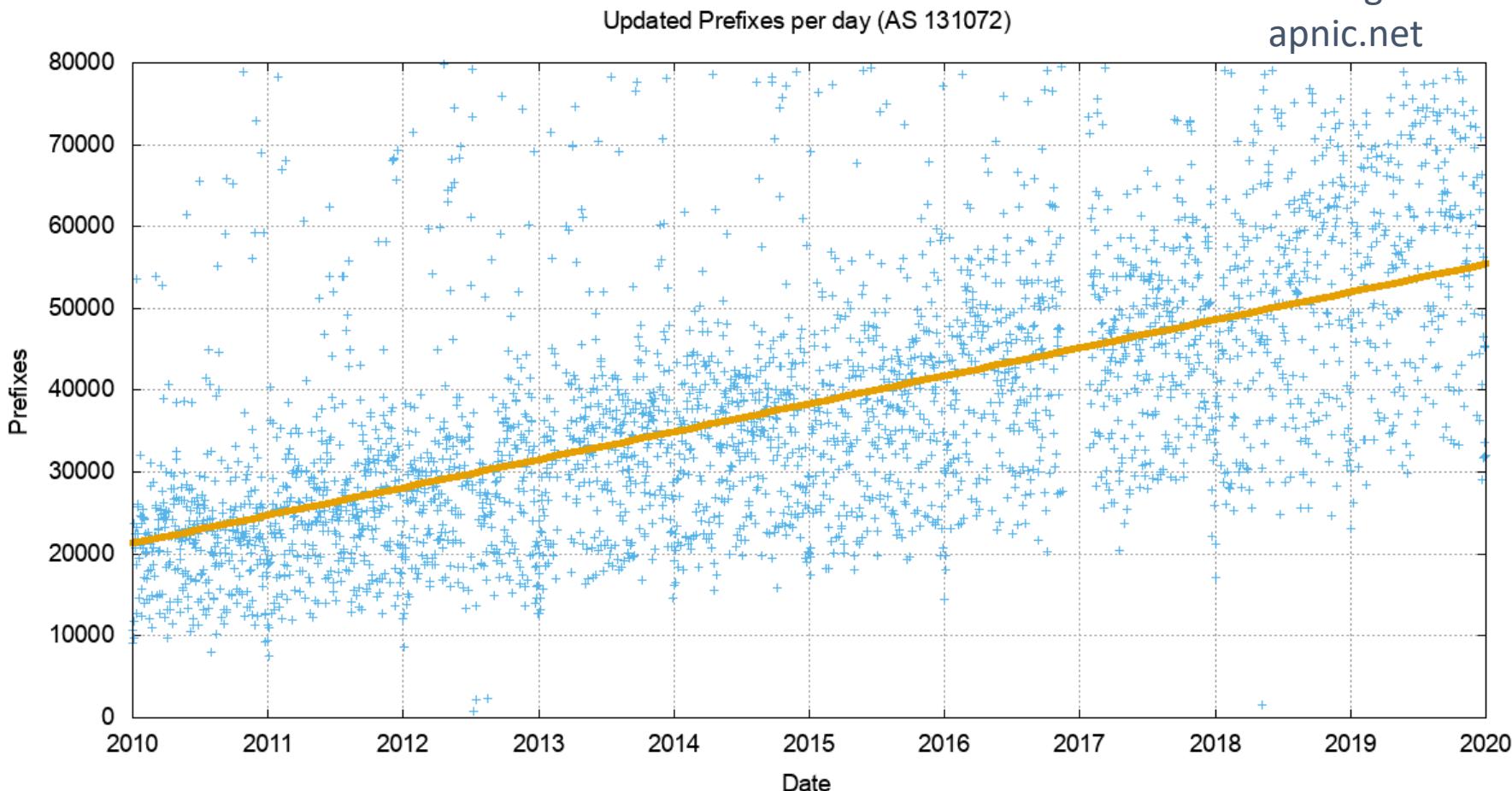
- ❑ Il pacchetto viene valutato dai filtri in ingresso
- ❑ Superati i filtri in ingresso, le rotte ivi contenute vengono **rimosse** dalla ADJ\_RIB\_IN
- ❑ Il best path per la destinazione viene rivalutato in base alla ADJ\_RIB\_IN aggiornata
- ❑ Nel caso in cui il best path fosse variato:
  - ❖ Aggiorna Routing Table
  - ❖ Aggiorna ADJ\_RIB\_OUT
  - ❖ Inoltra i cambiamenti ai vicini

# BGP nel tempo

- Negli anni BGP si è evoluto molto attraverso nuovi IETF RFC
- Se qualcuno volesse produrre la propria versione di un router BGP-compliant da zero, dovrebbe almeno includere circa 30 parametri tecnicamente "opzionali" ma di fatto richiesti
  - ❖ AS communities
  - ❖ RFD
  - ❖ MRAI settings
  - ❖ Extended Communities

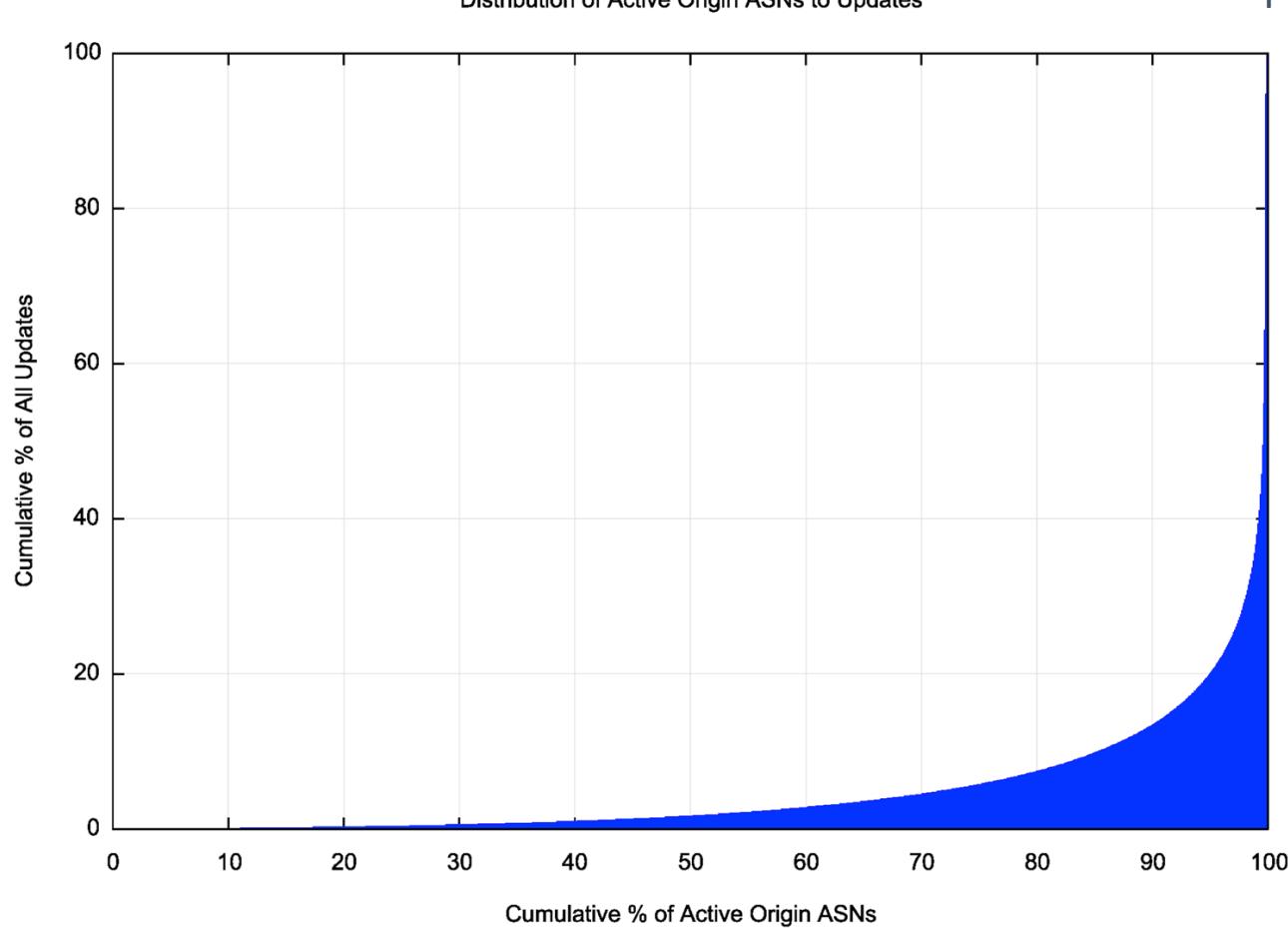
# Prefissi che ogni giorno vengono aggiornati

Immagine da  
apnic.net



# Chi manda più aggiornamenti?

Immagine da  
apnic.net



# Dimensione dei percorsi nel tempo

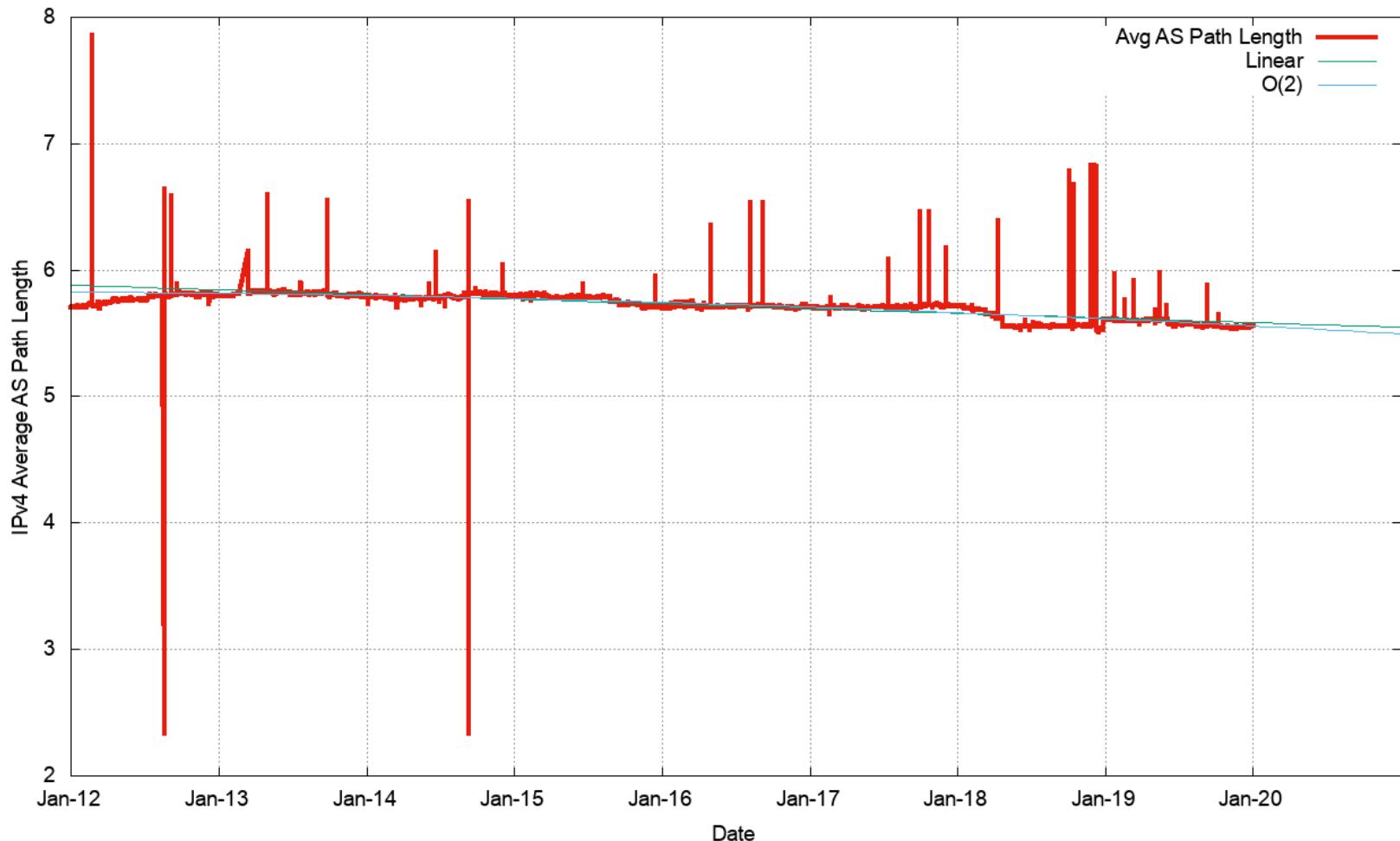
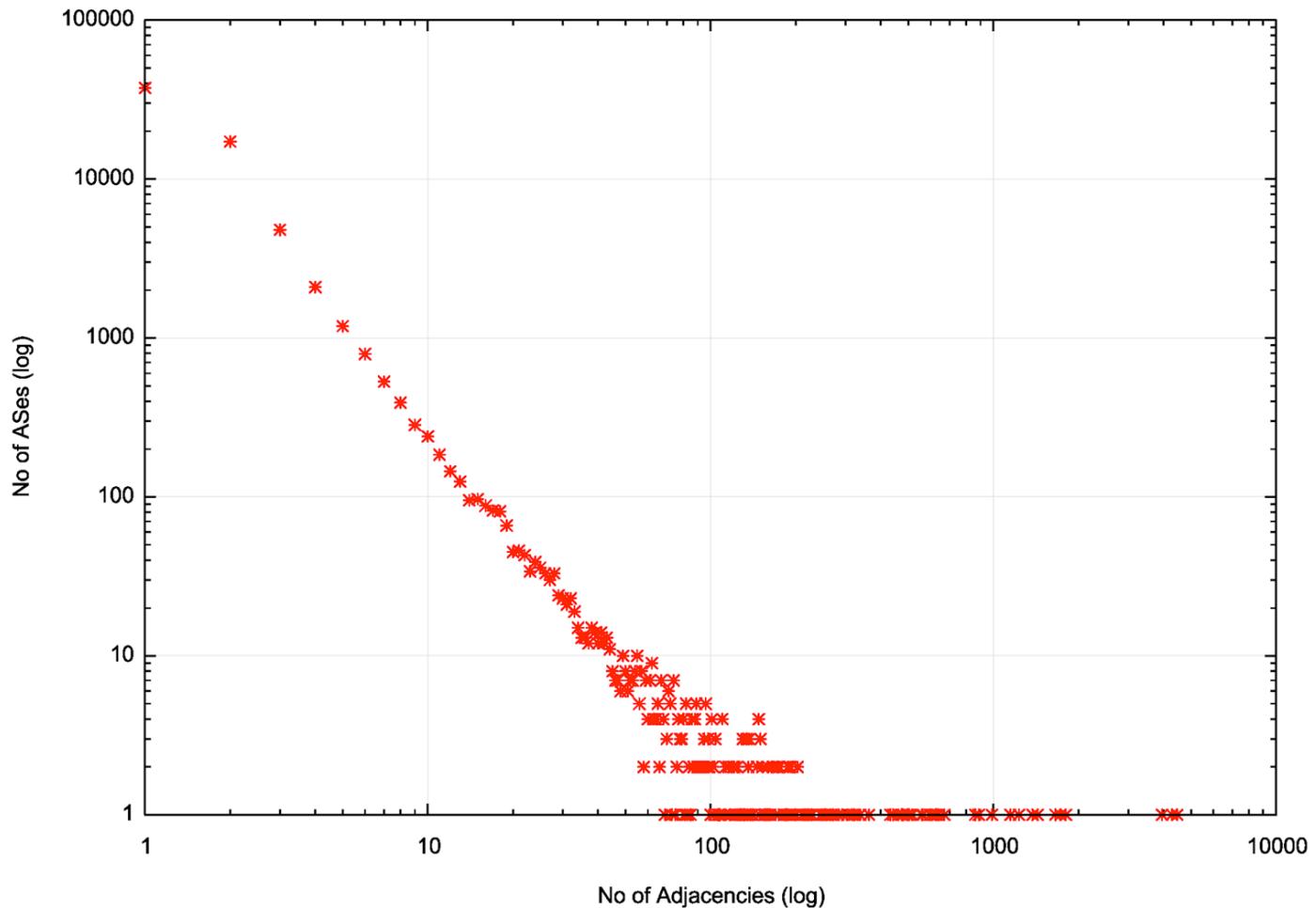


Immagine da  
apnic.net

# Dimensione delle adiacenze

Immagine da  
apnic.net



# Dimensione della routing table

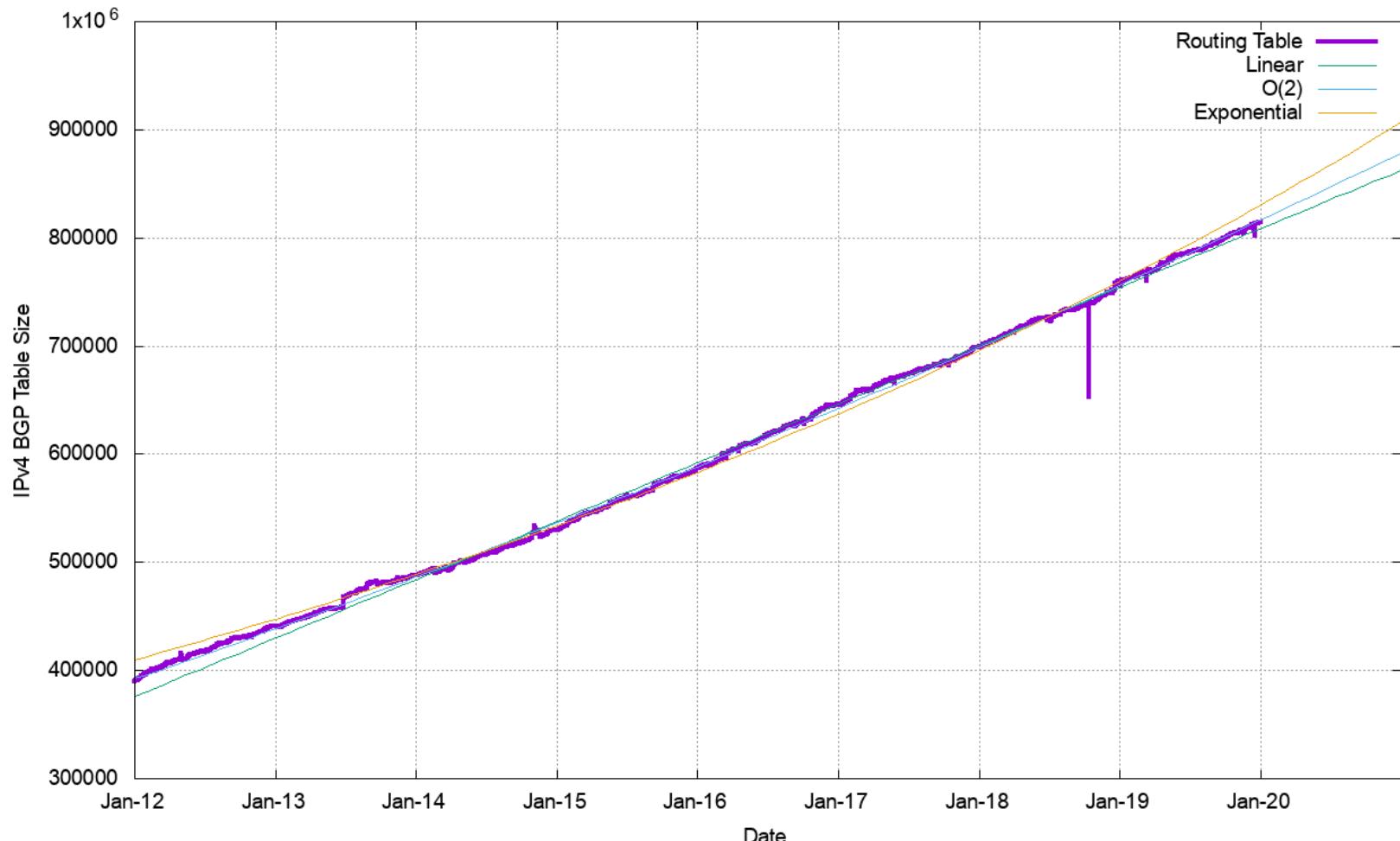


Immagine da  
apnic.net

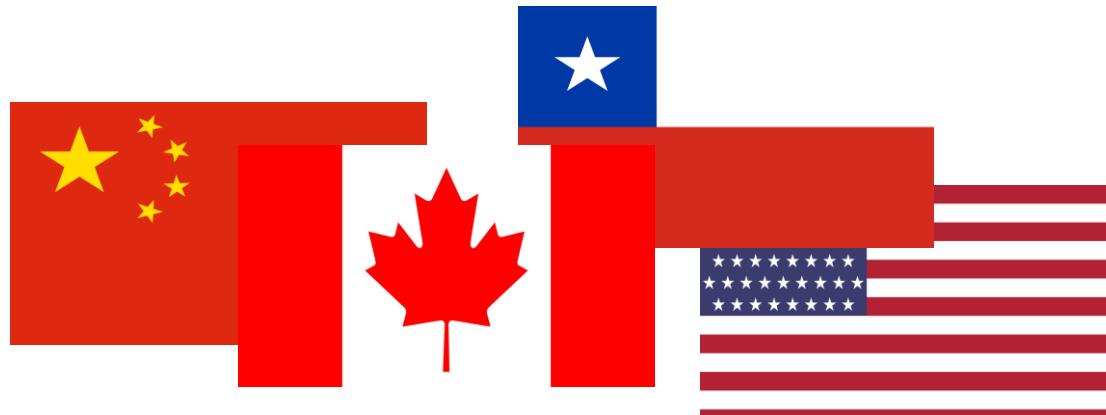
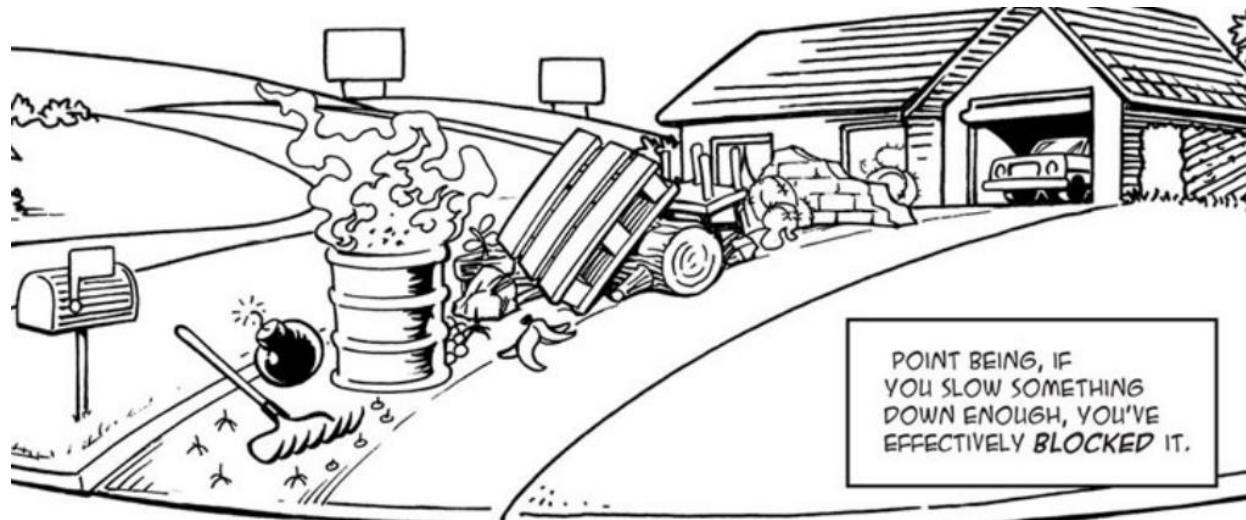
# Riassumendo

- Gli AS di Internet si interconnettono condividendo informazioni di raggiungibilità
- Gli AS formano un grafo gerarchico, e possono essere di solo transito
- Ogni AS è autonomo e può avere molteplici punti di ingresso e uscita
- **Il Best Path di BGP è policy-driven, ed è ottimo secondo le politiche commerciali e di traffic engineering degli operatori, non secondo le metriche di performance**
- BGP è un protocollo path vector (condivide anche il percorso per raggiungere una destinazione)
- BGP può inserire molteplici filtri a livelli differenti della catena decisionale del protocollo.
- BGP è un protocollo vasto e con moltissime sfaccettature

# Network neutrality

“Ogni sito web è uguale agli altri e non è trattato in modo differente dagli altri”

- Quanto siamo liberi quando navighiamo?  
Quanto invece siamo condizionati dalle decisioni prese da altri?
  
- Quanto i contratti economici degli operatori influiscono sulle nostre ricerche?
  
- Gli ISP possono discriminare informazioni che non portano guadagni sufficienti



# Network (un)neutrality examples

The screenshot shows the header of the ExtremeTech website. The logo 'EXTREMETECH' is at the top left, followed by a search bar 'Search Extremetech'. Below the header are navigation links for 'Computing', 'Phones', 'Security', 'Gaming', 'Science', and 'Space'. The main headline reads 'Verizon caught throttling Netflix traffic even after its pays for more bandwidth'. Below the headline is a sub-headline 'By Joel Hruska on July 20, 2014 at 7:08 am | [Comments](#)'. At the bottom of the screenshot are social sharing icons for Facebook, Twitter, Google+, LinkedIn, YouTube, and Flipboard.

## Verizon caught throttling Netflix traffic even after its pays for more bandwidth

By Joel Hruska on July 20, 2014 at 7:08 am | [Comments](#)



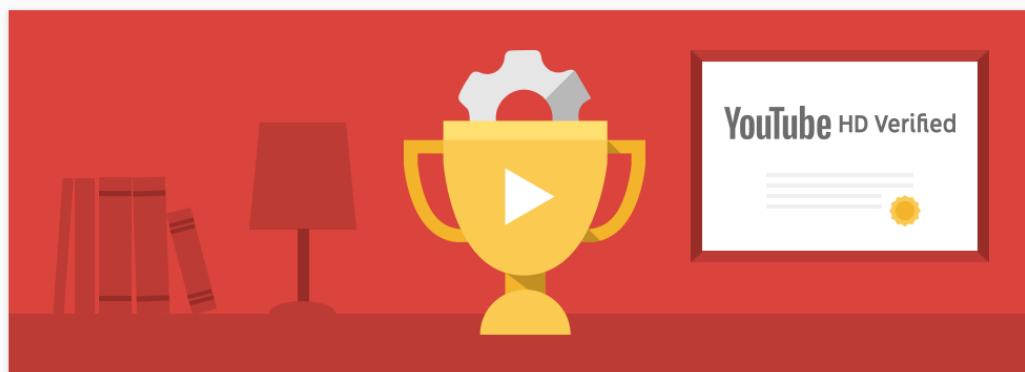
Netflix has recently begun paying both Comcast and Verizon to improve network performance and carry its video streams at higher bandwidths, but so far only Comcast has reciprocated with better service. Not only has Verizon's performance become dramatically worse, the company has continued to try and foist the blame for the problem on

Netflix, claiming that the online streaming giant is deliberately degrading performance by attempting to stuff data down specific congested Verizon pipes.

Unfortunately, a growing body of evidence suggests this isn't true. Verizon claims that Netflix "chose to attempt to deliver that traffic to Verizon through a few third-party transit providers with limited capacity over connections specifically to be used only for balanced traffic flows." Yesterday, backbone provider Level 3 posted a response to Verizon's claims, noting that in Los Angeles, the peering between Verizon and Level 3 is literally accomplished by connecting four 10 GigE ports between a pair of routers. What does that connection look like?

## The Methodology

How we verify that an Internet Provider can consistently serve YouTube in HD.



## The Ratings



### What do the ratings mean?

The ratings represent the video streaming quality you can expect (at least 90% of the time) when you watch YouTube on an Internet Service Provider in a specific area.

- **YouTube HD Verified:** Users on YouTube HD Verified networks should expect smooth playback most of the time when watching high-definition YouTube videos (720p and above).

- **Standard Definition:** Users on networks rated as Standard Definition should expect smooth playback on standard-definition YouTube videos (360p) and may experience occasional interruptions on high-definition YouTube videos (720p and above).