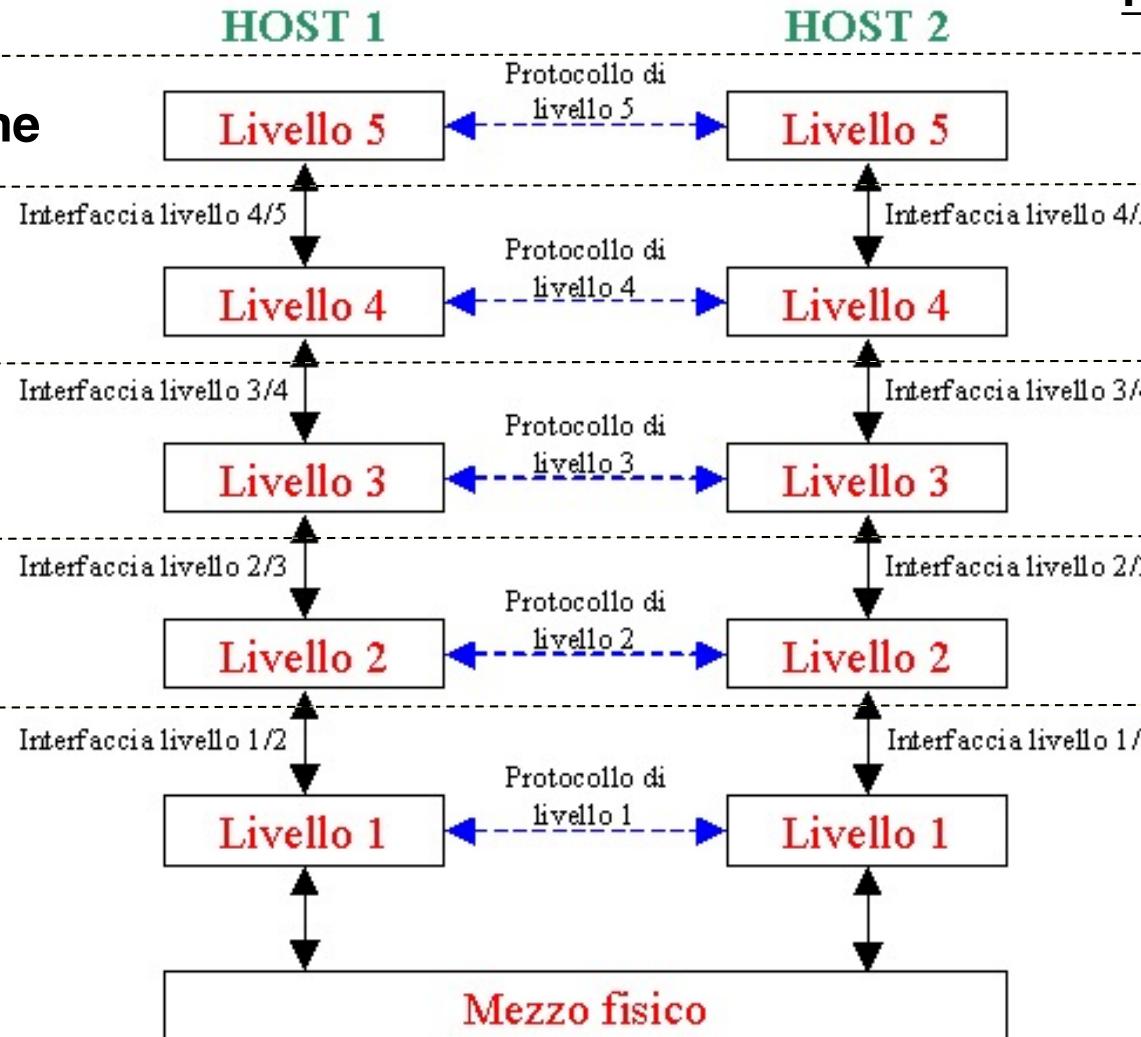
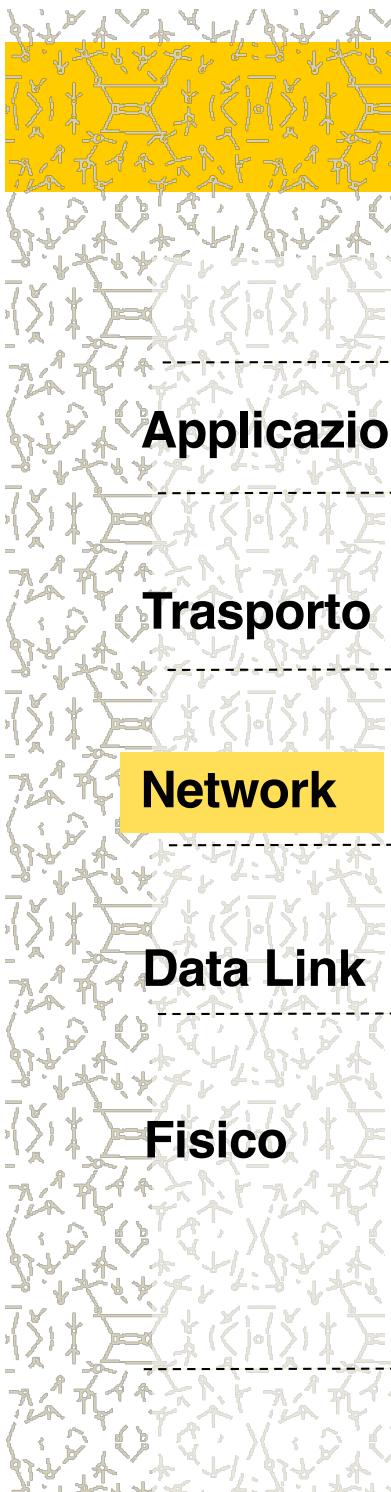


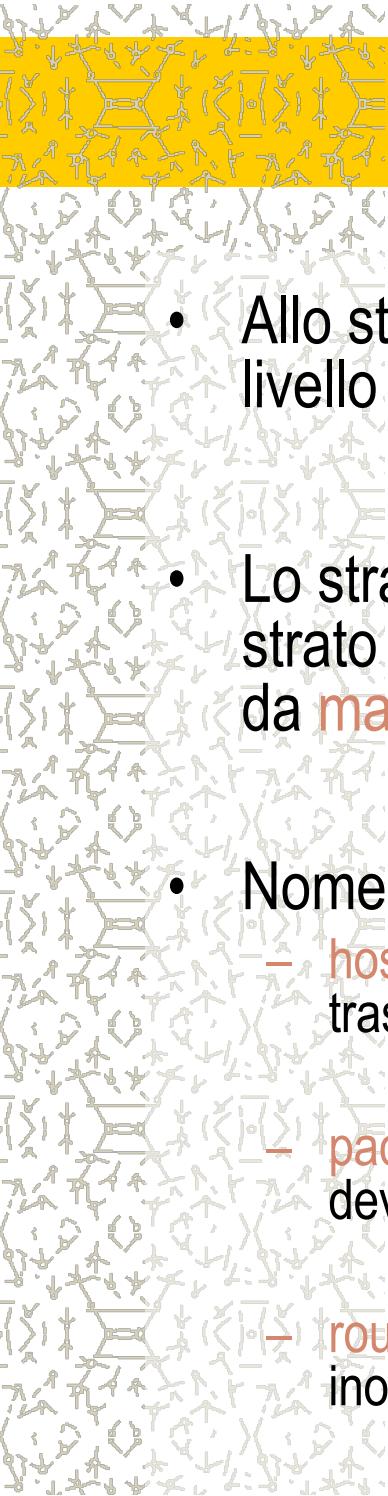
Reti di Calcolatori

Networking Basics



Modello ISO-OSI



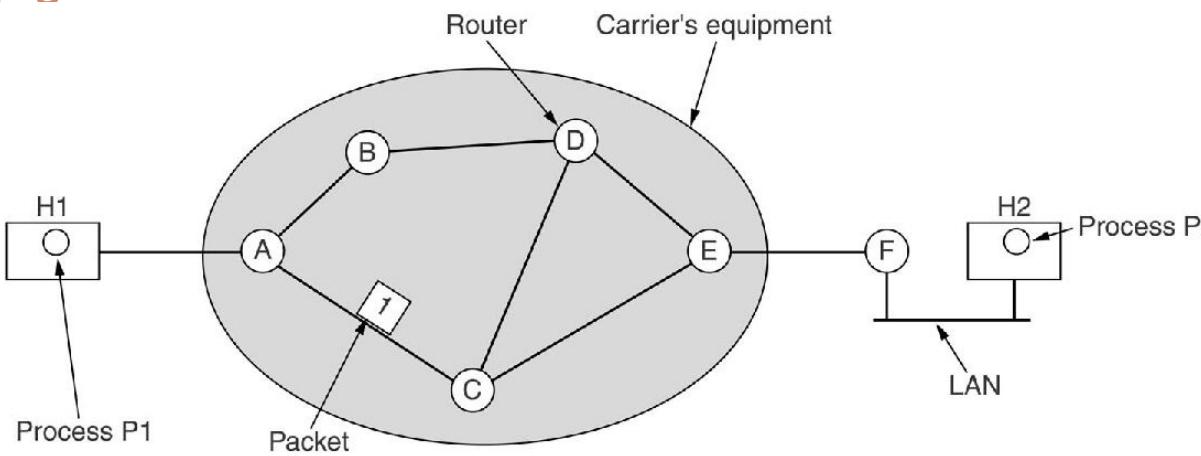


Funzioni dello strato di rete

- Allo strato di trasporto la comunicazione tra i processi peer di livello 4 **deve apparire** come una comunicazione **punto-punto**
- Lo strato di rete ha quindi come funzione quella di fornire allo strato di trasporto un servizio per la consegna dei dati in modo da **mascherare** l'infrastruttura della rete (la **sottorete**)
- Nomenclatura:
 - **host** o **end-node**: stazione su cui opera lo **strato di trasporto** che deve trasmettere o ricevere i dati utilizzando il servizio dello strato di rete
 - **pacchetto**: insieme di **dati+header+trailer** che lo strato di rete costruisce e deve trasmettere fino a destinazione
 - **router**: stazione **intermedia** che opera a **livello 3**, che riceve i pacchetti e li inoltra attraverso la (sotto)rete

Funzioni dello strato di rete (cont.)

- In generale due host sono separati da un **certo numero** di nodi, interconnessi da **svariate** linee
- Spesso sono possibili **più tragitti** tra i due nodi (ad esempio nelle reti magliate)
- Potenzialmente i nodi sono separati da reti funzionanti con **tecniche differenti**





Funzioni dello strato di rete (cont.)

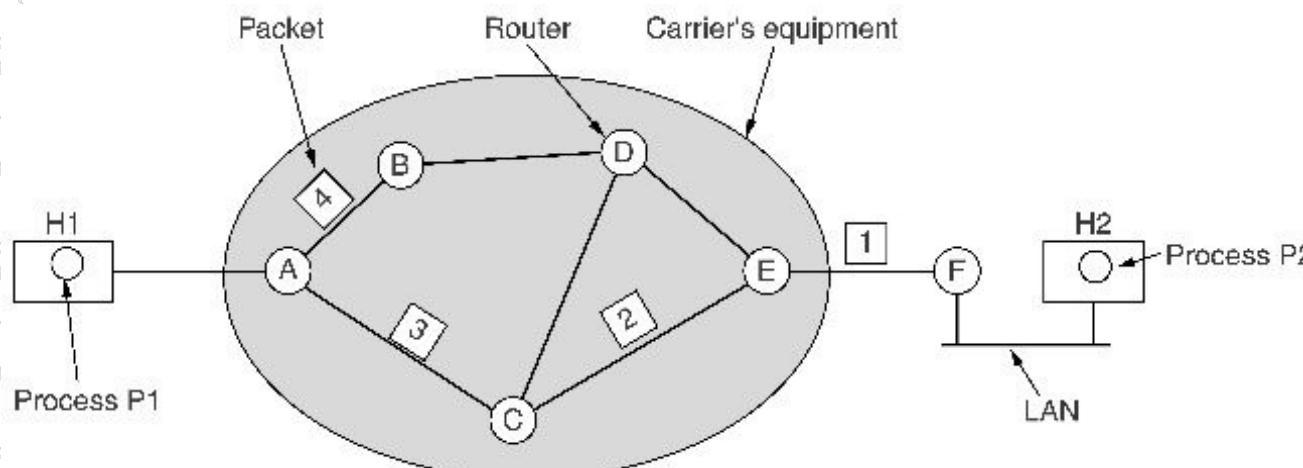
- Lo strato di rete dovrà quindi occuparsi dei seguenti argomenti
 - determinare quale tracitto tra quelli disponibili dovranno seguire i dati (**instradamento, routing**)
 - questo può richiedere che lo strato di rete conosca la **topologia** della rete
 - reagire a **modifiche** di topologie della rete
 - se esiste un meccanismo **dinamico** per l'apprendimento della topologia, questo permetterà di apprenderne anche le modifiche nel tempo
 - evitare di **sovrafficare** linee quando sono disponibili percorsi alternativi (**congestione**)
 - risolvere i problemi connessi al transito attraverso reti differenti (**internetworking**)

Connection Oriented o Connectionless?

- Inizialmente OSI prevedeva che lo strato di rete fornisse solo un servizio **connection oriented**
 - in modo analogo al funzionamento del servizio telefonico
 - questo servizio, caldeggiato dalle compagnie telefoniche, permette di operare **fatturazione a tempo** e di offrire **servizi di qualità** riservando le risorse a priori
- In seguito c'è stata forte richiesta di introdurre nello standard anche un servizio **connection less**, e così è stato fatto
 - l'**inaffidabilità intrinseca** della sottorete richiede un meccanismo più flessibile per il recapito dei dati
 - comunque lo strato di trasporto dovrà occuparsi della integrità dei dati, **inutile** farlo anche a livello di rete

Instradamento connectionless

- Il servizio senza connessione richiede che i pacchetti siano instradati **indipendentemente** uno dall'altro
- Generalmente un router dispone di una **tabella** che definisce su quale **linea di uscita** debba essere trasmesso un pacchetto in base alla **destinazione finale**
 - il router riceve il pacchetto, lo memorizza per analizzarlo, quindi lo trasmette in base alla tabella (store and forward)
- **Ogni** pacchetto deve quindi contenere l'indirizzo di destinazione
- Poichè le tabelle possono **modificarsi nel tempo**, non è detto che tutti i pacchetti seguano la **stessa strada**

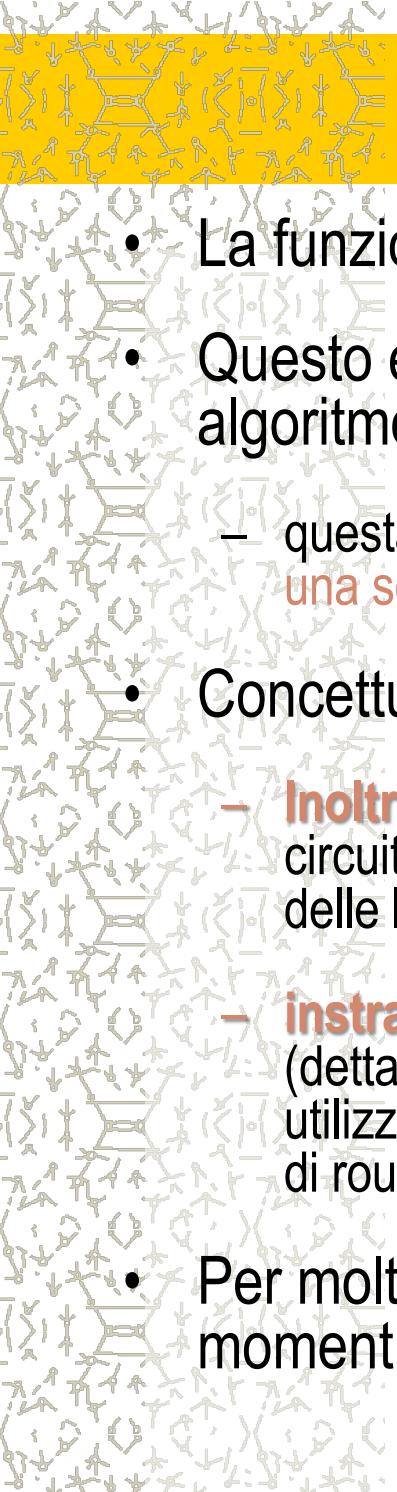


Instradamento connection oriented

- L'idea di base è di associare ad una connessione un **circuito virtuale** nella sottorete
- Si definisce **a priori** – durante la fase di inizializzazione della connessione – la **sequenza di router** che i pacchetti dovranno attraversare
- Tutti i pacchetti appartenenti alla **stessa connessione** seguiranno la **stessa strada**
- L'instradamento del pacchetto sarà quindi fatto in base alla sua appartenenza ad una connessione e non alla sua destinazione finale
- L'intestazione del pacchetto sarà **più semplice**, dovendo contenere solo l'identificativo della connessione
- La connessione potrà essere stabilita in modo da **garantire** le risorse necessarie alla trasmissione, rendendola più affidabile
- Una connessione successiva tra gli stessi nodi potrebbe definire un circuito virtuale **differente** dal precedente

Connectionless vs. Connection Oriented

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



Instradamento ed inoltro

- La funzione principale dello strato di rete è l'istradamento (**routing**)
- Questo è il processo che permette al router di **scegliere** – tramite un algoritmo – la **linea di uscita** verso cui instradare i dati
 - questa operazione sarà **ripetuta** per ogni pacchetto nel caso connectionless, o **una sola volta** all'inizio per l'istradamento connection oriented
- Concettualmente si possono distinguere due operazioni:
 - **Inoltro (forwarding)**: il processo che, in base all'indirizzo di destinazione o al circuito virtuale, sceglie la linea di uscita in funzione di dati noti (tabelle, stato delle linee, ...)
 - **instradamento**: il processo di **creazione ed aggiornamento** della tabella (detta tabella di routing) che associa alla destinazione la linea di uscita da utilizzare; questa operazione viene eseguita in base ad algoritmi detti algoritmi di routing
- Per molti algoritmi queste sono **operazioni distinte** eseguite in momenti diversi da processi distinti all'interno dello strato di rete



Caratteristiche di un algoritmo di routing

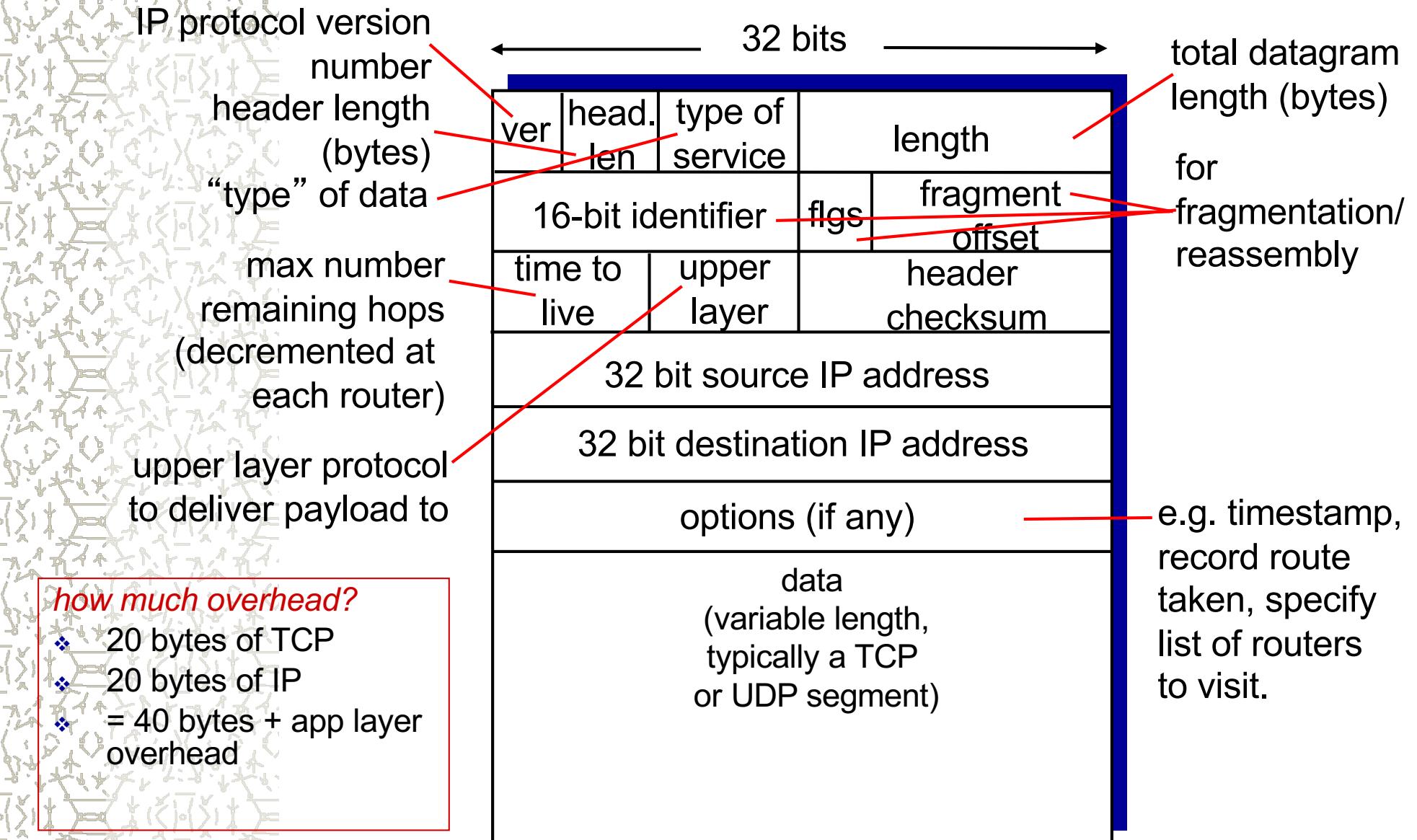
- E' desiderabile che un algoritmo di routing abbia le seguenti caratteristiche
 - **correttezza**: ovvio
 - **semplicità**: meno soggetto ad errori in implementazione o in esecuzione
 - **robustezza**: la rete non e' stabile, e l'algoritmo deve poter fare fronte alle modifiche di topologia
 - **stabilità**: convergenza verso l'equilibrio
 - **imparzialità**: servire qualunque tragitto possibile senza penalizzare nessuno
 - **ottimizzazione**: efficienza globale



Internet Protocol (IP)

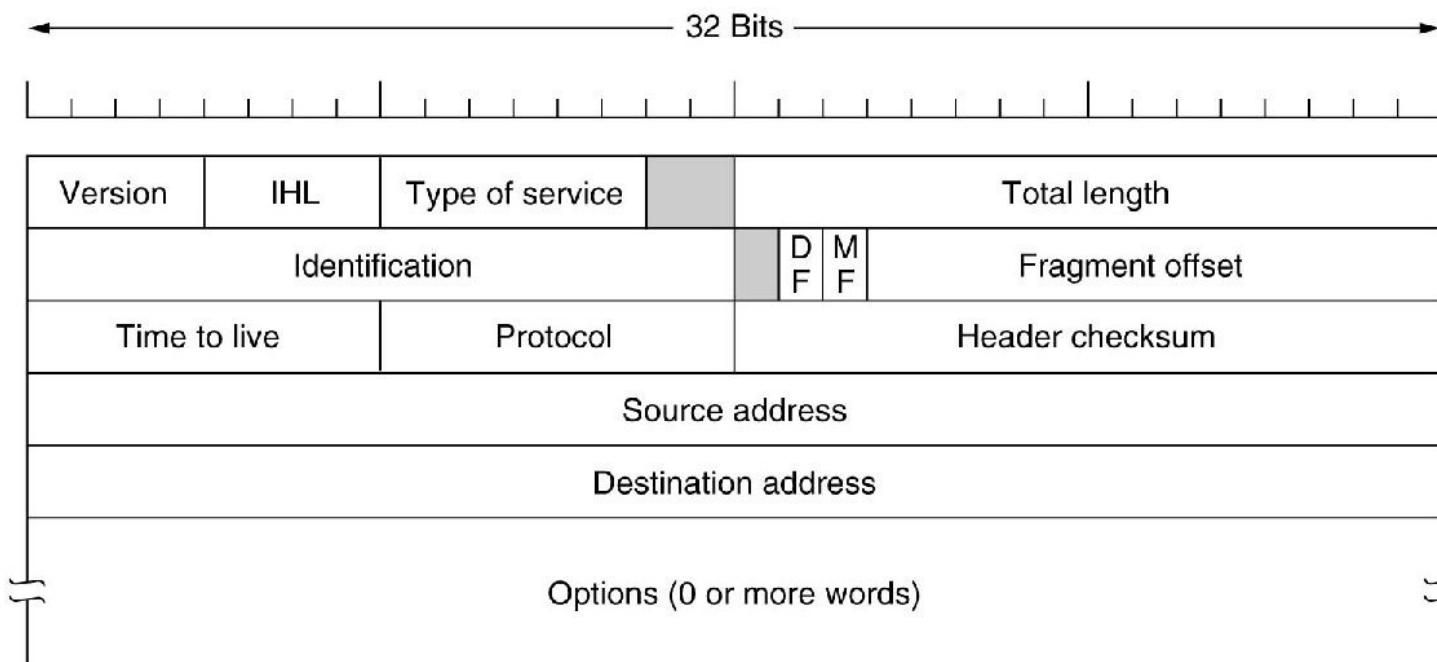
- IP è il protocollo di rete della suite TCP/IP
- Definito negli RFC 791 e 1122
- Dall'RFC 791:
 - IP ha la funzione di **recapitare** un insieme di bit (**internet datagram**) dalla sorgente alla destinazione attraverso un sistema di **reti interconnesse**
 - Non sono previsti meccanismi di **affidabilità**, **controllo di flusso**, **sequenzialità**, **rilevazione** o **correzione di errore**
 - Il recapito viene operato **direttamente** se la destinazione appartiene alla **stessa rete** della sorgente, attraverso un **sistema intermedio** (router) altrimenti
 - Se possibile il datagramma viaggia **intero**, altrimenti viene spezzato in **più parti**, ciascuna trasportata poi **individualmente**; in questo caso il datagramma viene riassemblato **a destinazione**
 - IP si preoccupa di trasmettere il datagramma da un host all'altro, fino alla destinazione, una rete alla volta
- Questa definizione corrisponde ad un protocollo che fornisce un servizio **connection less inaffidabile**

IP datagram format



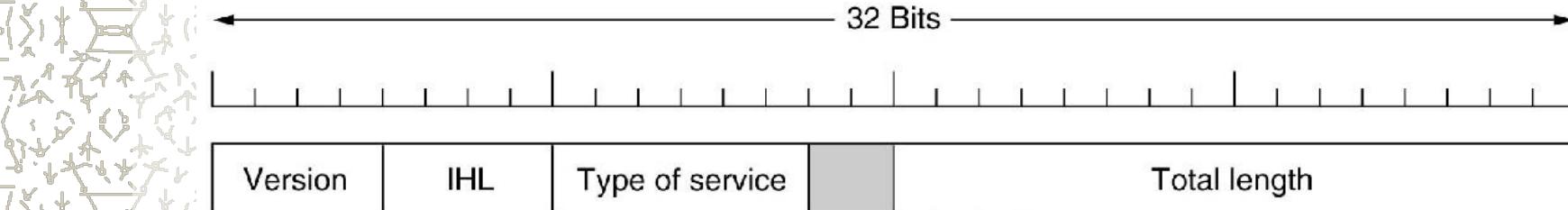
Pacchetto IP

- Il pacchetto IP è costituito da un header di lunghezza fissa **20 byte**, più una parte **opzionale** (fino a 40 byte)
- Il campo **version** contiene il numero identificativo della **versione di IP** (per IPv4 è 4, per IPv6 è 6)



Pacchetto IP (cont.)

- Il campo IHL (4 bit) contiene la **lunghezza dell'header** in parole di 32 bit (quindi un massimo di 60 byte complessivi)
- Il campo **type-of-service** serve ad indicare diverse **classi di servizio** (precedenza del pacchetto, basso ritardo, etc.), di solito ignorato dai router
- **total-length** indica la lunghezza totale del pacchetto in byte, che ha un valore massimo di **65535**

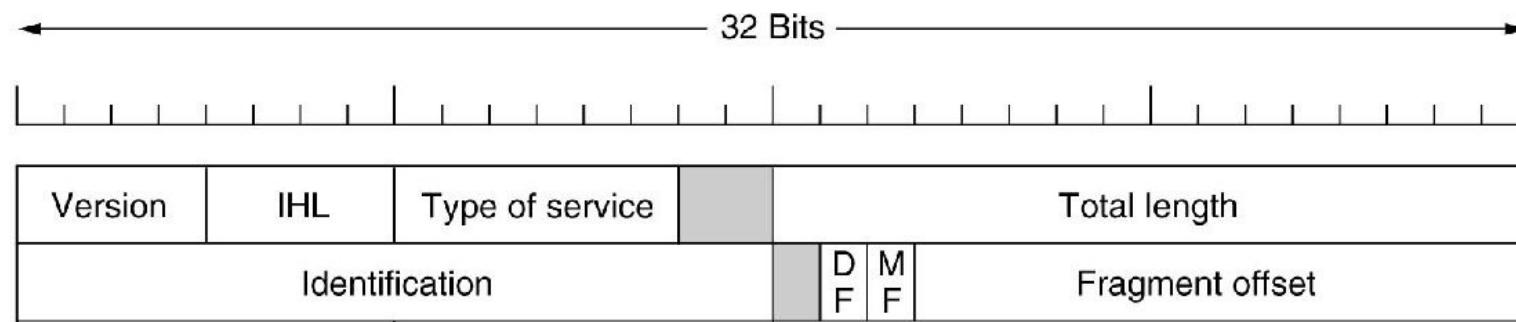
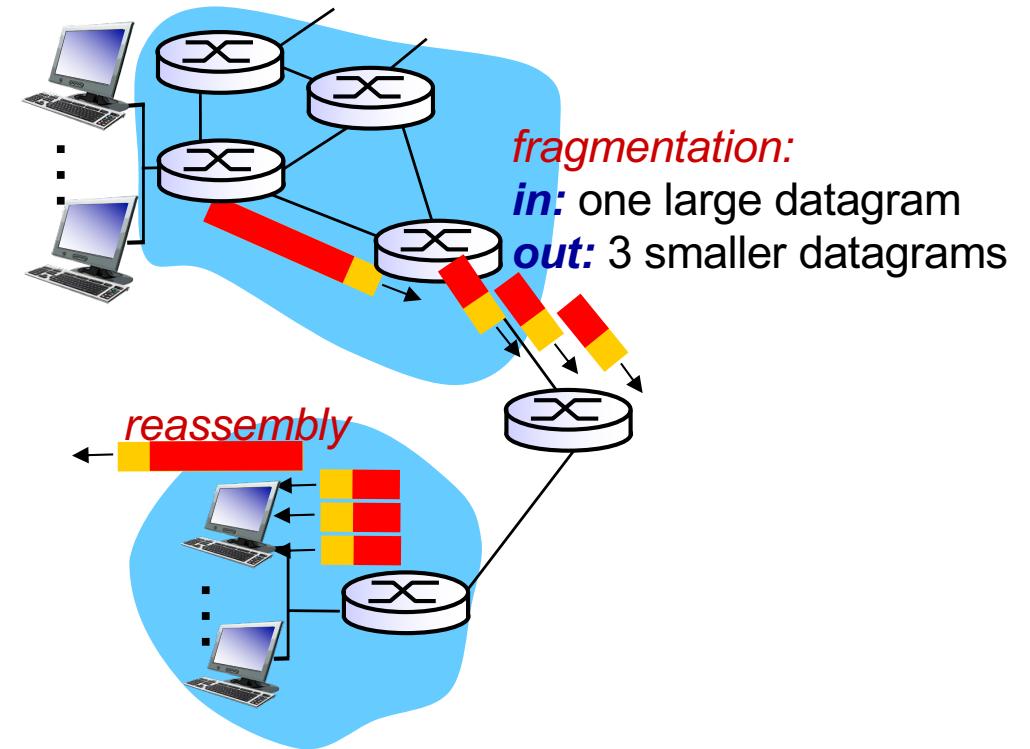


Pacchetto IP: frammentazione

- I campi **identification**, **DF**, **MF** e **fragment-offset** sono dedicati alla **frammentazione**:

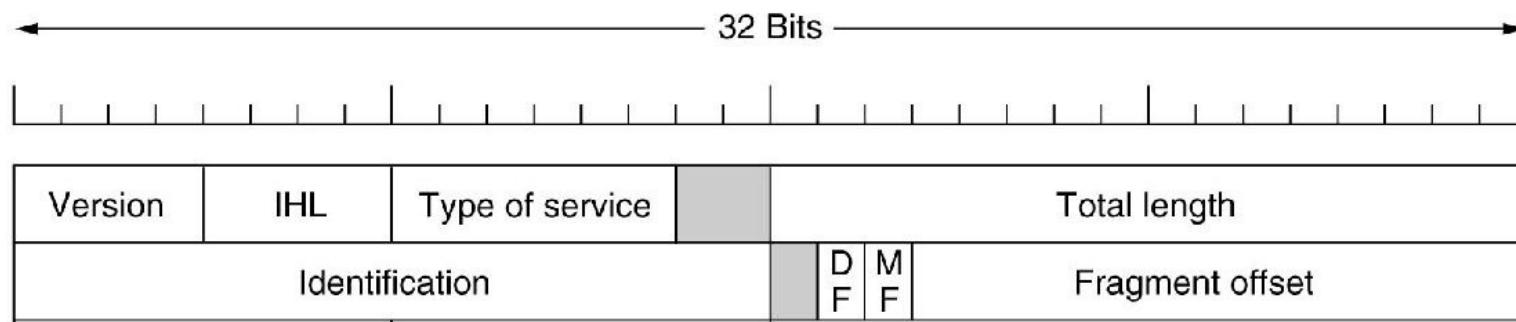
Alcuni protocolli dello strato di Collegamento possono trasportare pacchetti grandi, altri solo piccoli.
(Ethernet ≤ 1500 bytes)

Il datagramma IP di grandi dimensioni viene diviso ("frammentato") in datagrammi più piccoli.



Pacchetto IP: frammentazione

- I campi **identification**, **DF**, **MF** e **fragment-offset** sono dedicati alla **frammentazione**:
 - ogni datagramma IP inviato da una sorgente ha un **numero identificativo** differente dagli altri, riportato nel campo **identification**
 - se un datagramma **viene frammentato**, ogni frammento contiene nel campo **identification** lo **stesso valore**, mentre nel campo **fragment-offset** viene indicata la **posizione del primo byte** del frammento rispetto all'inizio del datagramma, espressa in **multipli di 8 byte**
 - in base all'identification la destinazione può **raggruppare** i diversi frammenti, in base a **total-length** ed agli **offset**, la destinazione può valutare se si fossero **persi frammenti** del datagramma
 - il bit **MF** (**More Fragments**) viene impostato a 0 **nell'ultimo frammento** (o nel datagramma se non viene frammentato), ad 1 altrimenti
 - il bit **DF** (**Don't Fragment**) viene impostato ad 1 se il datagramma non deve essere frammentato



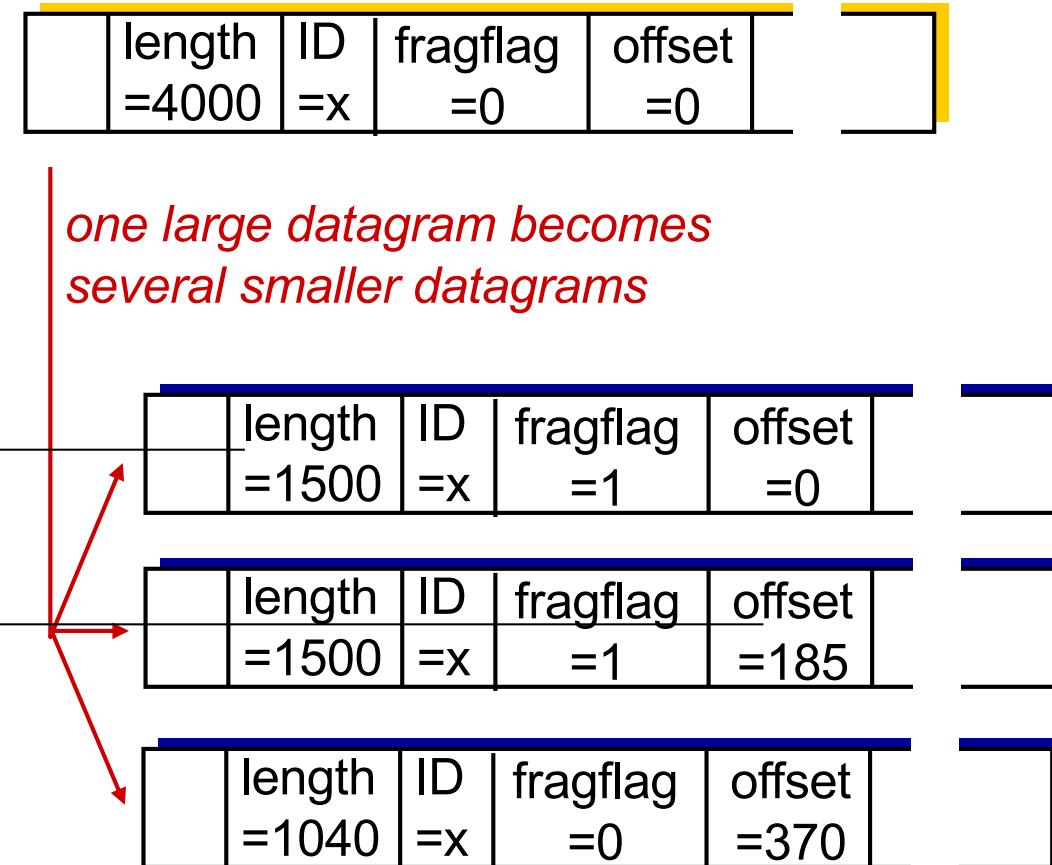
IP fragmentation, reassembly

example:

- 4000 byte datagram
- MTU = 1500 bytes

1480 bytes in
data field

offset =
 $1480/8$



MTU: Maximum Transfer Unit

Pacchetto IP (cont.)

- Il campo time-to-live è un contatore che viene **decrementato** via via che il pacchetto viaggia in rete
- Il pacchetto viene **buttato via** quando il time-to-live arriva a **zero**
- Normalmente dovrebbe essere decrementato ad **ogni secondo** o ad **ogni hop**, ma talvolta i router non tengono conto del tempo
- Il campo protocol indica il protocollo di **livello superiore** a cui sono destinati i dati del pacchetto
 - vi sono diversi protocolli che possono fare uso di IP, come **TCP** (6) ed **UDP** (17), ma anche **ICMP** (1) ed altri
- Il campo checksum contiene un codice CRC a 16 bit relativo al solo header
 - viene controllato solo l'header per motivi di **performance**, secondo la logica di TCP/IP che delega il controllo della affidabilità ai **livelli superiori**
 - il campo checksum viene **ricalcolato** ad **ogni hop**, in quanto alcuni dei campi precedenti (come quelli relativi alla frammentazione o time-to-live) **cambiano** durante il trasferimento del pacchetto
- Source e destination address contengono gli indirizzi a 32 bit del sorgente e del destinatario del pacchetto

Pacchetto IP (cont.)

- Le opzioni aggiuntive dell'header vengono utilizzate, se necessario, per svariati motivi, tra cui
 - **security options**: classifica il pacchetto da “non classificato” a “top secret”; router che onorano questi campi possono essere indotti a instradamenti differenti in base a questa opzione
 - **record route**: istruisce i router a registrare il loro indirizzo nei successivi campi opzionali via via che il pacchetto transita in rete (usato per motivi di debug del routing)
 - **loose** o **strict source routing**: istruisce i router a seguire un instradamento specifico definito dalla sorgente (che riempie i campi opzionali con gli indirizzi dei router che il pacchetto deve attraversare)
- Sono disponibili **40 byte** per queste opzioni
 - ogni campo inizia con un ottetto che definisce il tipo di estensione, seguito eventualmente da uno o più ottetti contenenti le informazioni relative (indirizzi IP, timestamp, ...)

Indirizzamento IP

- Per poter **identificare** il destinatario, ogni host e router devono avere un **indirizzo (IP) univoco**, che distingue:
 - la **rete di appartenenza** dalle altre, e
 - **l'host** dagli altri host appartenenti alla stessa rete
- l'indirizzamento IP è quindi **gerarchico**, a due livelli:
 - **indirizzo di rete**, ed
 - **indirizzo di host**,
- a differenza di quello Ethernet che è piatto



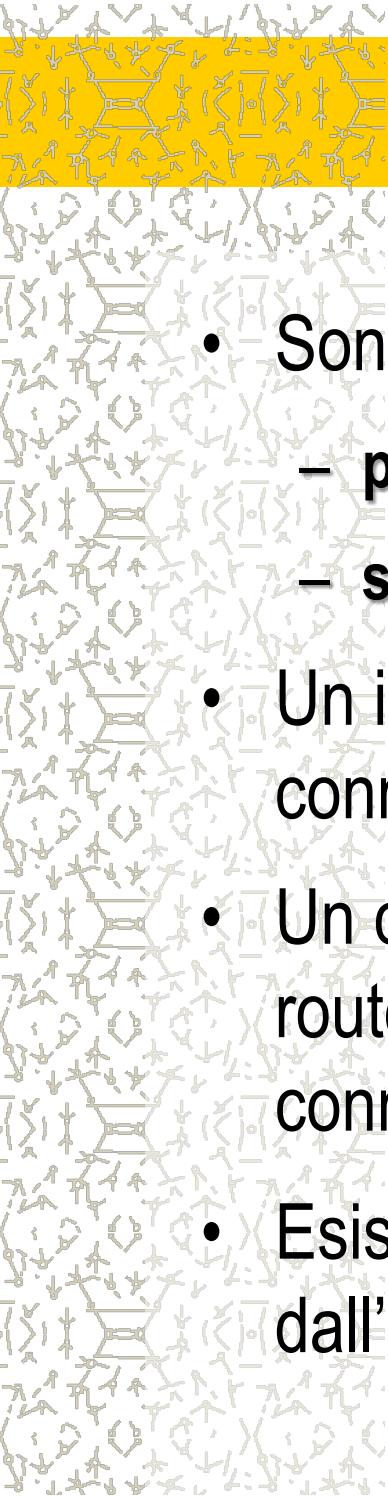
Indirizzamento IP

- In realtà **ogni interfaccia di rete** (cioè ogni connessione ad una rete) deve avere un indirizzo IP
- generalmente i PC hanno **una sola** interfaccia di rete,
- ma i router (sempre) o i server di grosse dimensioni (talvolta) hanno **più interfacce** di rete:
 - ciascuna di queste deve avere un indirizzo IP
- tutti i nodi IP hanno un ulteriore indirizzo:
detto **loopback**,
che rappresenta un indirizzo **fittizio** indicante “**se stesso**”, ed utilizzato per motivi di diagnostica o per simulare connessioni di rete di un host con se stesso



Struttura dell'indirizzo IP

- L'indirizzo IP è costituito da **32 bit**, o 4 byte, generalmente rappresentati da 4 **numeri decimali** di valore compreso tra 0 e 255, separati da un punto
 - ad esempio: **10.103.0.21**
- Questo indirizzo contiene:
 - una parte che specifica la rete,
 - ed una parte che identifica l'host all'interno di quella rete



Indirizzi IP

- Sono divisi in due parti
 - **prefisso**: identifica la rete
 - **suffisso**: identifica host/interface
- Un indirizzo IP **non identifica** un computer, ma una connessione computer-rete.
- Un computer con connessioni multiple di rete (e.g., un router) ha assegnato un indirizzo IP per ogni connessione
- Esiste una totale indipendenza dell'indirizzo IP dall'indirizzamento hardware (MAC)

Indirizzi IP

Dotted
Decimal

32 bits

Network

Host

Massimo

255

255

255

255

Modo sintetico per esprimere indirizzi IP: **rappresentare ogni ottetto in decimale usando punti come separatori.**

Esempio: 12.34.158.5

12

34

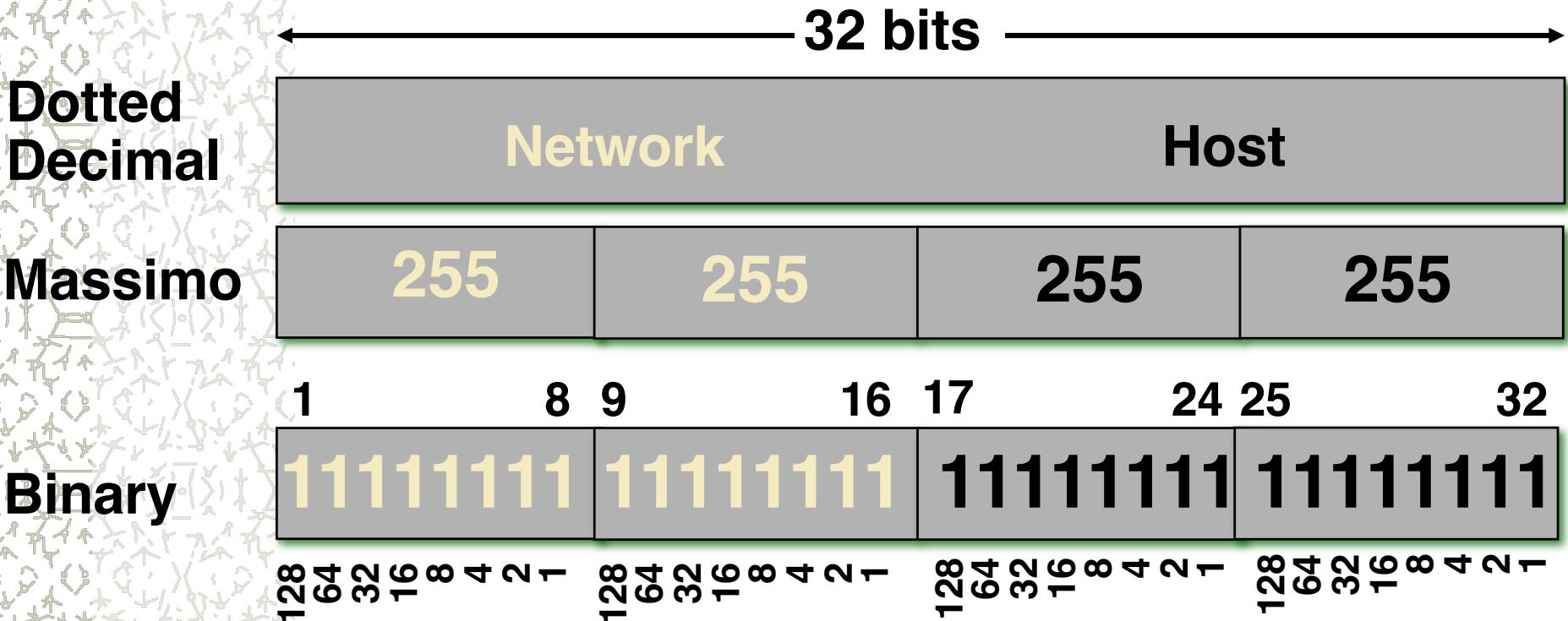
158

5

00001100 00100010 10011110 00000101

25

Indirizzi IP



Indirizzi IP

32 bits

Dotted
Decimal

Network

Host

Massimo

255

255

255

255

1

8

9

16

17

24 25

32

Binary

11111111

11111111

11111111

11111111

128 64 32 16 8 4 2 1

128 64 32 16 8 4 2 1

128 64 32 16 8 4 2 1

128 64 32 16 8 4 2 1

Example
Decimal

172

16

122

204

Example
Binary

10101100

00010000

01111010

11001100

Classi di indirizzi

Gli indirizzi IP sono raggruppati in diverse **categorie**, dette **classi**:

- **indirizzi di classe A**: tutti quelli che iniziano con **un bit 0**, cioè con il primo byte di valore compreso **tra 0 e 127**
 - gli indirizzi di classe A hanno il **primo byte** dedicato all'indirizzo di **rete**, i restanti all'indirizzo di host

Es.: l'indirizzo **20.9.0.200** individua l'host appartenente alla rete “**20**”, il cui indirizzo di host è **9.0.200**

- esistono quindi 125 network di classe A
(le reti 0, 10 e 127 non vengono utilizzate)
- ciascuna rete di classe A può indirizzare **2^{24}** host differenti
(quasi 17 milioni)

Classi di indirizzi

- indirizzi di classe B:

tutti quelli che iniziano con la sequenza di bit 10, cioè con il primo byte di valore compreso tra 128 e 191

- gli indirizzi di classe B hanno i primi due byte dedicati all'indirizzo di rete, i restanti due dedicati all'indirizzo di host

Es.: 131.154.10.21 indica l'host di indirizzo “10.21” appartenente alla rete “131.154”

- esistono quindi 16383 ($2^{14} - 1$) reti di classe B, ciascuna contenente 65533 host

Classi di indirizzi (cont.)

- **indirizzi di classe C:**

tutti quelli che iniziano con la sequenza di bit 110, cioè con il primo byte di valore compreso tra 192 e 223

- questi indirizzi hanno i primi tre byte dedicati alla rete, il quarto all'indirizzo di host (193.206.144.1 indica l'host "1" della rete "193.206.144")
- in classe C esistono circa 2 milioni di reti, ciascuna contenente al più 254 host

- **indirizzi di classe D:**

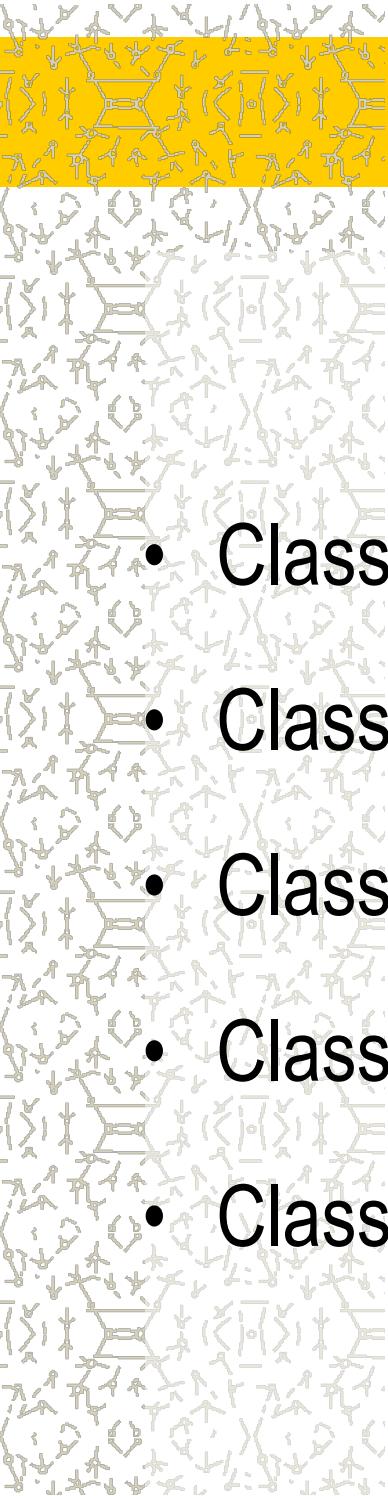
tutti quelli che iniziano con la sequenza di bit 1110, cioè con il primo byte compreso tra 224 e 239

– gli indirizzi di classe D sono dedicati all'indirizzamento dei gruppi multicast

- **indirizzi di classe E:**

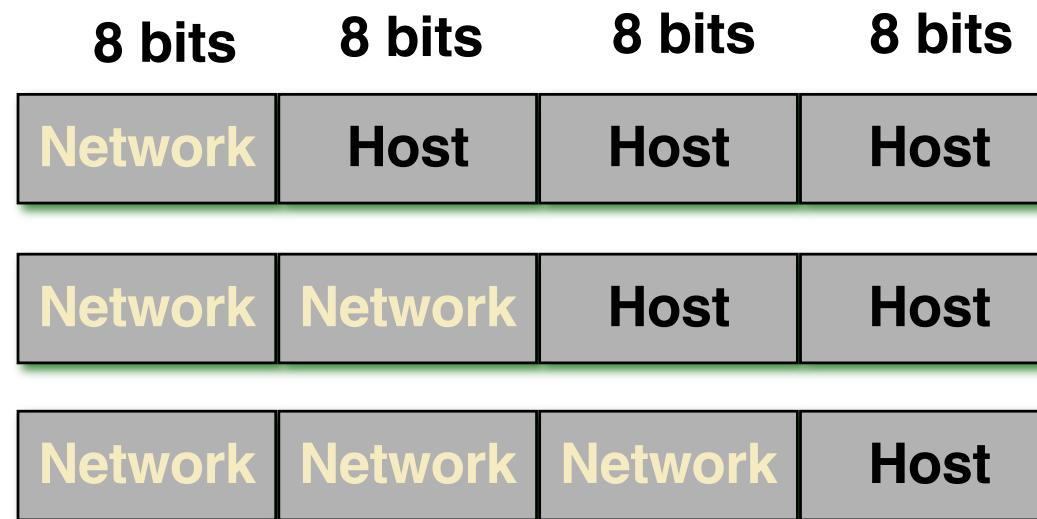
tutti quelli che iniziano con la sequenza di bit 1111, cioè con il primo byte compreso tra 240 e 255

– gli indirizzi di classe E sono dedicati ad utilizzi sperimentali, e non devono mai essere utilizzati come effettivo indirizzo di macchine sulla rete



Classi di indirizzi (cont.)

- Classe A:
- Classe B:
- Classe C:
- Classe D: Multicast
- Classe E: Research



I bit iniziali determinano la classe, che a sua volta determina il confine tra prefisso e suffisso.

Classi di indirizzi (cont.)

Bits:

Classe A:

Bits:

Classe B:

Bits:

Classe C:

Bits:

Classe D:

1	8 9	16 17	24 25	32
0NNNNNNN	Host	Host	Host	

Range (1-126)

1	8 9	16 17	24 25	32
10NNNNNN	Network	Host	Host	

Range (128-191)

1	8 9	16 17	24 25	32
110NNNNN	Network	Network	Host	

Range (192-223)

1	8 9	16 17	24 25	32
1110MMMM	Multicast Group	Multicast Group	Multicast Group	

Range (224-239)

32

Indirizzi speciali

- L'indirizzo contenente **tutti “0”** nel campo di host viene utilizzato **per indicare la rete**
 - l'indirizzo **10.0.0.0** indica la rete **“10”** (di classe A)
 - l'indirizzo **193.206.144.0** indica la rete **“193.206.144”** (di classe C)
- L'indirizzo **0.0.0.0** ha il significato di **“questo host di questa rete”**, e viene utilizzato dai calcolatori che, in **fase di boot**, non conoscono ancora il proprio indirizzo IP
- L'indirizzo IP con **tutti “0” nella parte di rete** ha il significato di **“questa rete”**
 - ad esempio, se l'host **193.206.144.10** vuole inviare sulla rete locale un pacchetto all'host **193.206.144.20**, può indirizzarlo a **0.0.0.20**
- Queste convenzioni spiegano perchè la rete di classe A: **0.0.0.0** non venga utilizzata come **rete indirizzabile** in IP: ad esempio, se così non fosse, il pacchetto indirizzato all'host 1 di una **qualsiasi rete** tramite la notazione **“questa rete”.1** non potrebbe essere **distinto** dal pacchetto indirizzato all'host 1 della rete **0.0.0.0**

Indirizzi speciali (cont.)

- L'indirizzo **255.255.255.255** (tutti bit 1) rappresenta l'indirizzo **broadcast** della rete locale direttamente connessa
 - è l'indirizzo utilizzato per inviare un pacchetto IP broadcast sulla **propria rete**
- L'indirizzo con tutti **1** nel campo **host** rappresenta l'indirizzo **broadcast** della rete specificata nel **campo rete**
 - ad esempio: l'indirizzo 130.90.255.255 indica l'indirizzo broadcast della rete 130.90.0.0
 - questo meccanismo permette di **indirizzare** un pacchetto a **tutti** gli host di una rete remota

Indirizzi dedicati a scopi speciali

- La rete di classe A **127.0.0.0** è dedicata all'interfaccia di **loopback**
 - l'interfaccia prende sempre l'indirizzo 127.0.0.1
- Tre **range** di indirizzamento stabiliti dalla RFC 1918 sono dedicati ad **indirizzi privati**
 - **10.0.0.0** (una rete di classe A)
 - da **172.16.0.0** a **172.31.0.0** (16 reti di classe B)
 - da **192.168.0.0** a **192.168.255.0** (256 reti di classe C)
- Il range **100.64.0.0/10** è riservato secondo la RFC 6598 alle comunicazioni fra provider e subscriber (in presenza di **carrier grade NAT**)
- Il range **198.18.0.0/15** è riservato secondo la RFC 2544 per i test di comunicazioni fra diverse internetworks
- Gli indirizzi privati possono essere utilizzati all'interno di una **rete privata**, ma **non devono** mai venire annunciati nelle **tabelle di routing** (così come la rete dell'interfaccia di loopback)
- Il routing verso le macchine ad indirizzo privato deve essere fatto dal router di interconnessione con la rete pubblica **ad insaputa** del resto della rete
 - lo scopo degli indirizzi privati è quello di poter utilizzare la tecnologia TCP/IP in una **realtà locale** senza dover necessariamente chiedere ed utilizzare **indirizzi pubblici**
 - una tecnica diffusa che fa uso di questi indirizzi per dare **connettività** senza sprecare indirizzi pubblici è il **NAT (Network Address Translation)** che vedremo in seguito



Assegnazione degli indirizzi in Internet

- Affinchè tutto funzioni correttamente in una internet gli indirizzi devono essere assegnati da una **autorità centrale** che garantisca innanzi tutto **l'unicità** delle assegnazioni
- Per **Internet** gli indirizzi sono assegnati dalla **ICANN (Internet Corporation for Assigned Names and Numbers)**
- La ICANN ha poi **delegato** organizzazioni **regionali** (Europa, Asia, America, ...) assegnando loro **gruppi di indirizzi** da riassegnare al loro interno
 - per l'Europa: RIPE NCC
- A loro volta le organizzazioni regionali **possono delegare** verso il basso, partizionando gli indirizzi a loro destinati dalla ICANN
 - in Italia: diverse istituzioni (ISP); per gli enti di ricerca si deve chiedere a GARR



Carenza di indirizzi

- Lo spazio di indirizzamento disponibile conta **due miliardi di indirizzi**, raggruppabili in **16500 reti** di **enormi** dimensioni e **2 milioni** di reti di **piccole** dimensioni
- Sembrava impossibile esaurire lo spazio di indirizzamento
- Tre i fattori che hanno determinato l'insorgere di carenza di indirizzi:
 - lo spazio di indirizzamento delle classi A, e spesso anche quello delle classi B, è **tropppo vasto**: nessuna rete può contenere 16 milioni di nodi distinti, o anche solo 65000
 - un **enorme numero** di indirizzi rimangono **inutilizzati**
 - una azienda o campus a cui è stata assegnata una classe A che deve **estendere** la sua rete per interconnettere **diversi dipartimenti** su reti locali distinte hanno bisogno di **altre reti**, benchè il numero di indirizzi disponibile ecceda di gran lunga la necessità di **indirizzi di host**
 - la connessione **punto-punto** tra **due router** richiede l'utilizzo di **una rete IP**, per la quale sono utilizzati solo due indirizzi
 - lo spazio di indirizzamento delle reti di **classe C** risulta **tropppo piccolo** con il crescere delle reti locali

Carenza di indirizzi

IANA Unallocated Address Pool Exhaustion:

03-Feb-2011

Projected RIR Address Pool Exhaustion Dates:

RIR

APNIC:

RIPE NCC:

LACNIC:

ARIN:

AFRINIC:

Projected Exhaustion Date

19-Apr-2011

14-Sep-2012

10-Jun-2014

05-Jul-2015

13-Apr-2019

Remaining Addresses in RIR Pool (/8s)

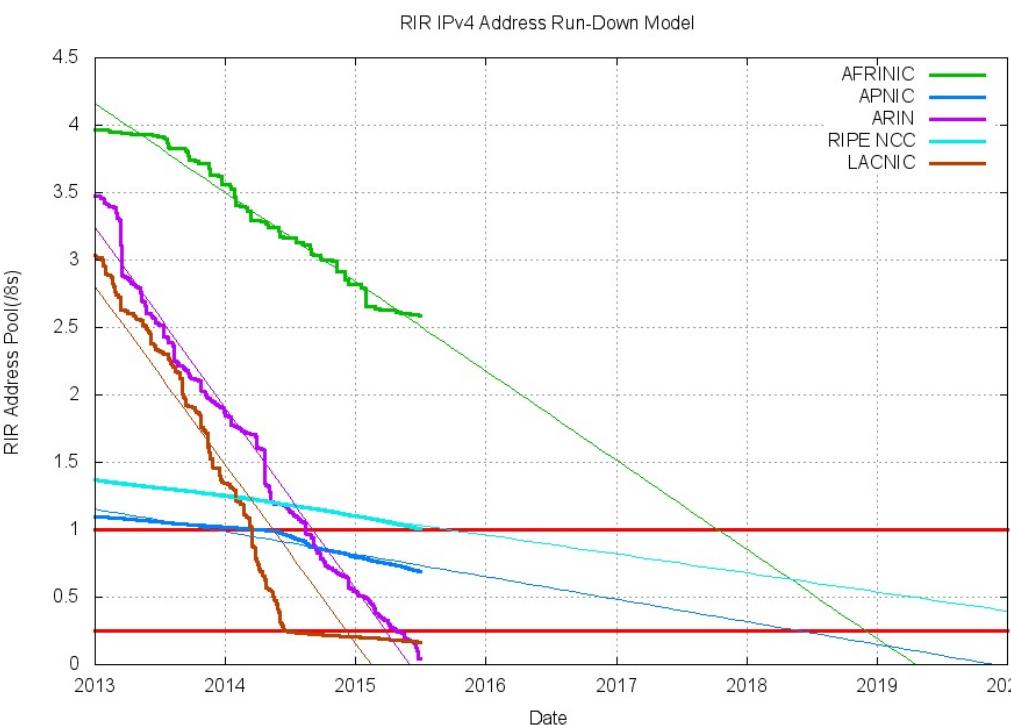
(actual) 0.6900

(actual) 1.0058

(actual) 0.1645

0.0089

2.5904



Calcolo indirizzi disponibili in una rete

Network

172

16

10101100 00010000

Host

0

0

165432109 87654321 N

00000000 00000000
00000000 00000001
00000000 00000011

1
2
3
⋮

11111111 11111101
11111111 11111110
11111111 11111111

65534
65535
65536
- 2

$$2^{N-2} = 2^{16-2} = 65534$$

65534

- Il massimo numero di hosts dipende dalla classe
 - Classe A grande
 - Classe B media
 - Classe C piccola

Esercizio

Address	Class	Network	Host
10.2.1.1			
128.63.2.100			
201.222.5.64			
192.6.141.2			
130.113.64.16			
256.241.20.10			

Soluzione

Address	Class	Network	Host
10.2.1.1	A	10.0.0.0	0.2.1.1
128.63.2.100	B	128.63.0.0	0.0.2.100
201.222.5.64	C	201.222.5.0	0.0.0.64
192.6.141.2	C	192.6.141.0	0.0.0.2
130.113.64.16	B	130.113.0.0	0.0.64.16

256.241.20.10 Non existent



IP addressing: CIDR

Per risolvere i problemi di carenza di indirizzi sono state applicate due soluzioni:

- **CIDR: Classless InterDomain Routing**
- **Subnetting**

IP addressing: CIDR

Per risolvere i problemi di carenza di indirizzi la IETF standardizzò il:

CIDR: Classless InterDomain Routing

- La parte di un indirizzo IP relativa alla rete può avere lunghezza a piacere (qualsiasi numero di bit), invece di essere costretta a 8, 16 o 24 bit.
- Il formato diventa: **a.b.c.d/x**
 - Con x il numero di bit più significativi dei 32 bit che rappresentano la parte dell'indirizzo relativa alla rete.



Subnetting

- Per risolvere i problemi di carenza di indirizzi di rete è stata sviluppata una tecnica detta **subnetting**:
- **Un'organizzazione può usare i bit rimasti per indirizzare gli hosts per creare altre reti, interne alla rete principale**



Es.: Questo bit potrebbe essere usato per creare altre 2 sottoreti:

- 200.23.16.0/24
- 200.23.17.0/24

Subnetting

- Per risolvere i problemi di carenza di indirizzi di rete è stata sviluppata una tecnica detta **subnetting**
 - un campus a cui è stata assegnata una rete di classe A **può suddividere** il suo campo di indirizzi in gruppi più piccoli, trattando ogni gruppo come se fosse **una "rete" a se stante**
 - ad esempio, se la rete assegnata è la **100.0.0.0**, il campus può dedicare gli indirizzi **100.1.0.0** ad un dipartimento, gli indirizzi **100.2.0.0** ad un secondo dipartimento e così via, trattando le due reti come se fossero reti di classe B
 - affinchè tutto funzioni a dovere, il router del campus dovrà **annunciare** verso l'esterno la **sola rete di classe A**, mentre internamente potrà trattare i vari pezzi come se fossero reti più piccole
 - **per implementare le sottoreti è necessario introdurre una informazione aggiuntiva agli indirizzi di rete, che specifichi quali bit dell'indirizzo definiscano l'indirizzo della (sotto)rete e quali definiscano l'indirizzo degli host**

Network mask

- Per identificare quali bit definiscono la rete e quali bit l'host, si utilizza una “maschera”,

anch'essa costituita da **32 bit**, col significato seguente:

- se un bit della maschera vale **1**, il **corrispondente bit** dell'indirizzo fa parte dell'indirizzo della **rete**
- se un bit della maschera vale **0**, il **corrispondente bit** dell'indirizzo fa parte dell'indirizzo di **host**
- Con questa convenzione, gli

indirizzi di **classe A** hanno maschera **255.0.0.0**,

quelli di **classe B** hanno maschera **255.255.0.0**,

quelli di **classe C** hanno maschera **255.255.255.0**

Network mask e Subnetting

- Utilizzando opportunamente le maschere è possibile spezzare una rete in sottoreti:
 - Es.: la rete **193.206.144.0** (classe C) può essere ad esempio suddivisa in **quattro sottoreti**:

- 193.206.144.0 255.255.255.192 (indirizzi **da 0 a 63**)
- 193.206.144.64 255.255.255.192 (indirizzi **da 64 a 127**)
- 193.206.144.128 255.255.255.192 (indirizzi **da 128 a 191**)
- 193.206.144.192 255.255.255.192 (indirizzi **da 192 a 255**)

Perchè ??

Network mask e Subnetting

Net. Mask:

255.255.255.192

Utilizzo 2 bit degli hosts
per la rete.

11111111.
11000001.

11111111.
11001110.

11111111.
10010000.

11000000
00000000

IP: 193.206.144. xx

Prima Rete: 193.206.144.00000000 (0) Host: da 0 a 63

Seconda Rete: 193.206.144.01000000 (64) Host: da 64 a 127

Terza Rete: 193.206.144.10000000 (128) Host: da 128 a 191

Quarta Rete: 193.206.144.11000000 (192) Host: da 192 a 255

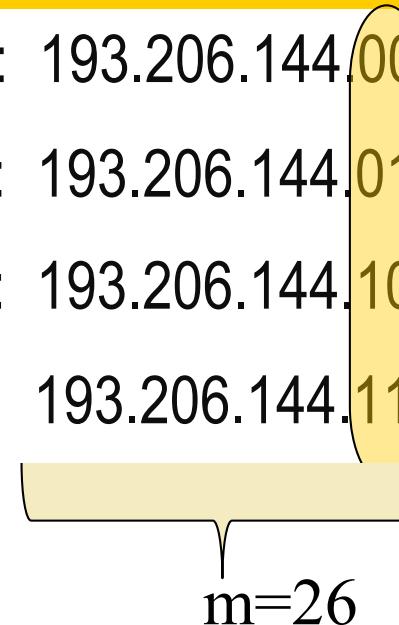
Network mask e Subnetting

Prima Rete: 193.206.144.**00**000000 (0) Host: da 0 a 63

Seconda Rete: 193.206.144.**01**000000 (64) Host: da 64 a 127

Terza Rete: 193.206.144.**10**000000 (128) Host: da 128 a 191

Quarta Rete: 193.206.144.**11**000000 (192) Host: da 192 a 255



NetAddress(n)

$$\text{NetAddress}(n) = \begin{cases} (n - 1)2^{(32-m)} & \text{se } \frac{2^{(32-m)}}{256} < 1 \\ (n - 1)\frac{2^{(32-m)}}{256} & \text{se } \frac{2^{(32-m)}}{256} \geq 1 \end{cases}$$

Esempio: NetAddress(Terza rete) = $(3-1)2^{(32-26)} = 128$

Network mask e Subnetting

Come scoprire a quale rete appartiene un indirizzo??

IP: 193.206.144.77 ---> 11000001.11001110.10010000.01001101 AND

Net. Mask: 255.255.255.192 ---> 11111111.11111111.11111111.11000000

NetAddress 11000001.11001110.10010000.01000000

193.206.144.64

Prima Rete: 193.206.144.00000000 (0) Host: da 0 a 63

Seconda Rete: 193.206.144.01000000 (64) Host: da 64 a 127

Terza Rete: 193.206.144.10000000 (128) Host: da 128 a 191

Quarta Rete: 193.206.144.11000000 (192) Host: da 192 a 255

Network mask e Subnetting

Altro Esempio:

IP: **128.75.87.34** ---> 10000000.01001011.01010111.00100010 AND

Net. Mask: 255.255.252.0 ---> 11111111.11111111.11111100.00000000

NetAddress 10000000.01001011.01010100.00000000

128.75.84.0

IP: **128.75.84.34** ---> 10000000.01001011.01010100.00100010 AND

Net. Mask: 255.255.252.0 ---> 11111111.11111111.11111100.00000000

NetAddress 10000000.01001011.01010100.00000000

128.75.84.0

Network mask (cont.)

- Una notazione molto diffusa per indicare la maschera è quella di indicare in coda all'indirizzo il **numero di bit**, a partire dal più significativo, che costituiscono l'indirizzo di rete

– la rete **131.154.20.0 255.255.255.0** si indica anche con la notazione

131.154.20.0 / 24

(i **primi 24 bit** costituiscono l'indirizzo di rete)

– la subnet **193.206.144.64 255.255.255.192** si indica con

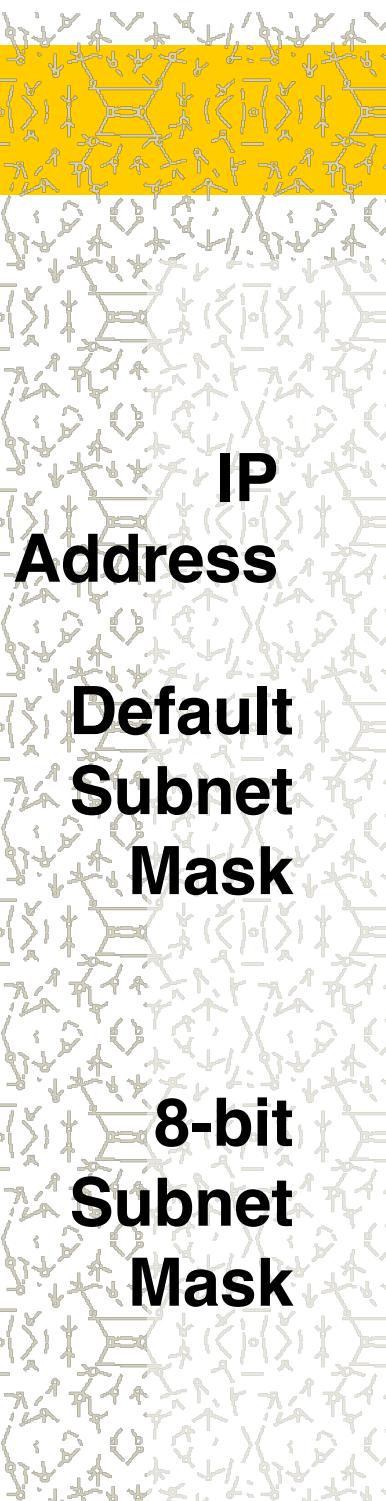
193.206.144.64 / 26

(26 bit per l'indirizzo di rete)

Network mask (cont.)

- Vale la pena di osservare che la **sottorete** di dimensioni **minime** deve avere un campo di **4 indirizzi**:
 - uno per indicare la **sottorete**,
 - uno per indicare il **broadcast**,
 - ed **almeno uno** per indirizzare un host;
- poichè al campo host vanno assegnati un certo numero di bit, un bit non è sufficiente, quindi ne servono **almeno due**, che forniscono due indirizzi per host
- questa tecnica è utilizzata per assegnare indirizzi di rete alle connessioni punto-punto tra i router, risparmiando il maggior numero di indirizzi possibile
- La **definizione delle sottoreti** non coinvolge la **authority internazionale** (o quella regionale) per gli indirizzi
 - le sottoreti fanno tutte parte dell'insieme degli indirizzi già assegnati

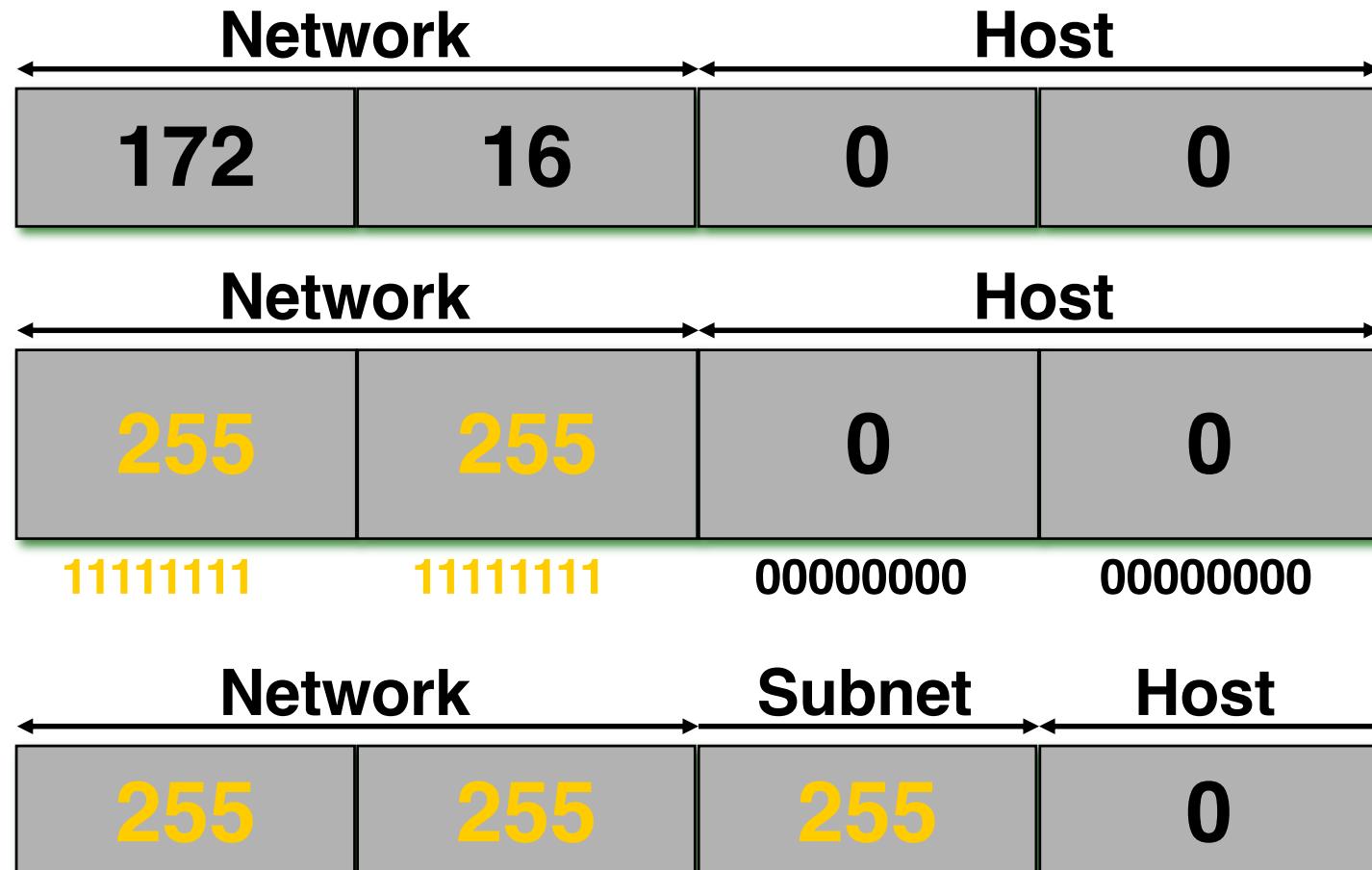
Network mask (cont.)



**IP
Address**

**Default
Subnet
Mask**

**8-bit
Subnet
Mask**

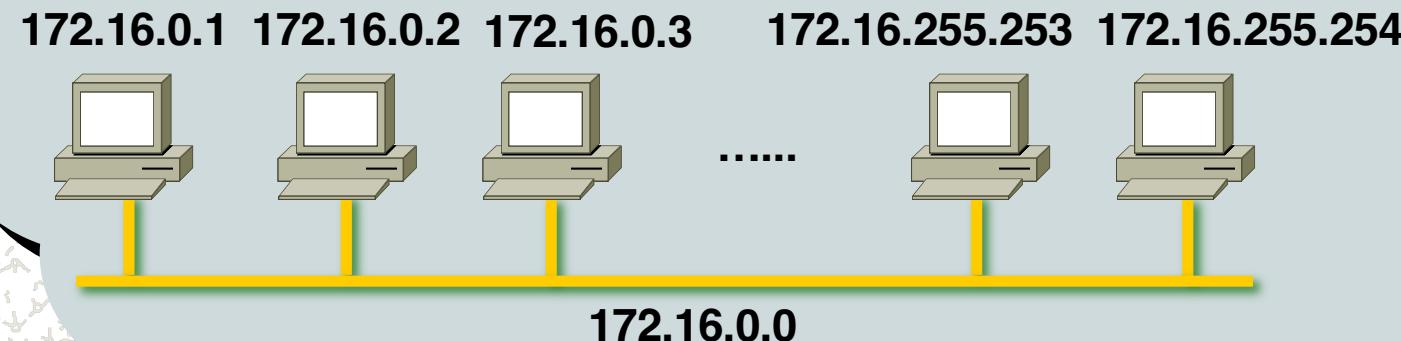


Rispettivamente indicabili come “/16” e “/24 dove il numero dopo lo slash rappresenta il numero di 1 nella subnet mask.

Equivalenti Decimali dei Bit Patterns

128	64	32	16	8	4	2	1	=	128
1	0	0	0	0	0	0	0	=	128
1	1	0	0	0	0	0	0	=	192
1	1	1	0	0	0	0	0	=	224
1	1	1	1	0	0	0	0	=	240
1	1	1	1	1	0	0	0	=	248
1	1	1	1	1	1	0	0	=	252
1	1	1	1	1	1	1	0	=	254
1	1	1	1	1	1	1	1	=	255

Network Mask senza Subnets



- Network 172.16.0.0

Network Mask senza Subnets

172.16.2.160

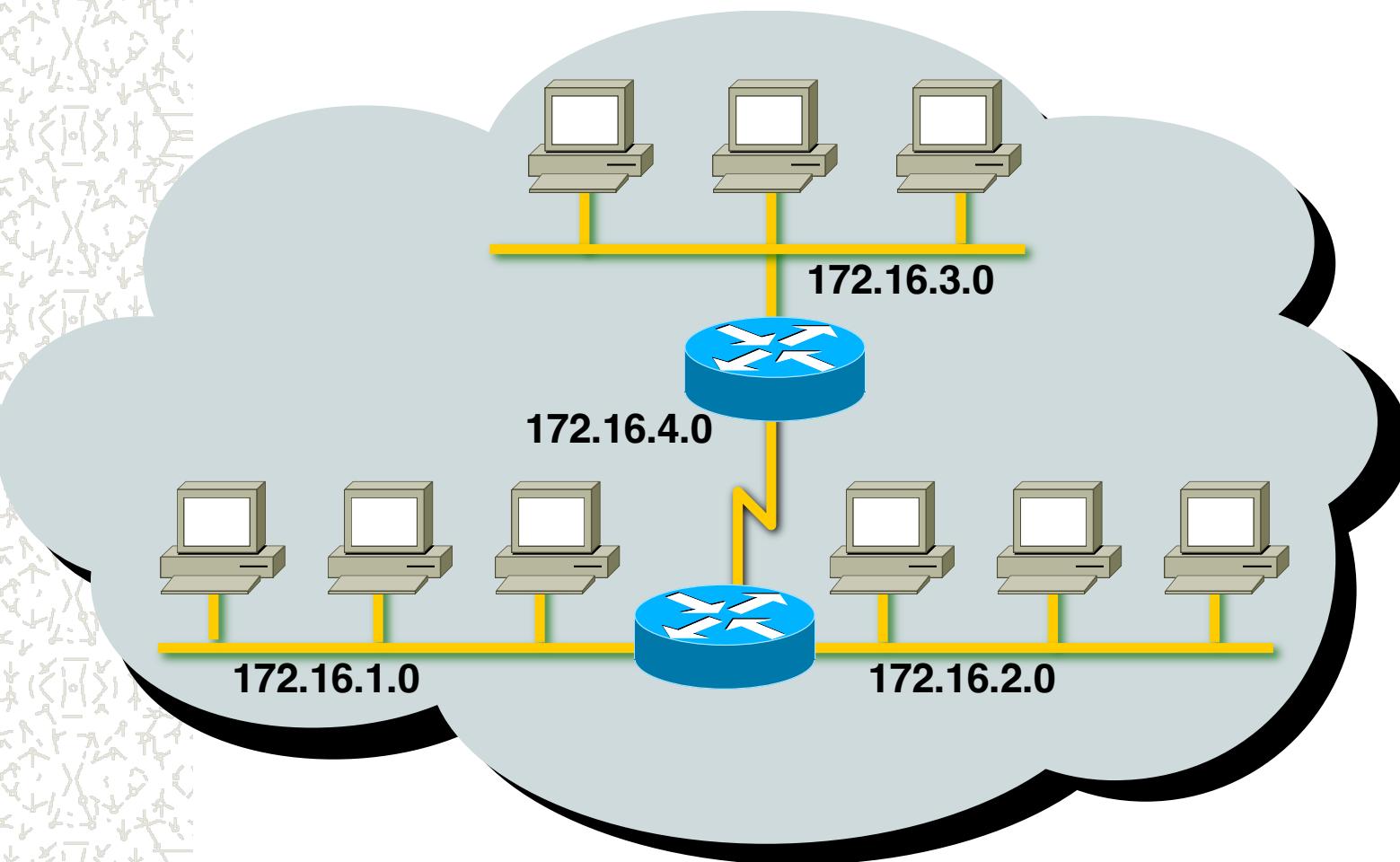
255.255.0.0

Network
Number

	Network		Host
	10101100	00010000	00000010
	11111111	11111111	00000000
	10101100	00010000	00000000
	172	16	0

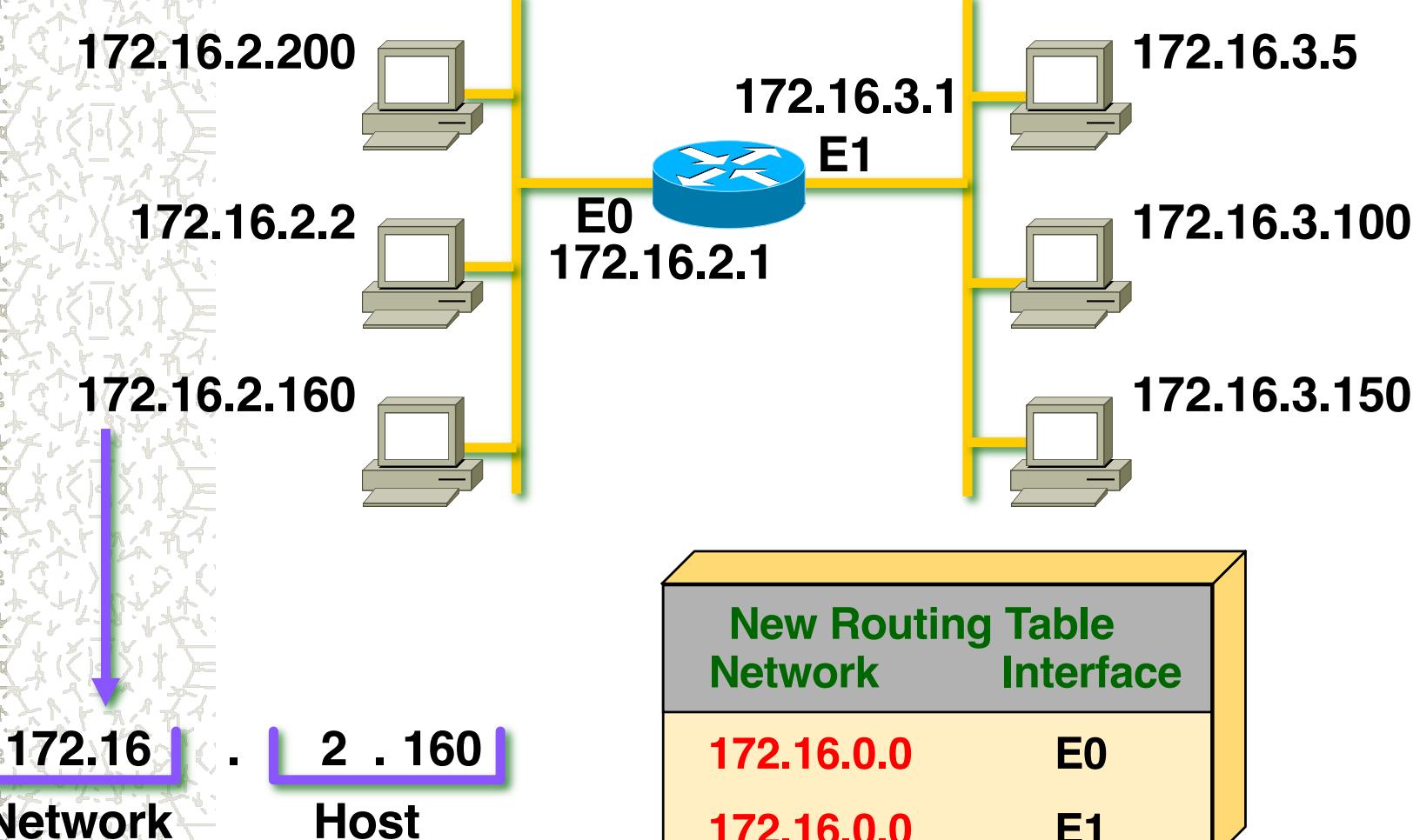
- Subnets non in uso — schema di default

Network Mask con Subnets

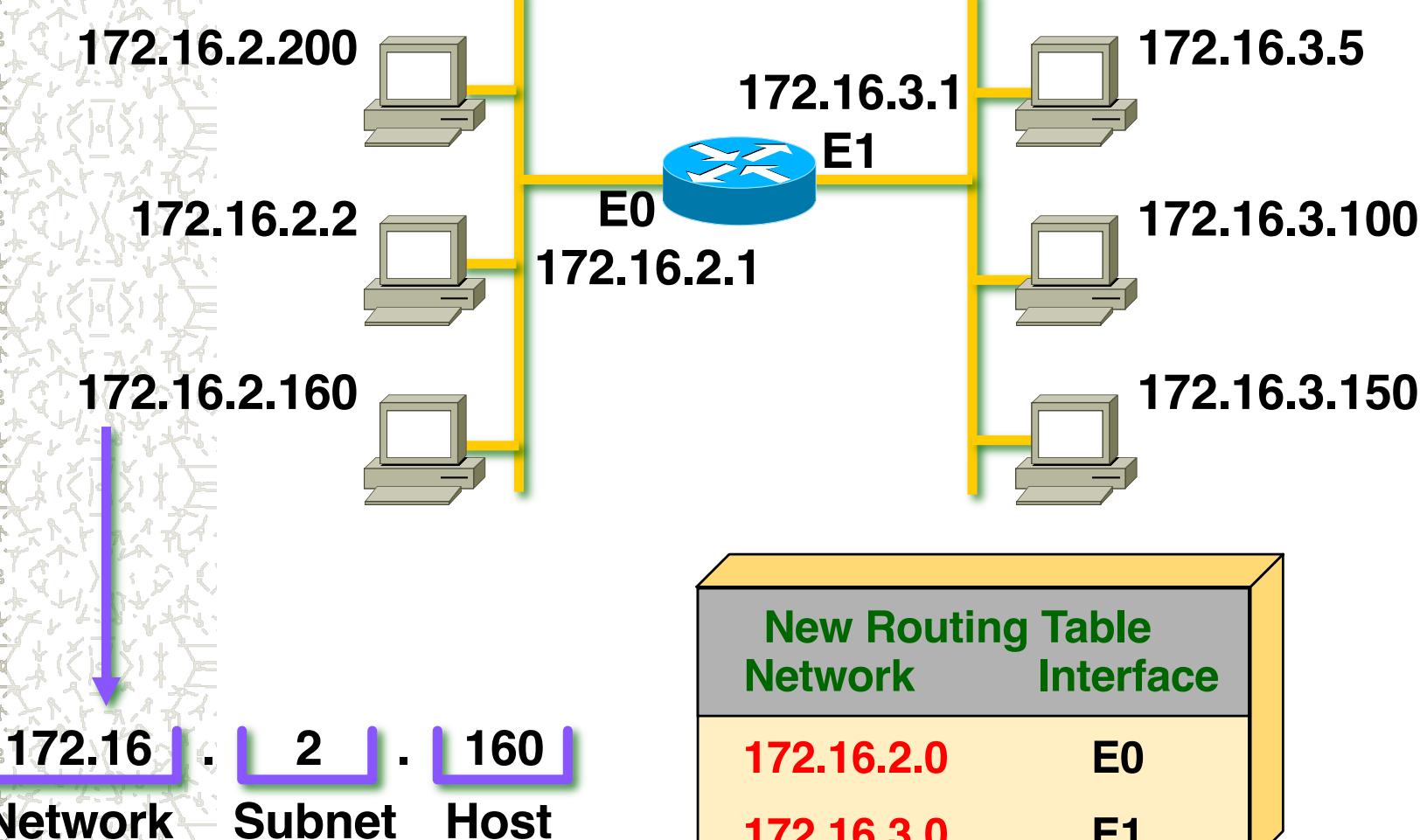


- Network 172.16.0.0

Network Mask con Subnets



Network Mask con Subnets



Network Mask con Subnets

172.16.2.160

255.255.255.0

Network
Number

	Network	Subnet	Host
172.16.2.160	10101100	00010000	00000010
255.255.255.0	11111111	11111111	11111111
	10101100	00010000	00000010

128 192 224 240 248 252 254 255

172	16	2	0
-----	----	---	---

- L'indirizzo di rete viene esteso di 8 bit a discapito degli hosts

Network Mask con Subnets

172.16.2.160
255.255.255.192

Network
Number

Network	Subnet	Host	
10101100	00010000	00000010	10100000
11111111	11111111	11111111	11000000
10101100	00010000	00000010	10000000

128 192 224 240 248 252 254 255
128 192 224 240 248 252 254 255

172 16 2 128

- L'indirizzo di rete viene esteso di 10 bit a discapito degli hosts

Esempio

Assumiamo di voler partizionare il netblock 172.16.32.0/20 per ottenere 5 classi da 62 hosts.

- Per indirizzare 64 hosts abbiamo bisogno di 6 bit.
- Quindi restano $32-6= 26$ bit per la rete. La rete assumerà la forma

x.x.x.x/26

- Il netblock originale impiega I primi 20 bit per la rete, quindi possiamo usare i successive 6 bit per le sottoreti.

Rete di partenza : 172.16.32.0/20

In binario

10101100. 00010000.00100000.00000000

Rete di partenza

Nuova rete

Hosts

63

Esempio

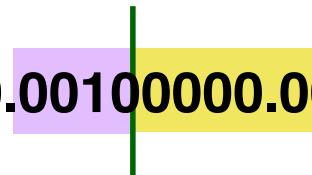
- Assumiamo di voler partizionare il netblock 172.16.32.0/20 per ottenere 5 classi da 62 hosts (/26)

Netblock da partizionare: 172.16.32.0/20

In binario 10101100. 00010000.00100000.00000000

Prima subnet /26: 172.16.32.0/26

In binario 10101100. 00010000.00100000.00000000



1st subnet:

10101100 . 00010000 .0010 0000.00 000000=172.16.32.0/26

2nd subnet:

172 . 16 .0010 0000.01 000000=172.16.32.64/26

3rd subnet:

172 . 16 .0010 0000.10 000000=172.16.32.128/26

4th subnet:

172 . 16 .0010 0000.11 000000=172.16.32.192/26

5th subnet:

172 . 16 .0010 0001.00 000000=172.16.33.0/26

Network

Subnet

**VLSM
Subnet**

Host

64

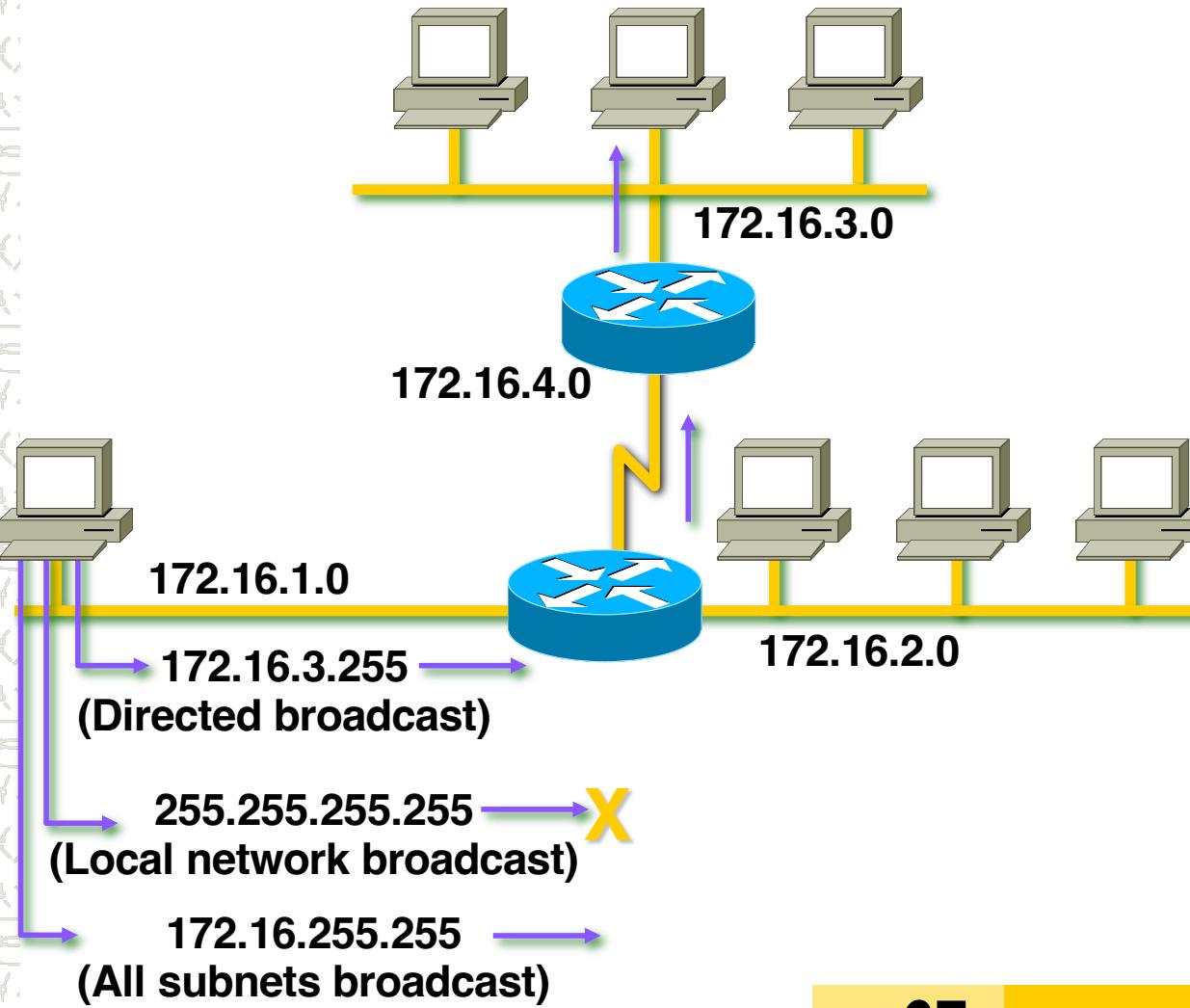
Esercizio

Address	Subnet Mask	Class	Subnet
172.16.2.10	255.255.255.0		
10.6.24.20	255.255.240.0		
10.30.36.12	255.255.255.0		

Soluzione

Address	Subnet Mask	Class	Subnet
172.16.2.10	255.255.255.0	B	172.16.2.0
10.6.24.20	255.255.240.0	A	10.6.16.0
10.30.36.12	255.255.255.0	A	10.30.36.0

Indirizzi Broadcast per Subnets



Esempio Indirizzi Broadcast per Subnets

IP Host Address: 172.16.2.121

Subnet Mask: 255.255.255.0

Network	Network	Subnet	Host
172.16.2.121: 10101100	00010000	00000010	01111001
255.255.255.0: 11111111	11111111	11111111	00000000
Subnet: 10101100	00010000	00000010	00000000
Broadcast: 10101100	00010000	00000010	11111111

- Subnet Address = 172.16.2.0
- Host Addresses = 172.16.2.1–172.16.2.254
- Broadcast Address = 172.16.2.255
- Eight bits of subnetting

Esercizio

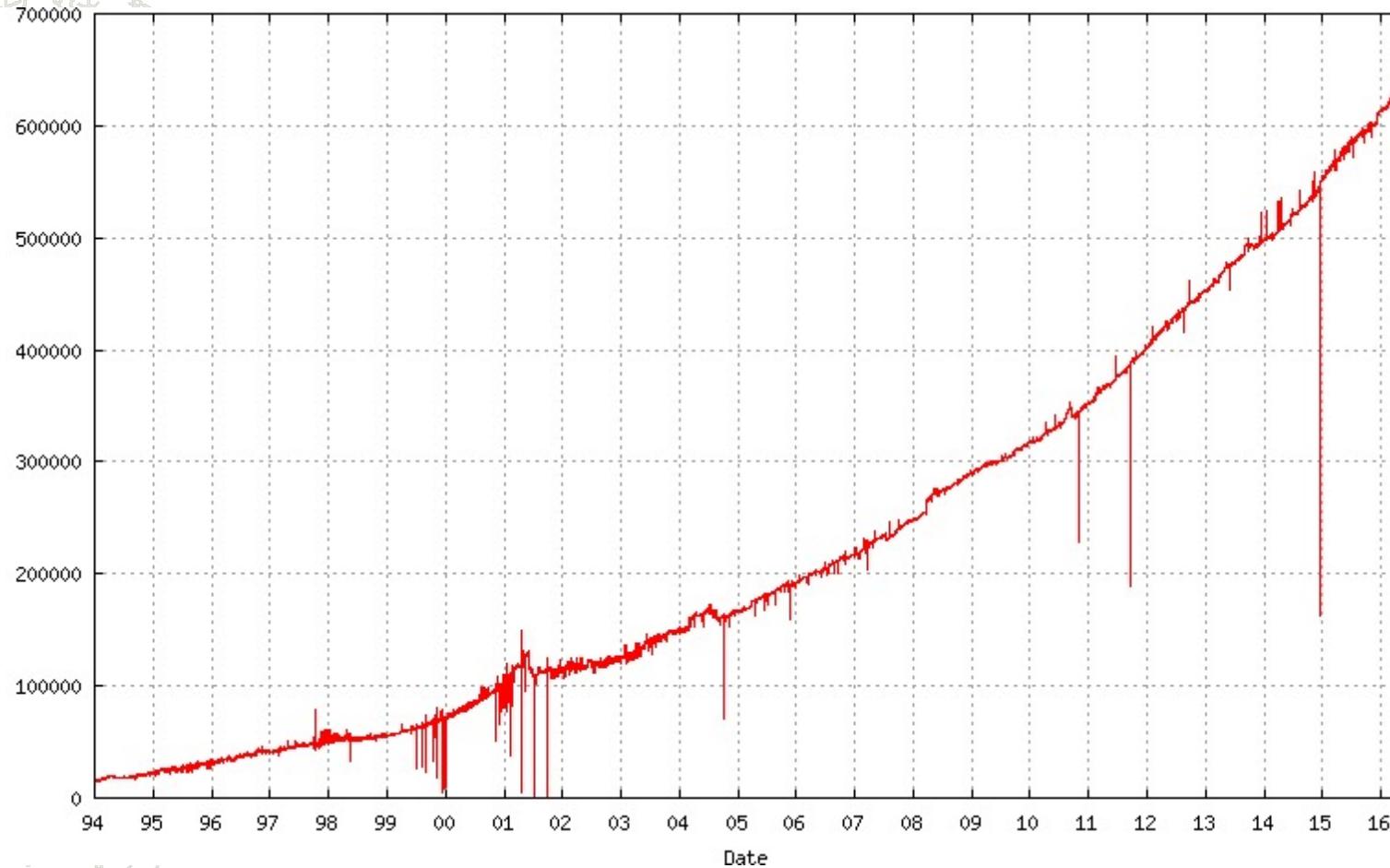
Address	Subnet Mask	Class	Subnet	Broadcast
201.222.10.60	255.255.255.248			
15.16.193.6	255.255.248.0			
128.16.32.13	255.255.255.252			
153.50.6.27	255.255.255.128			

Soluzione

Address	Subnet Mask	Class	Subnet	Broadcast
201.222.10.60	255.255.255.248	C	201.222.10.56	201.222.10.63
15.16.193.6	255.255.248.0	A	15.16.192.0	15.16.199.255
128.16.32.13	255.255.255.252	B	128.16.32.12	128.16.32.15
153.50.6.27	255.255.255.128	B	153.50.6.0	153.50.6.127

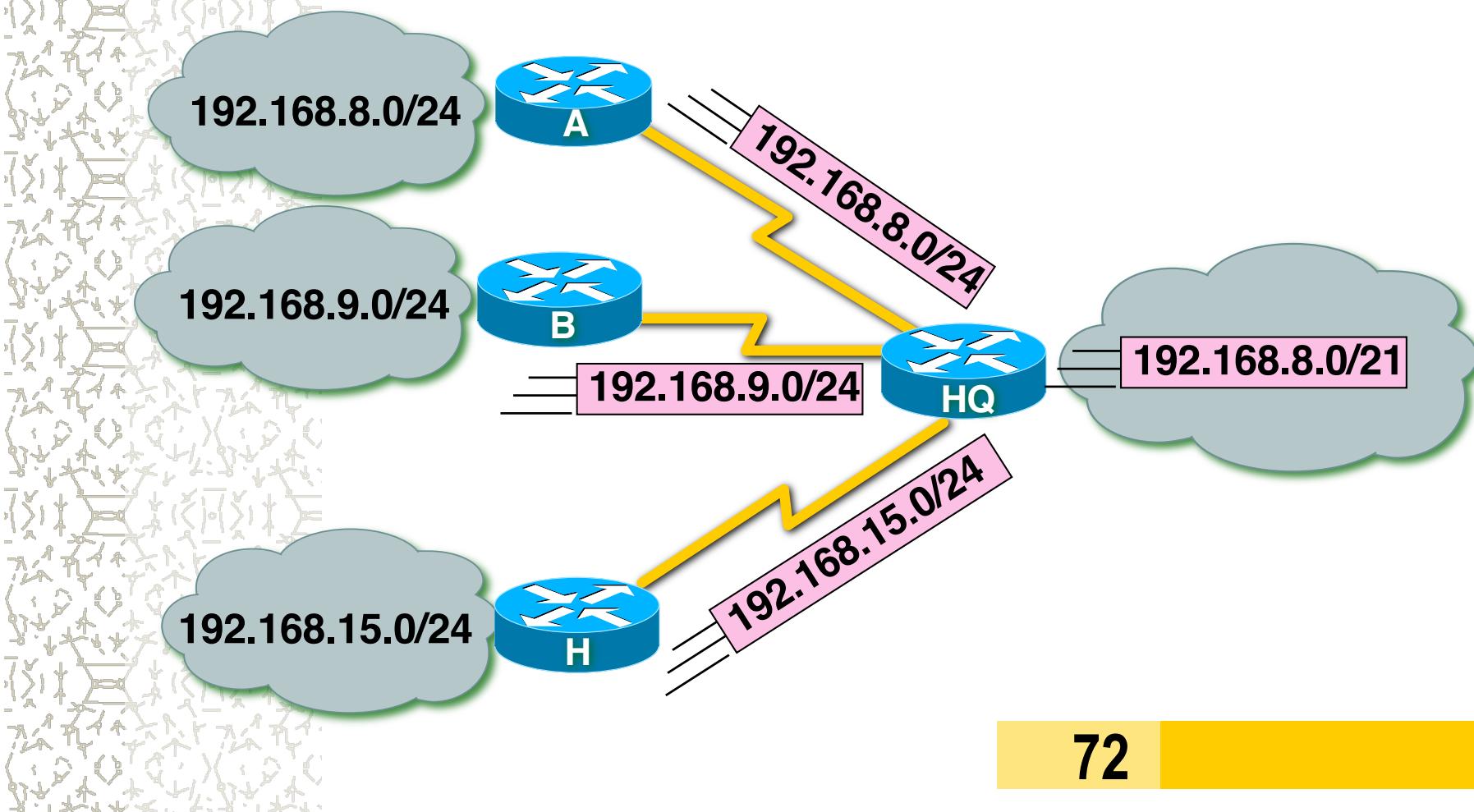
Problema legato al Classless InterDomain Routing

- Questa soluzione comporta però un **problema potenziale** grave:
 - l'aumento considerevole delle reti indirizzabili può far **esplodere** la dimensione delle **tabelle di routing**, che virtualmente potrebbero dover contenere **milioni di record**



Soluzione: Aggregazione di subnets

- Per ovviare a ciò gli indirizzi vengono assegnati per quanto possibile a **blocchi** alle varie organizzazioni regionali e locali che devono **annunciare verso l'esterno** della loro area solo **una rete**, che costituisce **l'aggregato** delle sottoreti al suo interno.



Aggregazione di reti

- L'indirizzamento a classi ha anche portato al problema opposto:
 - una classe C prevede un **massimo** di 254 indirizzi (lo 0 ed il 255 non sono utilizzabili)
 - spesso aziende o università hanno aumentato il numero di host connessi in rete fino ad **eccedere** questo limite
- Utilizzando la tecnica della maschera è possibile **accorpare** classi C con **indirizzi contigui** opportuni
 - ad esempio, la sezione INFN di Genova ha avuto assegnate **4 classi C**, dalla 193.206.144.0 alla 193.206.147.0
 - il valore binario di queste reti è
11000001 11001110 100100**00** 00000000
11000001 11001110 100100**01** 00000000
11000001 11001110 100100**10** 00000000
11000001 11001110 100100**11** 00000000
 - utilizzando una maschera a **22 bit** è possibile accorpare queste quattro reti in **una unica rete IP** indicata come 193.206.144.0/22 (o con maschera 255.255.252.0)

Classless InterDomain Routing

172.16.0.0-172.31.0.0
255.255.240.0

Network
Number

Network	Subnet	Host	
10101100	00010000	00000000	00000000
11111111	11110000	00000000	00000000
10101100	00010000	00000000	00000000

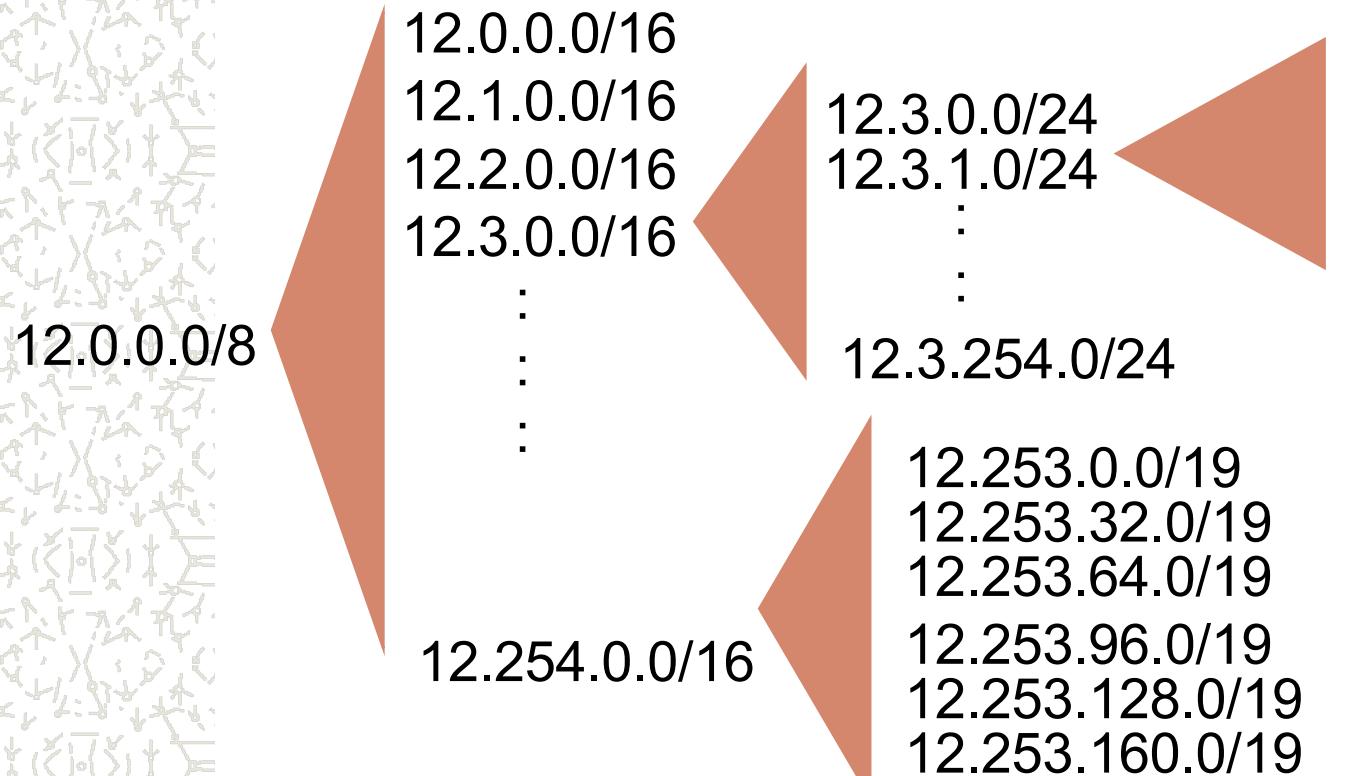
128 192 224 240 252 254 255 128 192 224 240 248 252 254 255 128 192 224 240 248 252 254 255

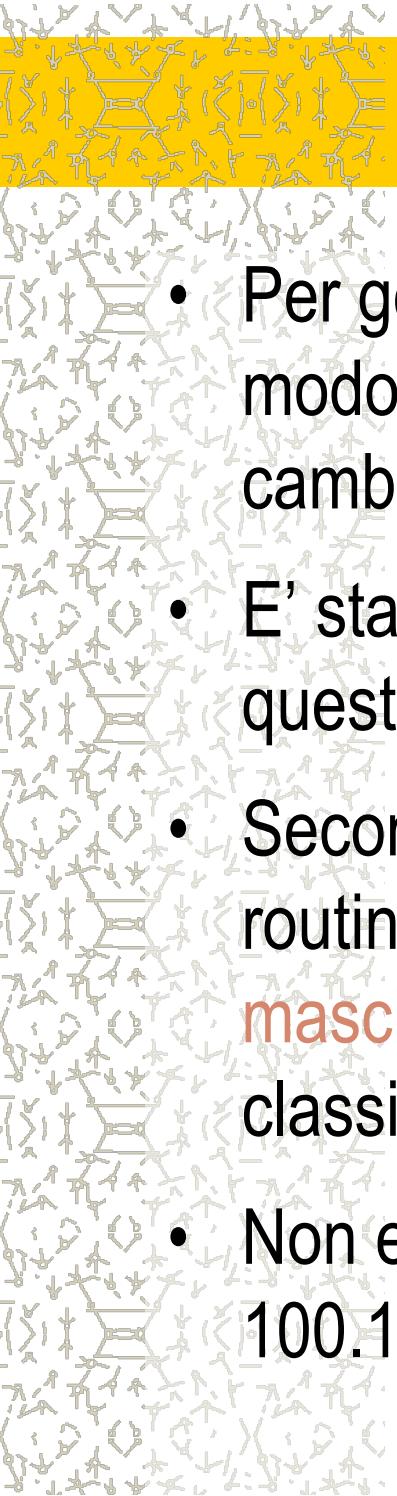
172 16 0 0 /12

- Aggregate 16 sottoreti riducendo il network address di 4 bits

CIDR: Hierarchical Address Allocation

- Il CIDR è il meccanismo di base per la scalabilità di Internet
 - Indirizzi vengono allocati e distribuiti in blocchi contigui (prefix blocks)
 - I protocolli e i meccanismi di routing lavorano su questi prefissi
 - **Aggregando si riesce a contenere la dimensione delle routing tables**





Classless InterDomain Routing

- Per gestire questo **nuovo schema** di indirizzamento il modo in cui il **router** gestisce le **tabelle** di routing deve cambiare
- E' stato introdotto un **nuovo standard** che specifica queste modifiche (**RFC 1519**), col nome di **CIDR**
- Secondo questo standard ogni **record** della tabella di routing specifica l'indirizzo della destinazione **con la sua maschera**, in modo da superare la definizione delle classi
- Non esiste più una vera distinzione tra una rete 100.1.2.0/24 ed una rete 200.201.20.0/24

Instradamento e CIDR (longest prefix match)

- **Un indirizzo di destinazione può corrispondere a più di una voce della tabella di inoltro di un router.**
- I pacchetti IP **non sanno nulla** delle maschere: come instradare?
- Supponiamo di dover instradare un pacchetto indirizzato a **130.251.61.129**, e di avere nelle tabelle di routing:
 - 130.0.0.0/8 verso l'interfaccia 1
 - 130.251.0.0/16 verso l'interfaccia 2
 - 130.251.61.0/24 verso l'interfaccia 3
 - 130.251.61.64/26 verso l'interfaccia 4
- **La scelta viene sempre fatta verso la rete (adatta) che ha la maschera più lunga**
 - nell'esempio si ha:
 - 1000010 1111011 0011101 10000001 (indirizzo di destinazione)
 - 1000010 (130.0.0.0/8)
 - 1000010 1111011 (130.251.0.0/16)
 - 1000010 1111011 0011101 (130.251.61.0/24)
 - 1000010 1111011 0011101 01 (130.251.61.64/26)
 - In questo caso l'indirizzo **non fa parte** della rete relativa alla **quarta riga**, ma può far parte delle reti relative alle altre righe; tra queste si sceglierà **l'interfaccia 3** perché è quella verso la rete adatta (**matching prefix**) con la maschera più lunga

Il futuro: IPv6 (versione 6)

- All'inizio degli anni '90 l'IETF iniziò la ricerca di un **successore** di IPv4
 - Motivazione primaria: la necessità di **ampliare lo spazio di indirizzi** che potesse soddisfare tutte le nuove esigenze di interconnessione di nuovi elaboratori
- Basandosi sul trend dell'epoca si affacciavano previsioni di **esaurimento** degli indirizzi tra il 2008 e il 2018
 - Nel **1996** l'American Registry for Internet Numbers dichiarava **esaurita la classe A**, risultava assegnato il 62% dello spazio indirizzi di classe B e il 37% di quelli della classe C
 - Sebbene questo consentisse comunque ancora del tempo prima dell'esaurimento totale, lo sforzo necessario e il tempo previsto per trovare un'alternativa stimolò l'avvio dello studio e progettazione di un **nuovo protocollo**
- Fu sviluppato un protocollo che i progettisti modellarono sull'IPv4 esistente, ampliando e migliorando alcune sue caratteristiche, chiamato **IPv6** (o IPng: next generation)



Obiettivi di IPv6

- Indirizzamento illimitato
- Semplicità del protocollo per ridurre i tempi di elaborazione nei router
- Sicurezza
- Supporto per pacchetti di grosse dimensioni
- Gestione del tipo di servizio
- Prevedere evoluzioni future del protocollo
- Supportare i protocolli di livello superiore che si appoggiano ad IPv4

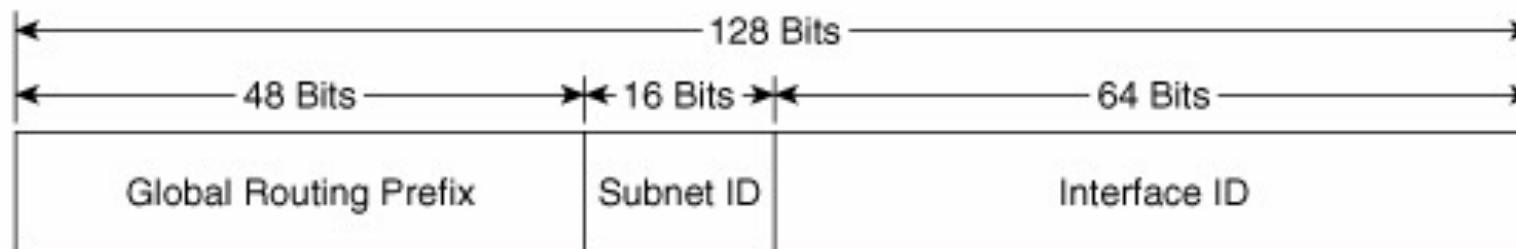
Indirizzi IPv6

- IPv6 prevede l'utilizzo di indirizzi a **16 byte** (128 bit)
- La notazione utilizzata è una sequenza di otto gruppi di quattro cifre esadecimali, separate da ":"

8000:0000:0000:0000:0123:4567:89AB:CDEF

- Per rappresentare più comodamente gli indirizzi si possono utilizzare **ottimizzazioni**:
 - si possono **omettere gli zeri** ad inizio di un gruppo (:0123: diventa :123:)
 - si possono **omettere gruppi di zeri consecutivi**, rappresentati da una sequenza "::"

8000::123:4567:89AB:CDEF

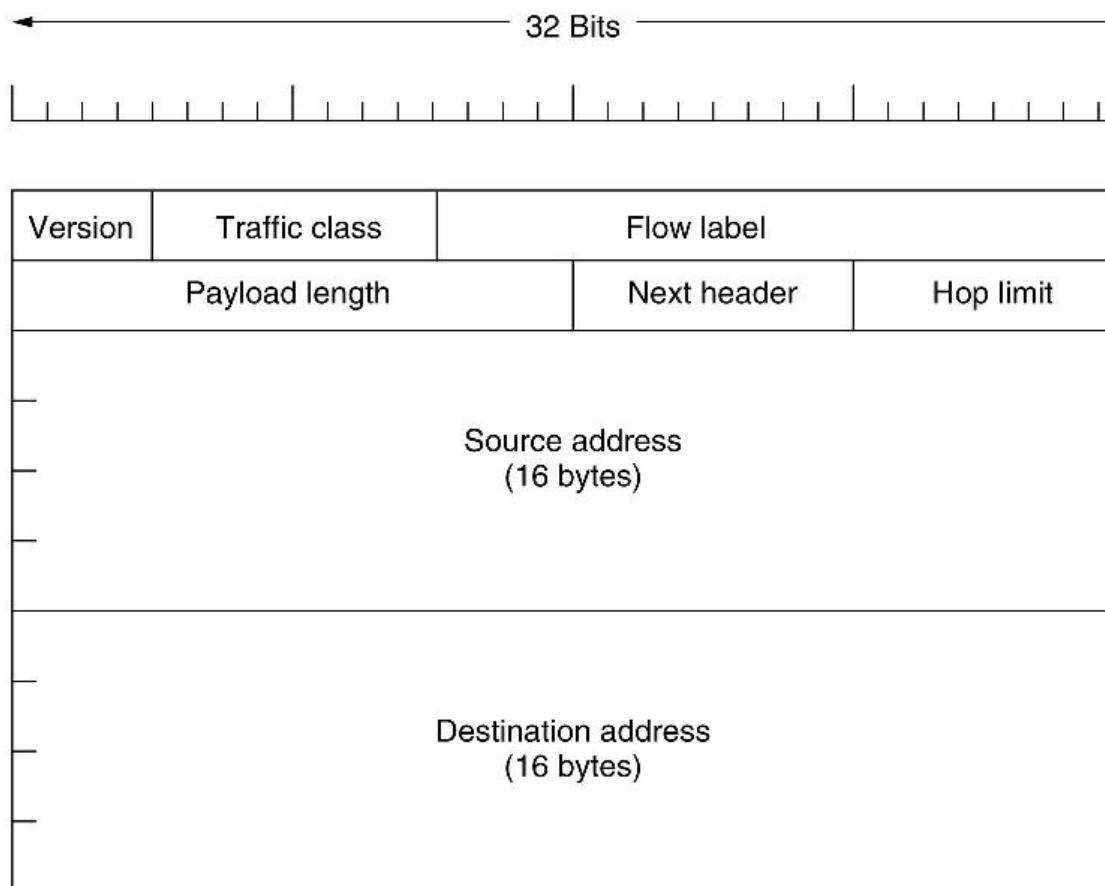


Indirizzi IPv6 (cont.)

- Come per IPv4 l'indirizzo contiene una informazione di **rete** ed una informazione di **host**
- La notazione per definire **quale parte** dell'indirizzo è dedicato alla rete è quella di specificare la **lunghezza in bit** dell'indirizzo di rete dopo un “**/**” in coda all'indirizzo (**come in IPv4**)
- Anche in IPv6 la “**rete**” è identificata dall'indirizzo con **tutti “0”** nel campo di indirizzo dell'host

Pacchetto IPv6

- Il pacchetto IPv6 è costituito da un **header** di lunghezza **fissa (40 byte)** ed un campo dati; il campo dati puo' **opzionalmente** contenere altri **header** prima dei dati veri e propri



Header di IPv6

- Il campo **version** non cambia significato, ed assume il valore 6
- Il campo **traffic class** serve ad identificare i pacchetti che necessitano di un instradamento **particolare**
 - essenzialmente introdotto per supportare il **traffico prioritario**, o il traffico di tipo **voce** o **video stream** che richiede **ritardi costanti**
 - un campo simile esiste nell'header di IPv4, inutilizzato
- Il campo **flow label** è stato introdotto per identificare in qualche modo i pacchetti appartenenti allo **stesso flusso trasmissivo**
 - sempre al fine di supportare meglio il traffico **voce/video**
 - è un tentativo di identificare **il flusso** di dati di una “**connessione**” in un protocollo **connection less**
 - attualmente in fase **sperimentale**

Header di IPv6 (cont.)

- Il campo **payload length** indica la lunghezza del pacchetto in **byte**, esclusi i 40 byte fissi dell'header
 - la lunghezza massima del pacchetto è di **65536 bytes** (+ 40)
- Il campo **next header** indica
 - il **protocollo del livello di trasporto** a cui sono destinati i dati (TCP o UDP) se non c'è intestazione estesa
 - il **tipo di intestazione estesa** successiva utilizzata (se c'è): in questo caso sarà il corrispondente campo dell'header dell'**ultima** intestazione estesa a specificare il protocollo di trasporto di destinazione
- Il campo **hop limit** è equivalente al campo **time to live** dell'header IPv4, ma ora ha l'esclusivo significato di **conto degli hop**
 - viene decrementato ad ogni hop; raggiunto lo zero il pacchetto viene scartato
 - ha la **stessa funzione** del corrispondente campo IPv4: evitare che un pacchetto rimanga **tropppo a lungo** in rete in caso di problemi di routing
- Gli ultimi campi sono gli **indirizzi sorgente e destinatario** del pacchetto

Caratteristiche di IPv6

Il formato dell'intestazione dei datagrammi è stato notevolmente semplificato. Molti campi sono stati eliminati o modificati.

IPv6 prevede più di un intestazione. Con questa variazione è possibile creare intestazioni per ogni tipo di servizio ipotizzabile.

Cosa non c'è più

- Il campo **IHL** (Internet Header Length) che rappresenta la lunghezza dell'header
 - non è più necessario, perché la lunghezza dell'header è **fissata** a 40 bytes
- Il campo **protocol** è sostituito dal campo **next header**
- I campi riguardanti la **frammentazione**
 - IPv6 **non prevede** che i router eseguano frammentazione, perchè fa perdere tempo
 - i nodi IPv6 **tentano** di identificare la dimensione corretta dei pacchetti da scambiarsi in modo **dinamico**
 - non basta: se il router **non può inoltrare** un pacchetto, invia un messaggio (**ICMP**) indietro per notificare il fatto e butta il pacchetto
 - di fatto risulta **più efficiente** fare in modo che l'host di partenza invii i pacchetti di dimensione corretta che non frammentare nei router

Cosa non c'è più (cont.)

- Il campo **checksum**
 - IPv6 **non utilizza checksum** sui suoi pacchetti, per motivi di efficienza
 - pur nel caso ridotto di un checksum dedicato all'header, il controllo della correttezza risulta un processo **molto costoso** per i router
 - essendoci meccanismi **analoghi** a livello di **data link** ed a livello di **trasporto**, IPv6 ne fa a meno

Intestazione estesa

- I progettisti di IPv6 hanno previsto la possibilità di utilizzare **header aggiuntivi (extension header)**
 - il campo **next header** dell'header fisso identifica con un **codice** opportuno il **tipo** di extension header che segue
 - Ciascun extension header inizia con 2 byte:
 - **next header**: il tipo di extension header che segue il corrente (o il **protocollo di destinazione** se il corrente è **l'ultimo header**)
 - **extension header length**: la **lunghezza in byte** dell'extension header corrente (i diversi tipi hanno lunghezze differenti, ed alcuni tipi hanno lunghezza **variabile**)
- seguita dai dati specifici dell'hestension header

Tipi di estension header

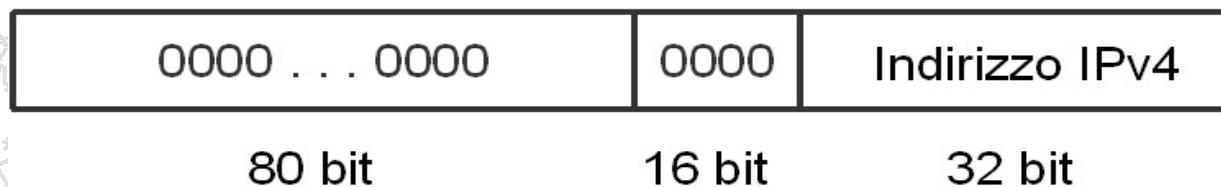
- Sono stati definiti **6 tipi** di estension header
 - opzioni **hop-by-hop**: sono opzioni che **tutti i router** devono esaminare; al momento è definito un solo tipo di opzione (**jumbo datagram**) che serve per poter inviare pacchetti di dimensione **superiore a 64 KB**
 - opzioni di **destinazione**: introdotte per essere interpretate dall'**host di destinazione**, attualmente non ancora definite (fornisce flessibilità al protocollo per utilizzi futuri)
 - opzioni di **routing**: per implementare **source-routing**, cioè instradamento definito dalla sorgente (simile al **loose source routing** di IPv4)
 - opzioni di **frammentazione**: da utilizzare se “**la sorgente**” deve frammentare (i router non frammentano); gestita come in IPv4
 - **autenticazione**: finalizzato a fornire un meccanismo per accettare l’identità del mittente del pacchetto
 - **crittazione**: finalizzato a proteggere i dati tramite codifiche di cifratura

Transizione da IPv4 a IPv6

- **Problema** che deve essere affrontato perchè mentre l'IPv6 può essere costruito **compatibile** con IPv4 nel senso che può spedire, instradare e ricevere datagram IPv4, **IPv4 non è in grado di gestire datagram IPv6**
- **Possibile opzione**
 - Definire **giorno e ora** in cui tutte le macchine si aggiornano da IPv4 a IPv6
 - Questa soluzione è stata **già provata** in passato per **altre transizioni** quando la rete era ancora agli esordi e provocò **non pochi** problemi
 - Diventa ancor più impensabile oggi che i nodi e i router convolti sono decine (centinaia?) di milioni

Indirizzi IPv6 compatibili IPv4

- Un indirizzo compatibile IPv4 consente a un host che supporta IPv6 di parlare IPv6 anche se il router o i router locali non parlano IPv6
- Gli indirizzi compatibili IPv4 avvisano il software del mittente di creare un tunnel, incapsulando il pacchetto IPv6 in un pacchetto IPv4
- Gli indirizzi IPv4 sono rappresentati con **6 gruppi di zeri**, e due gruppi che rappresentano l'indirizzo IPv4 (rappresentabili anche in notazione decimale):
 - ::89AB:CDEF oppure ::137.171.205.239



Gli indirizzi **IPv4 compatibili** sono stati deprecati in favore degli indirizzi **IPv4-Mapped address**

Indirizzo IPv6 mappato IPv4

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

0000 . . . 0000	FFFF	Indirizzo IPv4
-----------------	------	----------------

80 bit

16 bit

32 bit

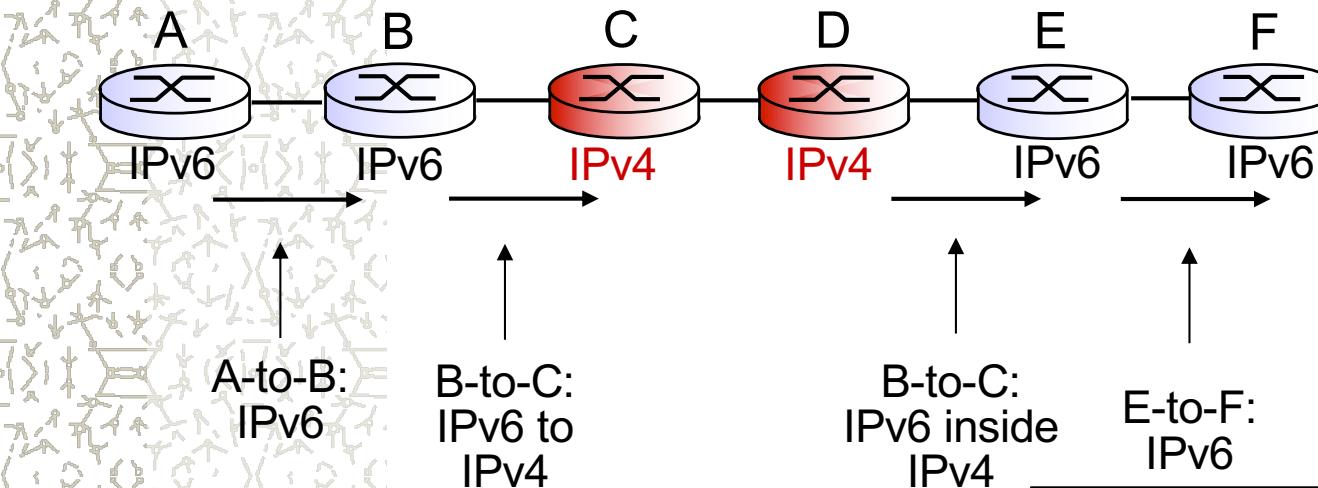
Opzione: nodi dual stack

- Questa opzione prevede l'introduzione di **nodi IPv6 compatibili**, in cui i nodi IPv6 dispongono pure di una completa implementazione IPv4
- I nodi IPv6/IPv4 (RFC 1933) hanno **entrambi gli indirizzi** e devono essere in grado di **determinare** se il nodo con cui devono parlare è un nodo IPv6 compatibile o solo IPv4
- Questo può comunque portare come risultato che 2 nodi IPv6 compatibili si **scambino** comunque tra loro **datagram IPv4**

Perché??

Opzione: nodi dual stack

- Questo può comunque portare come risultato che 2 nodi IPv6 compatibili si **scambino** comunque tra loro **datagram IPv4**

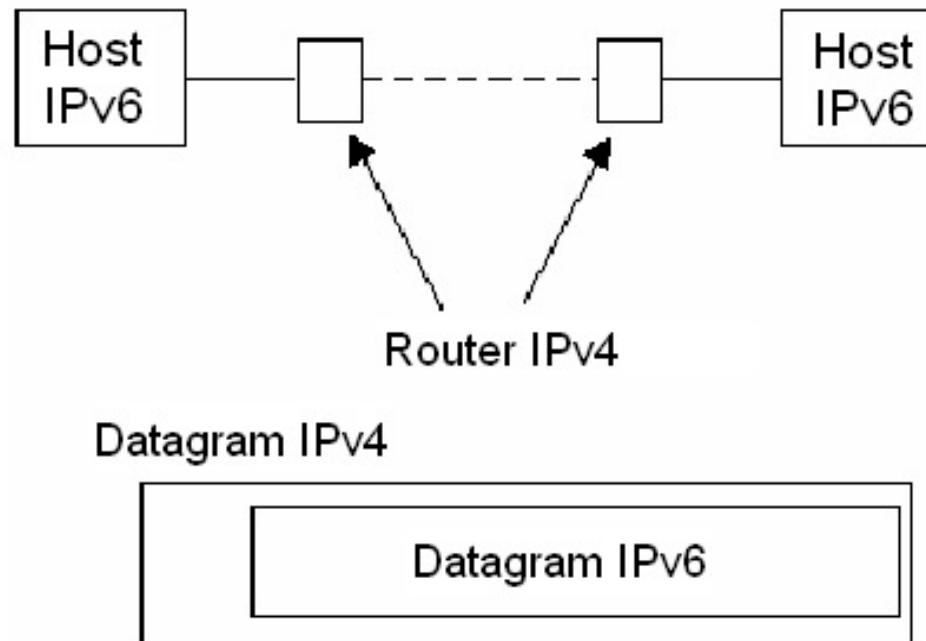


Il campo dati di IPv6 di B è copiato nel campo dati di IPv4 di C.
Alcuni campi IPv6 non hanno i corrispondenti in IPv6 e vengono persi.

Quindi, anche se A ed F possono scambiarsi datagram IPv6, il datagram in arrivo al nodo E da D non contiene tutti i campi IPv6 ma solo quelli di IPv4.
Di fatto, la comunicazione avviene in IPv4

Tunneling IPv6 – IPv4

Il datagram IPv6 è inserito nel campo dati dell'IPv4.

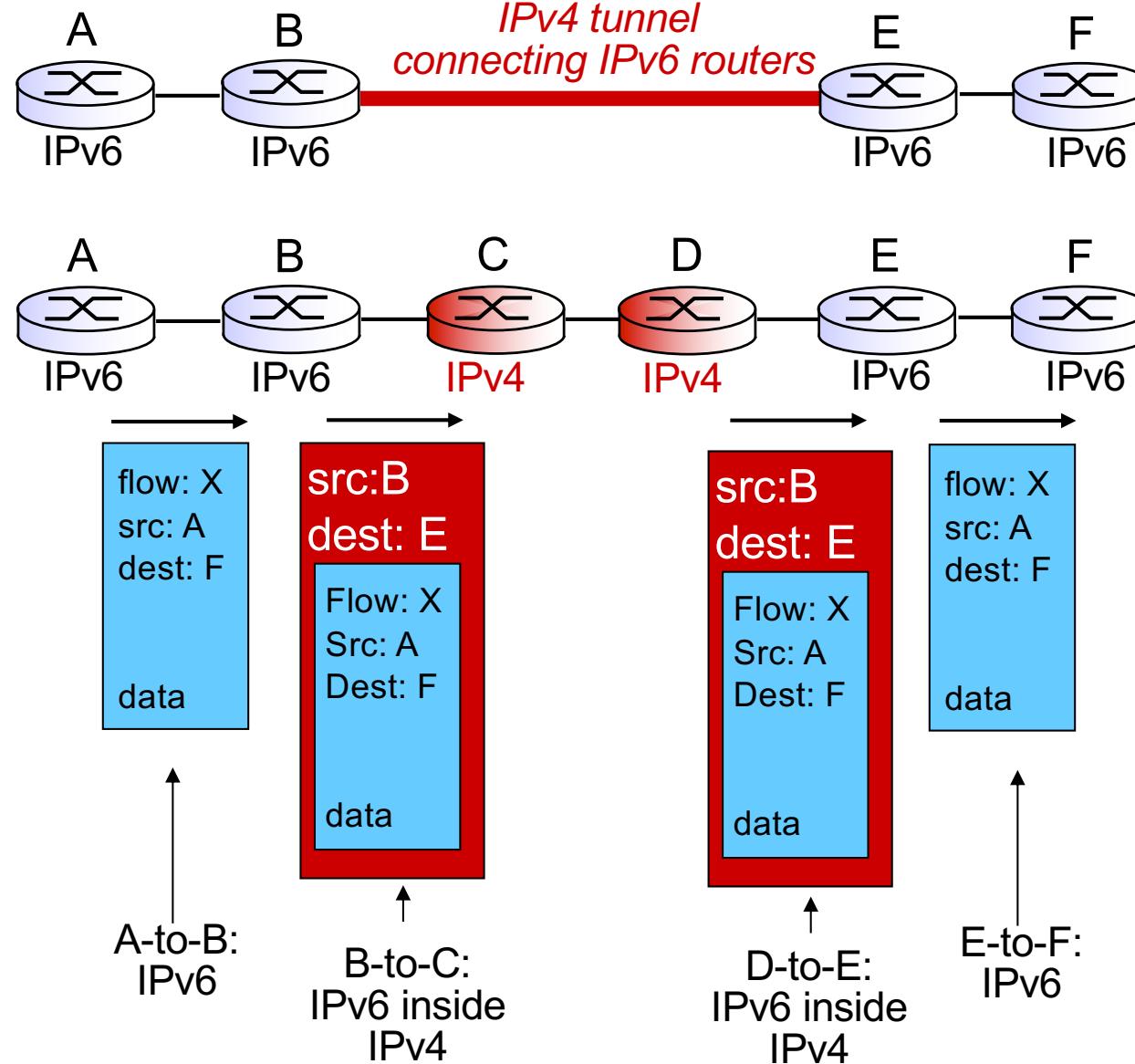




logical view:

physical view:

Tunneling



Funzionamento con DNS

- Un'applicazione IPv6 chiede al DNS (Domain Name System) l'indirizzo di un host, ma l'host ha solo un indirizzo IPv4
- Il DNS crea automaticamente l'indirizzo IPv6 mappato IPv4
 - Il kernel capisce che si tratta di un indirizzo speciale e usa la comunicazione IPv4

IP addresses: how to get one?

Q: In che modo un host ottiene un indirizzo in una rete?

- hard-coded dal system administrator in un file
 - Windows: control-panel->network->configuration->tcp/ip->properties
 - UNIX: /etc/rc.config
- **DHCP: Dynamic Host Configuration Protocol:** Ottiene un indirizzo dinamicamente da un server.
 - “plug-and-play”

DHCP: Dynamic Host Configuration Protocol

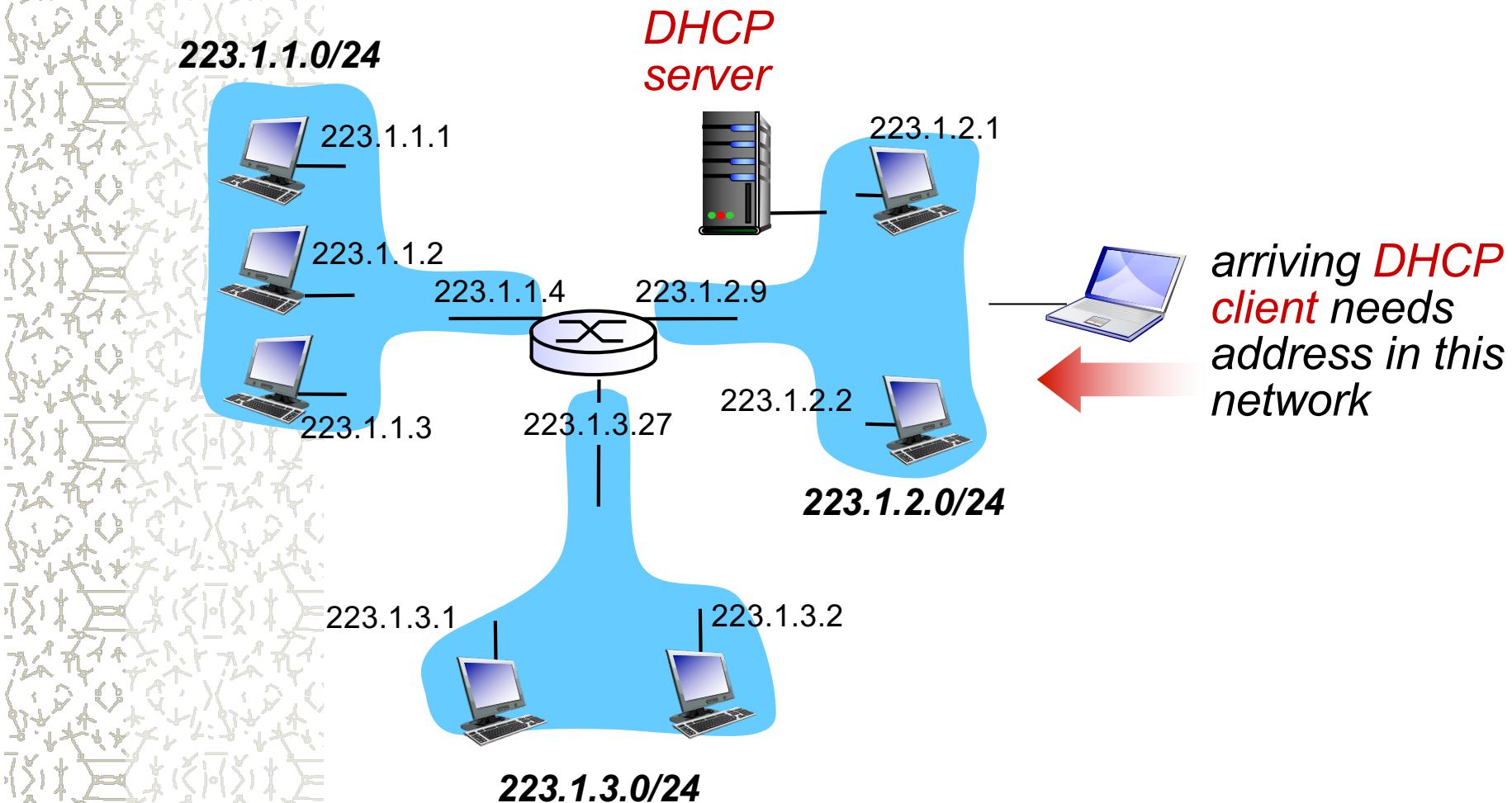
goal: consentire all'host di ottenere dinamicamente il proprio indirizzo IP dal server di rete quando si unisce alla rete. Generalmente avviene con un contratto di locazione dell'IP.

- può rinnovare il contratto di locazione dell'indirizzo in uso
- consente il riutilizzo degli indirizzi (conserva l'indirizzo solo quando è collegato / "acceso")
- supporto per gli utenti mobili che vogliono unirsi alla rete

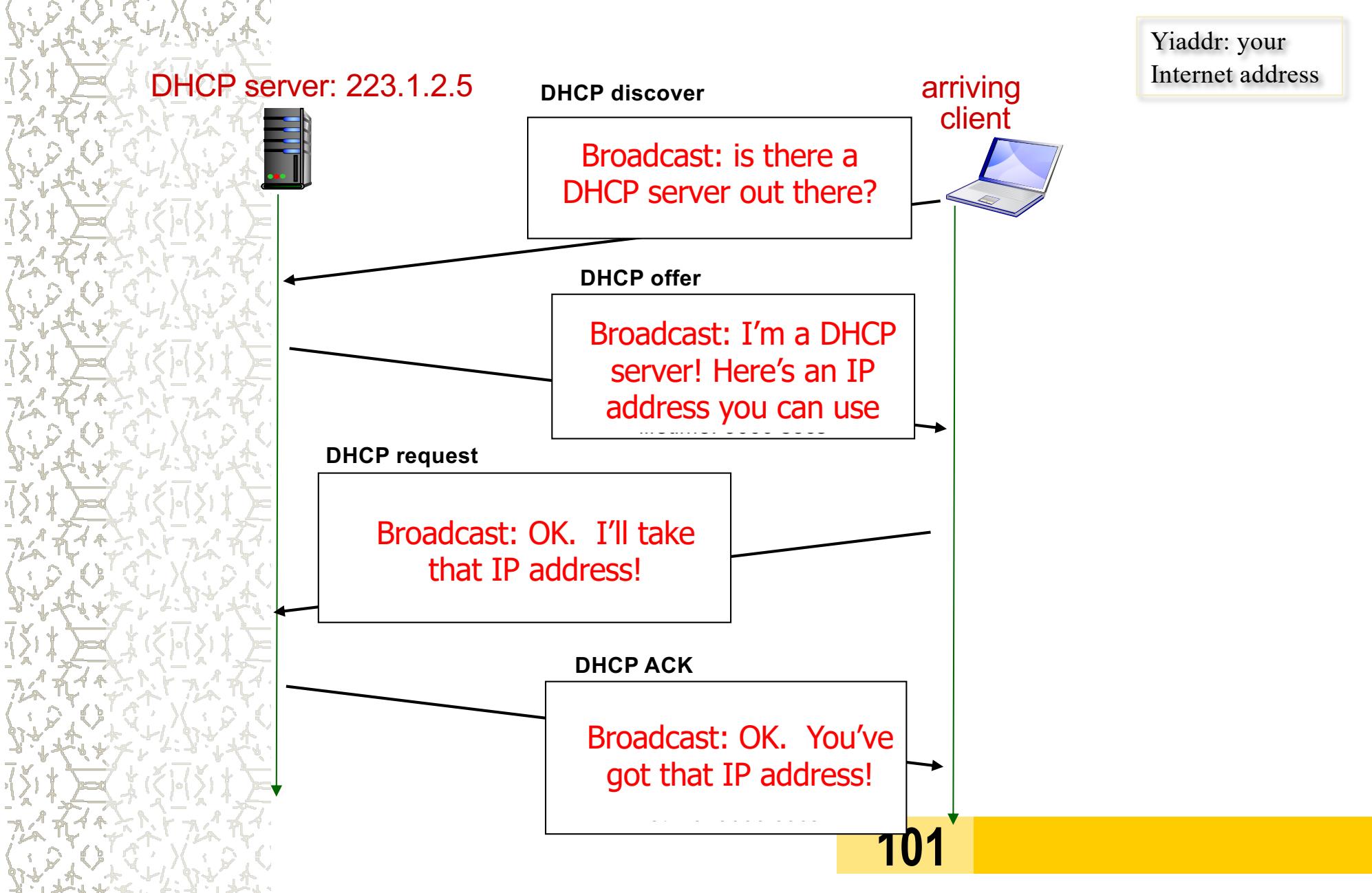
Messaggi DHCP (overview):

- host broadcasts “**DHCP discover**” msg [optional]
- DHCP server responds with “**DHCP offer**” msg [optional]
- host requests IP address: “**DHCP request**” msg
- DHCP server sends address: “**DHCP ack**” msg

DHCP client-server scenario

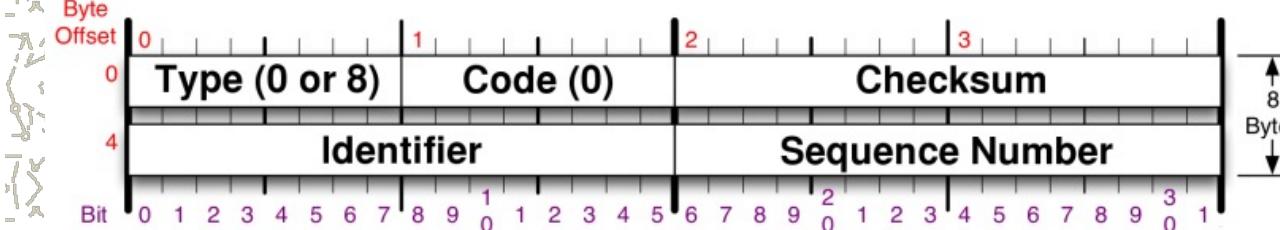


DHCP client-server scenario



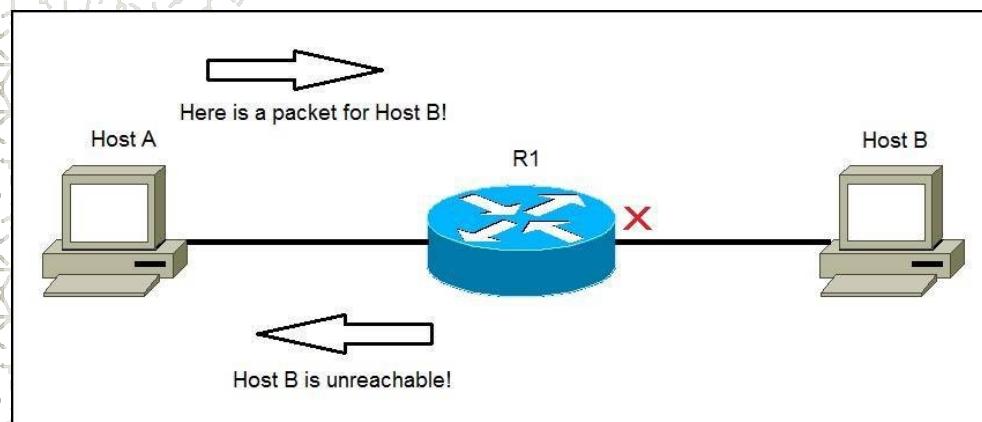
ICMP

- Internet Control Message Protocol è il protocollo utilizzato per monitorare il funzionamento del livello di rete
- Esistono una dozzina di messaggi ICMP destinati ad avvisare i router o gli host di qualche evento specifico della rete
- ICMP non ha lo scopo di rendere IP affidabile, ma di notificare allo strato di rete problemi non transienti nella comunicazione a livello 3 in modo da attivare quelle reazioni dinamiche al malfunzionamento della rete necessarie, ad esempio, a ridisegnare dinamicamente la topologia utilizzata per l'instradamento
- ICMP utilizza IP come protocollo di trasporto per instradare i propri messaggi
 - in questo senso c'è una sorta di miscuglio degli strati in IP:
 - ICMP è una parte del protocollo di rete, in quanto ha funzioni di (auto)controllo dello strato di rete e non fornisce servizi agli strati superiori
 - tuttavia ICMP utilizza IP come sottoprotocollo per trasmettere le sue informazioni tra gli host/router interessati



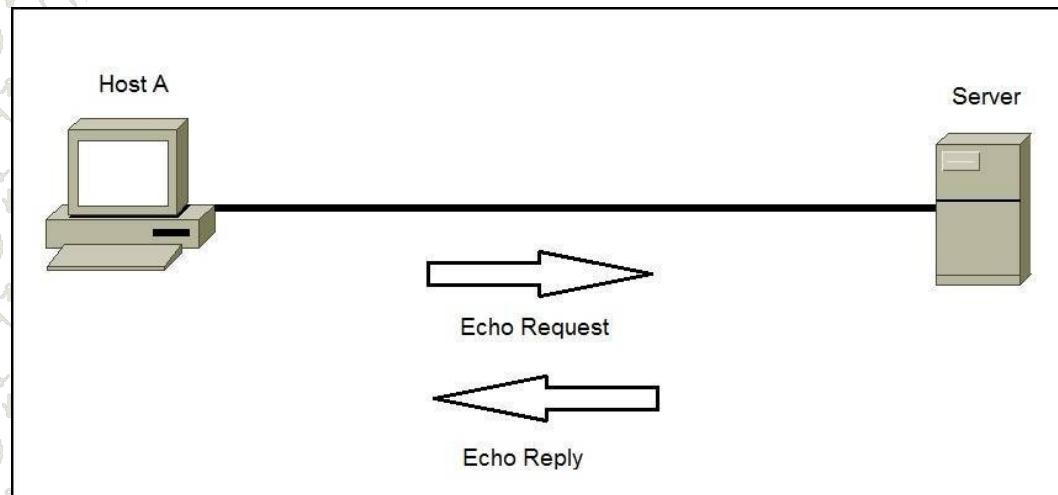
Messaggi ICMP

- I principali messaggi ICMP disponibili sono:
 - **destinazione irraggiungibile**: questo messaggio e' inviato dai router **agli host sorgenti** di un pacchetto IP per notificare che la destinazione non è raggiungibile
 - **time exceeded**: viene notificato **alla sorgente** di un pacchetto che il pacchetto ha raggiunto la **scadenza** del **time-to-live**
 - **problema di parametri**: un router annuncia al router che gli ha inviato un pacchetto che i parametri dell'header sono **inconsistenti**
 - **source quench**: utilizzato (in passato) per **rallentare la sorgente** che trasmette troppo velocemente in caso di congestione; l'evoluzione del TCP/IP ha spostato pero' il controllo della congestione sul **livello di trasporto**
 - **redirect** (reindirizzamento): il router avvisa l'host sorgente che ha inviato il pacchetto iniziale secondo un **instradamento errato** (ad esempio: se ci sono due router sulla LAN, ed un pacchetto viene inviato da un host verso il router sbagliato)



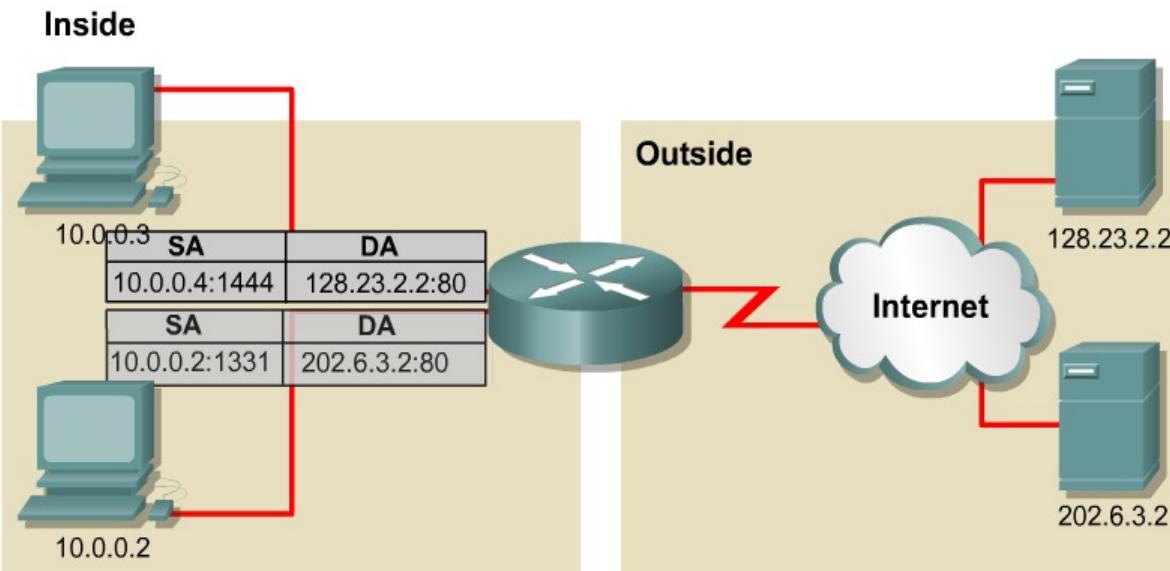
Messaggi ICMP (cont.)

- **echo ed echo reply:** utilizzati per verificare la **raggiungibilità** di un host:
 - quando un host riceve un ICMP ECHO da una sorgente, deve **immediatamente** rispondere con un ICMP ECHO REPLY
 - molte utility fanno uso di questo messaggio ICMP (ad esempio **ping**)
- **timestamp e timestamp response:** analoghi ai messaggi ECHO/ECHO REPLY, inseriscono nei pacchetti **informazioni di tempo** per valutare il **ritardo** della connessione



NAT

- Il router traduce indirizzi interni in esterni, mappando indirizzi e porte nella NAT Table

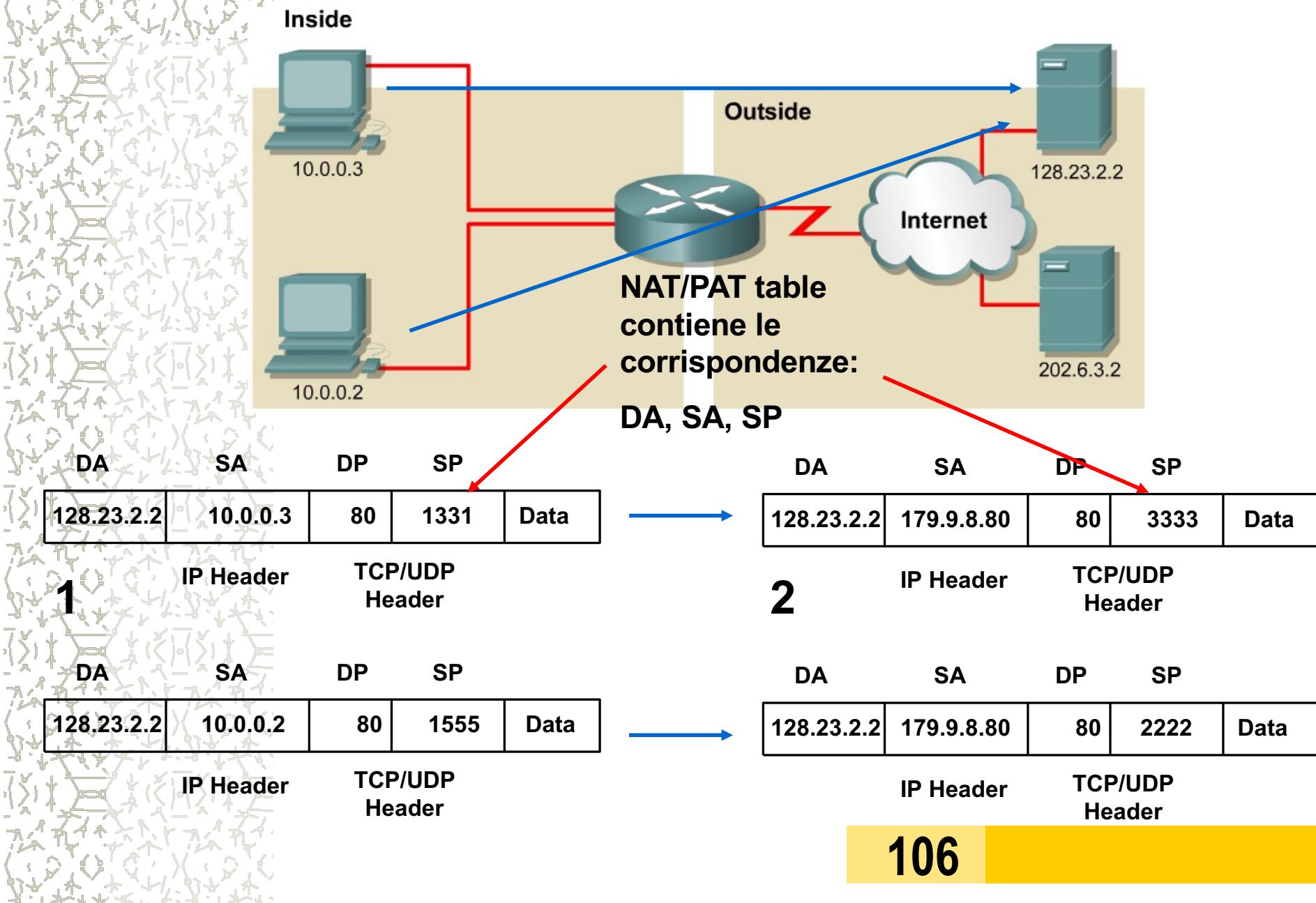


NAT Table

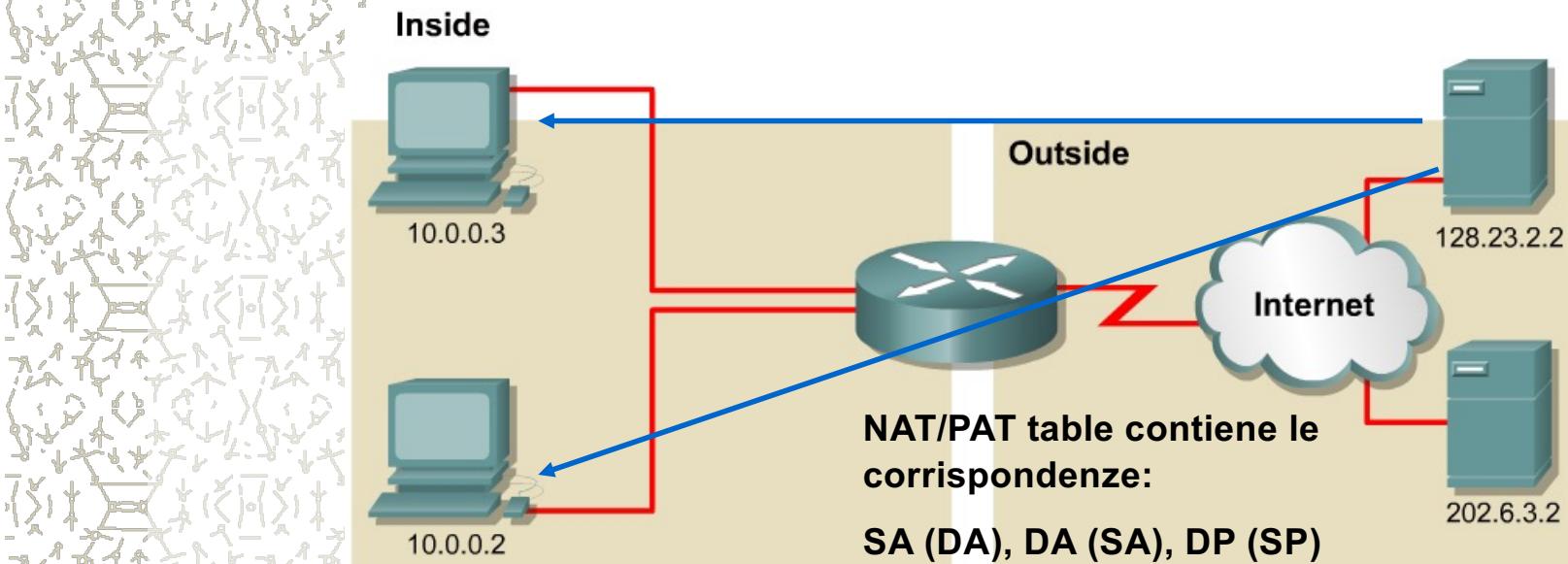
Inside Local IP Address	Inside Global IP Address	Outside Local IP Address	Outside Global Address
10.0.0.2:1331	179.9.8.20:1331	202.6.3.2:80	202.6.3.2:80
10.0.0.3:1555	179.9.8.20:1555	128.23.2.2:80	128.23.2.2:80

- Inside local** – L'indirizzo IP address assegnato all'host sulla rete interna.
- Inside global** – Un indirizzo IP pubblico assegnato dal service provider che rappresenterà uno o più IP locali verso il mondo esterno.
- Outside local** – L'indirizzo IP assegnato a un host sulla rete esterna come visto dagli hosts sulla rete interna
- Outside global** – L'indirizzo IP assegnato a un host sulla rete esterna.

NAT (cont)



NAT (cont)



DA	SA	DP	SP	Data
10.0.0.3	128.23.2.2	1331	80	Data

4

IP Header

TCP/UDP
Header

DA	SA	DP	SP	Data
10.0.0.2	128.23.2.2	1555	80	Data

IP Header

TCP/UDP
Header

DA	SA	DP	SP	Data
179.9.8.80	128.23.2.2	3333	80	Data

3

IP Header

TCP/UDP
Header

DA	SA	DP	SP	Data
179.9.8.80	128.23.2.2	2222	80	Data

IP Header

TCP/UDP
Header



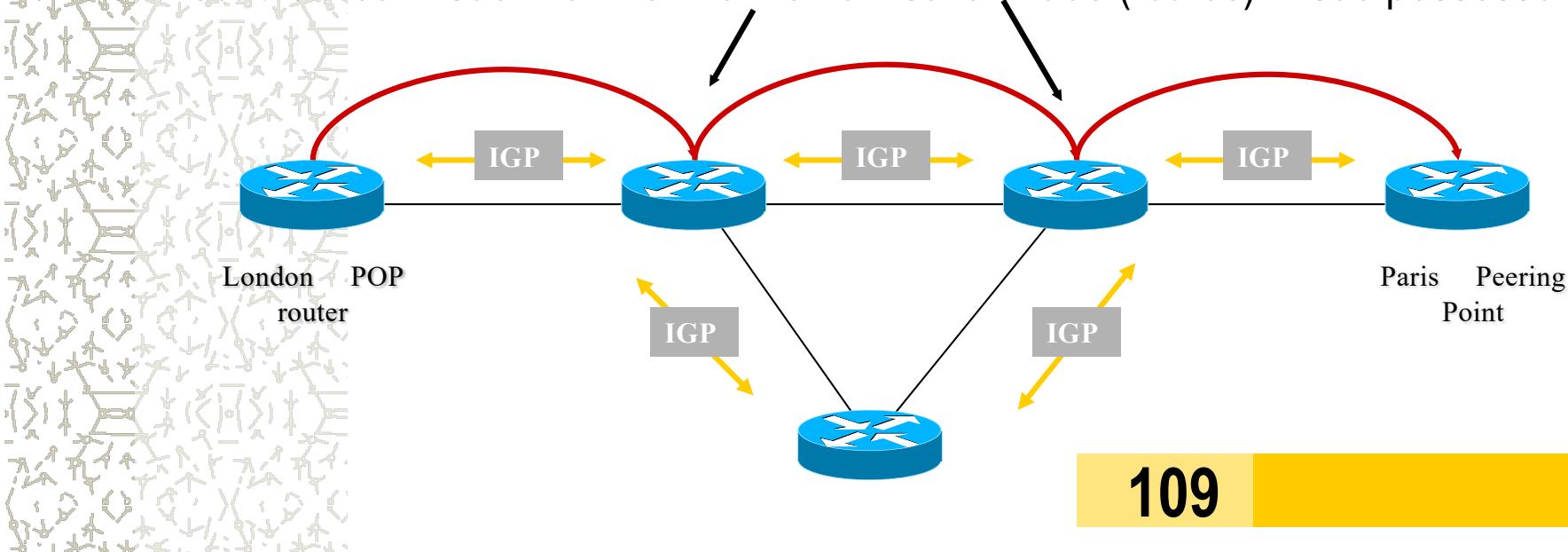
Principi del routing IP

Il routing IP è composto di due momenti:

- *routing control*: scambio di informazioni di routing tra i nodi della rete (*routing protocol*) per la definizione delle tabelle di routing (processo continuo)
- *packet forwarding* (decisione di instradamento per ogni pacchetto, basata sul solo indirizzo di destinazione)

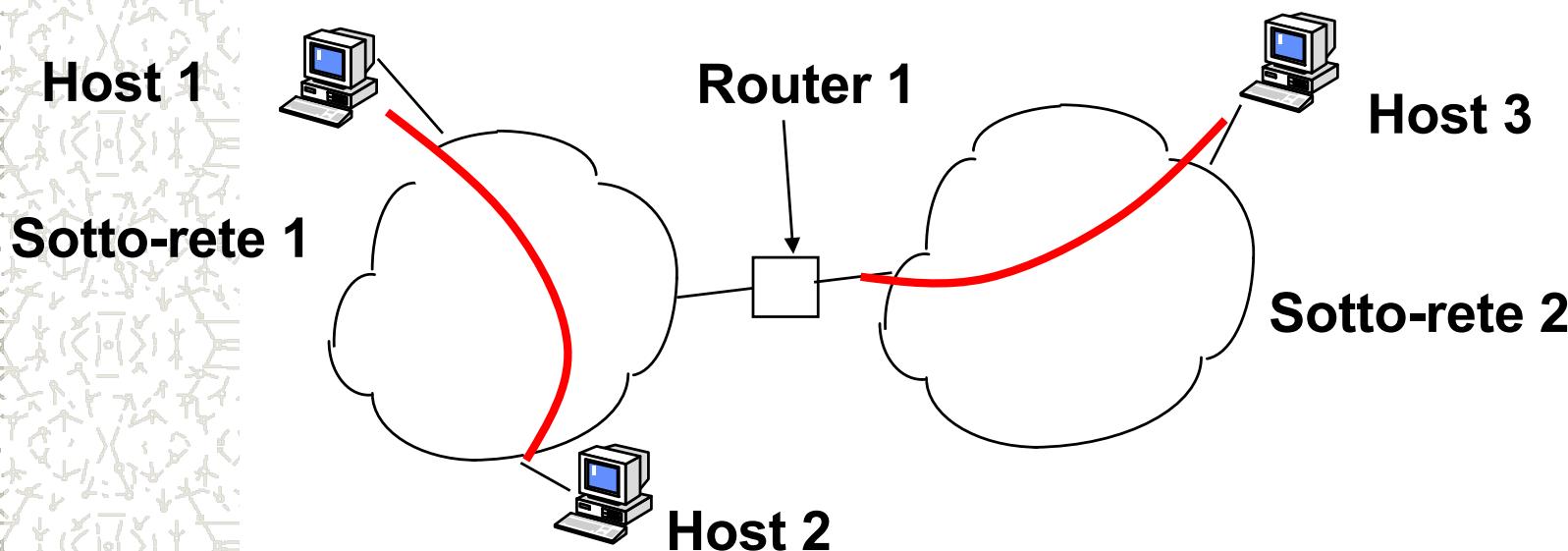
Principi del routing IP (continua)

- Il routing IP segue una **politica di best-effort** dove i pacchetti possono essere:
 - Ritardati
 - Duplicati
 - Distribuiti fuori ordine
 - Persi
- Possono cambiare percorso da pacchetto a pacchetto dello stesso messaggio
- Decisione di routing ad ogni hop: ogni router ha le proprie tabelle di routing, e scambia con i suoi vicini le informazioni sulle strade (*routes*) in suo possesso



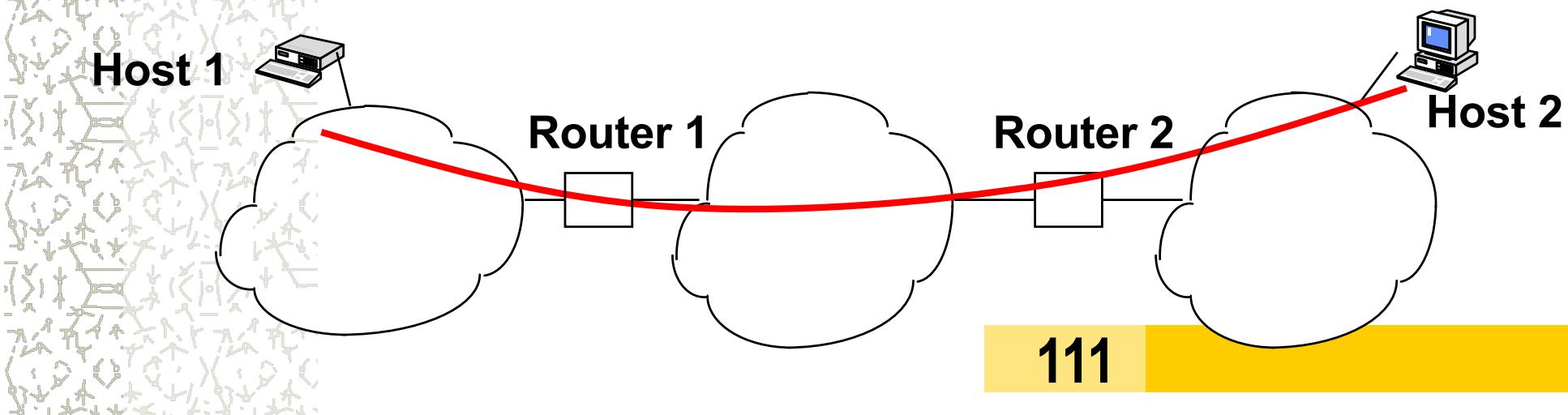
Instrandamento diretto

- La trasmissione di un datagramma IP tra due macchine connesse su una stessa sotto-rete (stesso Net_id)
- Non coinvolge router intermedi
- Il trasmittitore IP risolve l'indirizzo fisico dell'host destinatario (tramite il protocollo ARP), incapsula il datagramma nell'unità dati della rete fisica e lo invia verso destinazione
- Utilizza i meccanismi propri della rete fisica in questione per inviare il datagramma



Instradamento indiretto

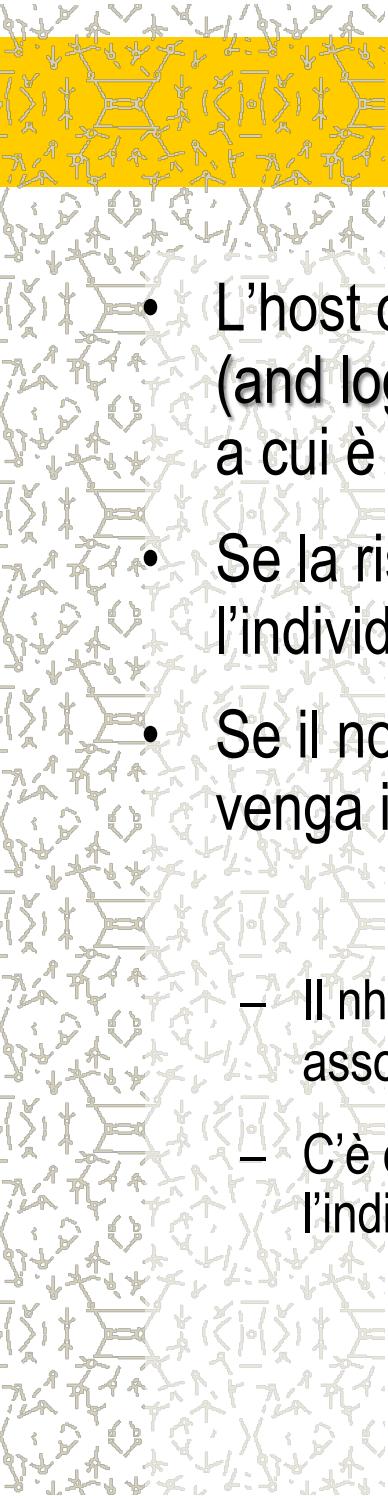
- L' host di destinazione non è sulla stessa sotto-rete del mittente
- Il mittente deve identificare un router a cui inviare il datagramma; il router deve inviare il datagramma verso la sotto-rete di destinazione.
- Il router esamina il datagramma IP ricevuto e, se l'host di destinazione non si trova in una sottorete a cui il router è direttamente connesso, decide il router successivo verso cui instradarlo
 - l'instradamento attraverso la sotto-rete che connette i due router avviene secondo i meccanismi della sotto-rete
- Il processo si ripete di router in router sino alla sotto-rete di destinazione





Forwarding di datagrammi IP

- E' il processo che consente a un pacchetto di essere trasferito da un ingresso a una uscita del nodo e di nodo in nodo di essere trasportato dalla sorgente alla destinazione
- Ipotesi
 - Ogni datagramma contiene l'indirizzo IP della destinazione
 - La parte di rete identifica in modo univoco la rete nell'ambito di Internet
 - Host e router con lo stesso indirizzo di rete si scambiano i pacchetti su quella rete
 - Ogni rete che è parte di Internet ha almeno un router collegato ad un'altra rete



Funzionamento

- L'host o il router stabilisce per confronto con il proprio indirizzo di rete (and logico) se la destinazione appartiene o meno alla rete o alle reti a cui è connesso
 - Se la risposta è positiva si innesca la procedura ARP per l'individuazione dell'indirizzo fisico
 - Se il nodo è connesso a una rete diversa occorre che il datagramma venga inviato a un router (next hop router)
 - Il nhr si determina leggendo la tabella di forwarding che è una lista di associazioni (numero di rete, next hop)
 - C'è comunque un default router a cui viene inviato il pacchetto nel caso l'indirizzo non venga trovato nella tabella precedente

IP Routing/Forwarding

Dato un datagramma:

1. Estrai il campo destinazione *DA* (Destination address)
2. Cerca *DA* nella routing table
3. Trova il prossimo “hop address”: *HA*
4. Spedisci il datagramma a *HA*

Concetto chiave

L’ indirizzo di destinazione nell’ header del datagramma si riferisce sempre all’ ultima destinazione.

Quando un router invia il datagramma ad un altro router, l’ indirizzo del “next hop” non appare nell’ header del datagramma.



Conseguenze (1)

- ad ogni “*hop*” viene “ricalcolata” la strada da seguire per tutti i pacchetti in transito
- i router devono poter sapere instradare tutti gli indirizzi => accrescimento tabelle di routing
- non c’è distinzione tra i tipi di servizio e le loro esigenze in termini di qualità di servizio
- *connectionless*: non si possono definire percorsi end-to-end o indirizzare il traffico in determinati percorsi

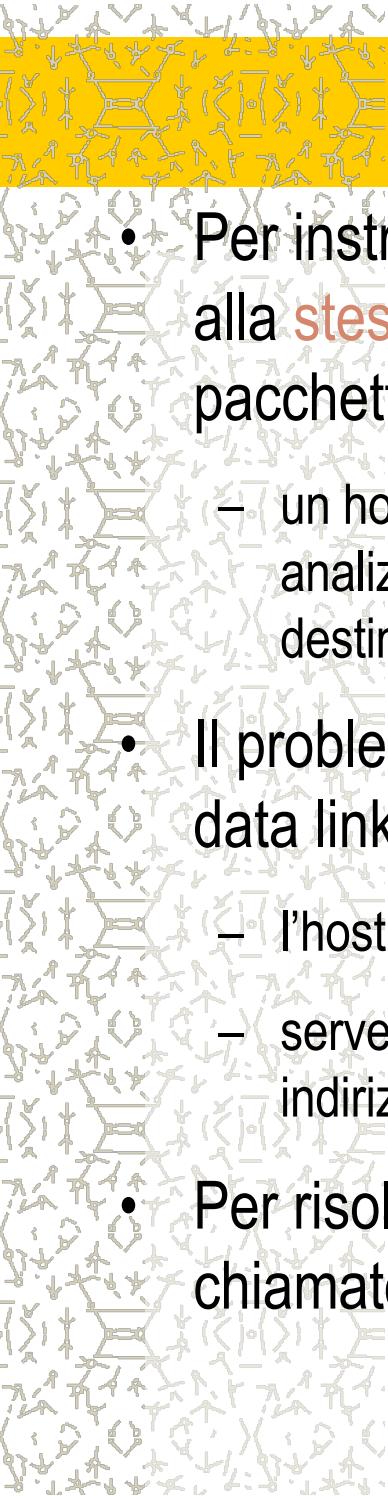


Conseguenze (2)

- Di fronte alla crescita “esponenziale” della Rete e della sua complessità, la soluzione è spesso stata di sovra-dimensionare l’infrastruttura:
 - trunk ad alta capacità
 - nodi ad elevato *throughput*
 - router con capacità d’instradamento potenziata al massimo

Determinazione degli indirizzi fisici

- Occorre determinare l'indirizzo di linea corrispondente all'indirizzo IP per poter trasferire fisicamente il datagramma
 - Ad esempio indirizzo Ethernet a 48 bit
- Ogni host costruisce una tabella di corrispondenze utilizzando il protocollo ARP (Address Resolution Protocol)
- La tabella si chiama ARP cache o ARP table e scade periodicamente



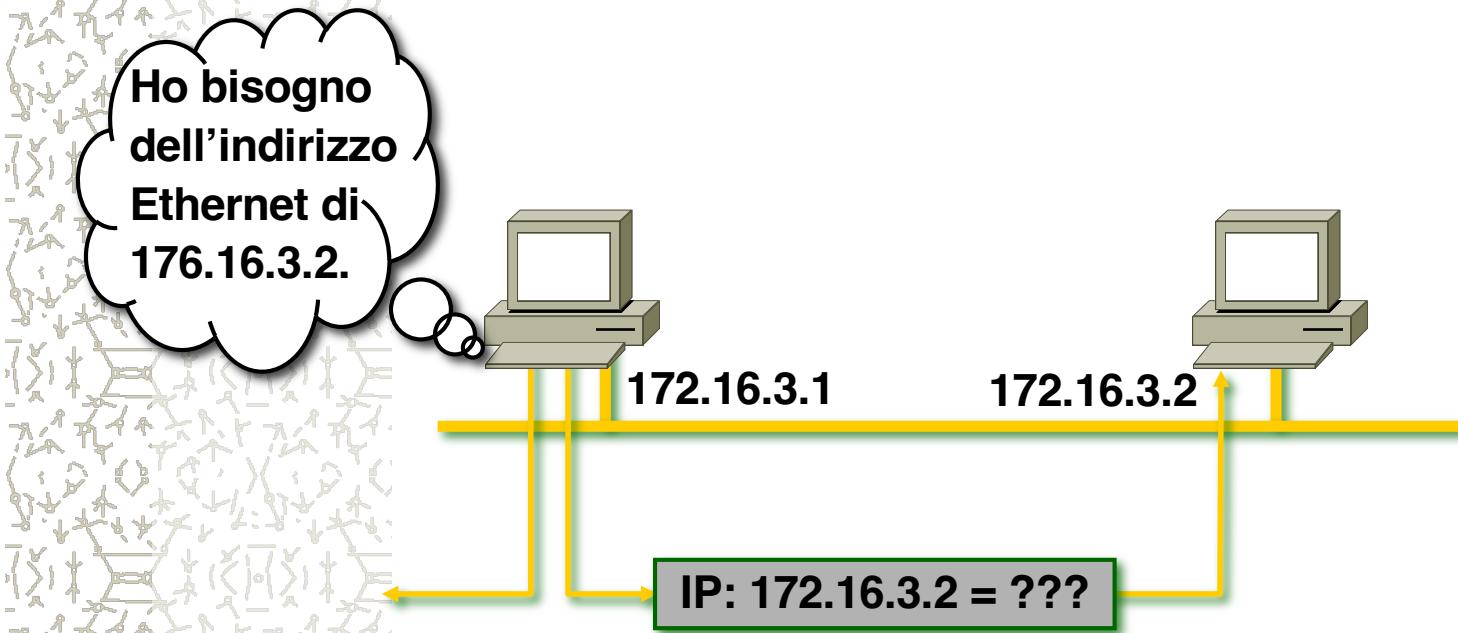
Instradamento locale: ARP

- Per instradare un pacchetto IP verso una destinazione appartenente alla **stessa rete** del mittente viene **incapsulato** il pacchetto IP in un pacchetto dello strato di data link **sottostante** (ad esempio: Ethernet)
 - un host **conosce** il proprio indirizzo IP e la propria rete di appartenenza: analizzando l'indirizzo di destinazione di un pacchetto l'host **può capire** se il destinatario appartiene alla sua **stessa rete**, e quindi operare il delivery locale
- Il problema da risolvere è **come** fare a sapere a quale indirizzo di data link (Ethernet) inviare il pacchetto
 - l'host conosce **solo** l'indirizzo IP del destinatario
 - serve quindi una **mappa** che associa un indirizzo IP della stessa rete al suo indirizzo di data link
- Per risolvere questo problema IP si appoggia ad un protocollo chiamato ARP (**Adderss Resolution Protocol**)

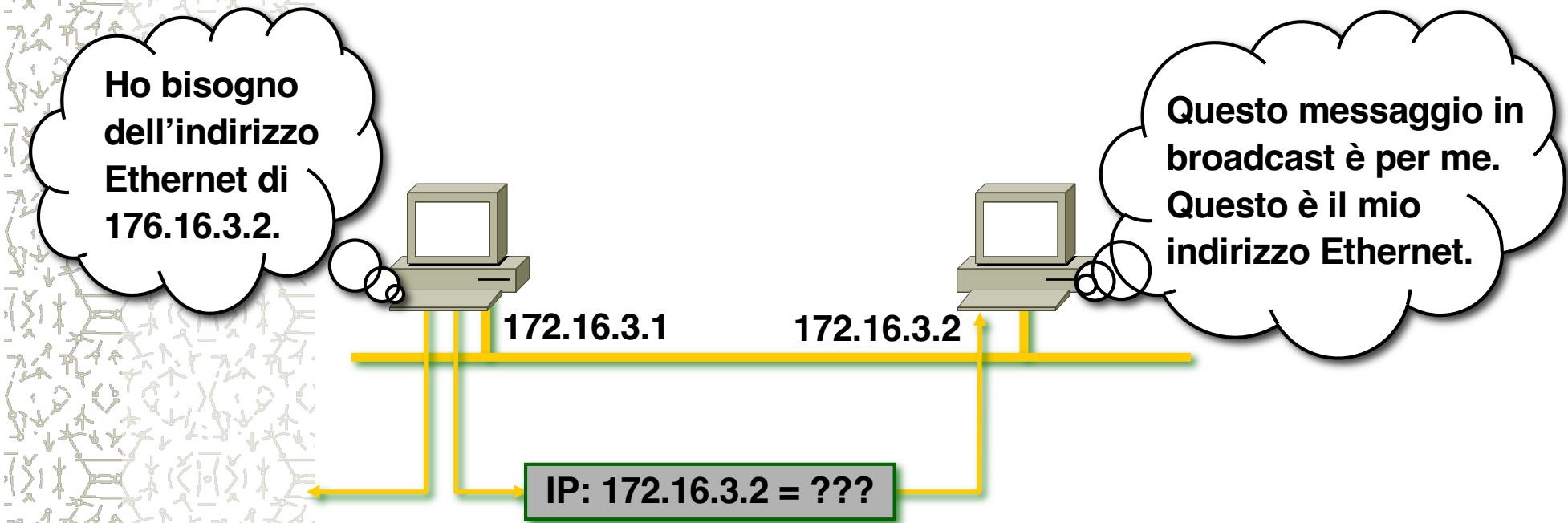
Address Resolution Protocol (ARP)

- Quando un host con indirizzo **IP1** ed indirizzo hardware **HW1** deve inviare un pacchetto IP ad un host con indirizzo **IP2** sulla stessa rete, ARP si procura l'informazione necessaria in questo modo:
 - viene costruito un pacchetto di data link (**ARP request**) contenente IP1, HW1, ed IP2, con un campo **dedicato** ad **HW2** riempito con tutti 0
 - questo pacchetto viene inviato **broadcast** sulla rete locale
 - **tutti** ricevono il pacchetto ARP, ma solo l'host che **ha l'indirizzo IP2** lo processa (gli altri lo scartano)
 - l'host costruisce un pacchetto di data link (**ARP response**) contenente l'informazione mancante, e lo invia **direttamente ad HW1** (non broadcast)
 - ARP sul primo host **acquisisce** quindi l'informazione dell'indirizzo Ethernet dell'host remoto, e lo **comunica ad IP**, che può così encapsulare i propri pacchetti IP in frame del protocollo di data link indirizzati alla destinazione corretta

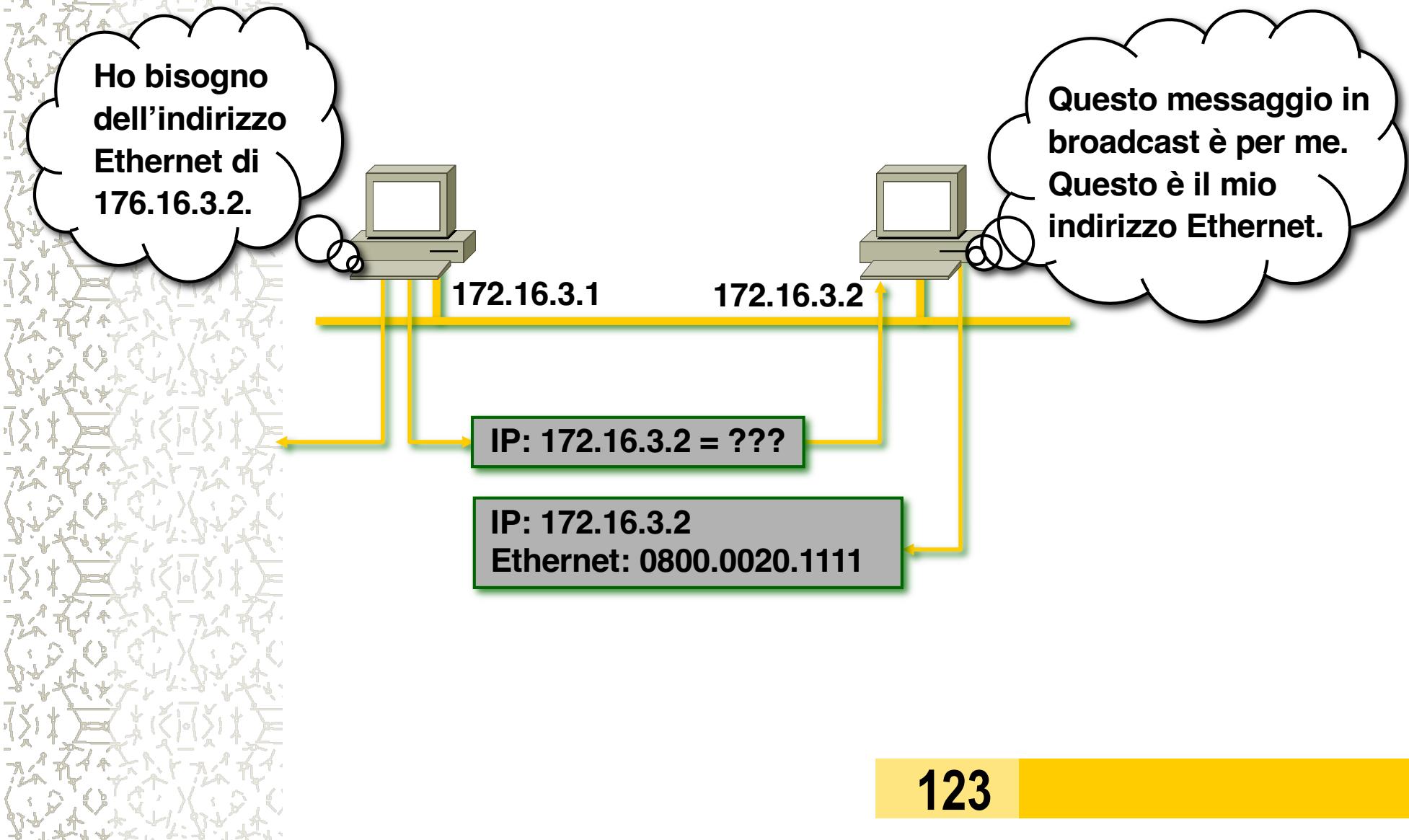
Address Resolution Protocol



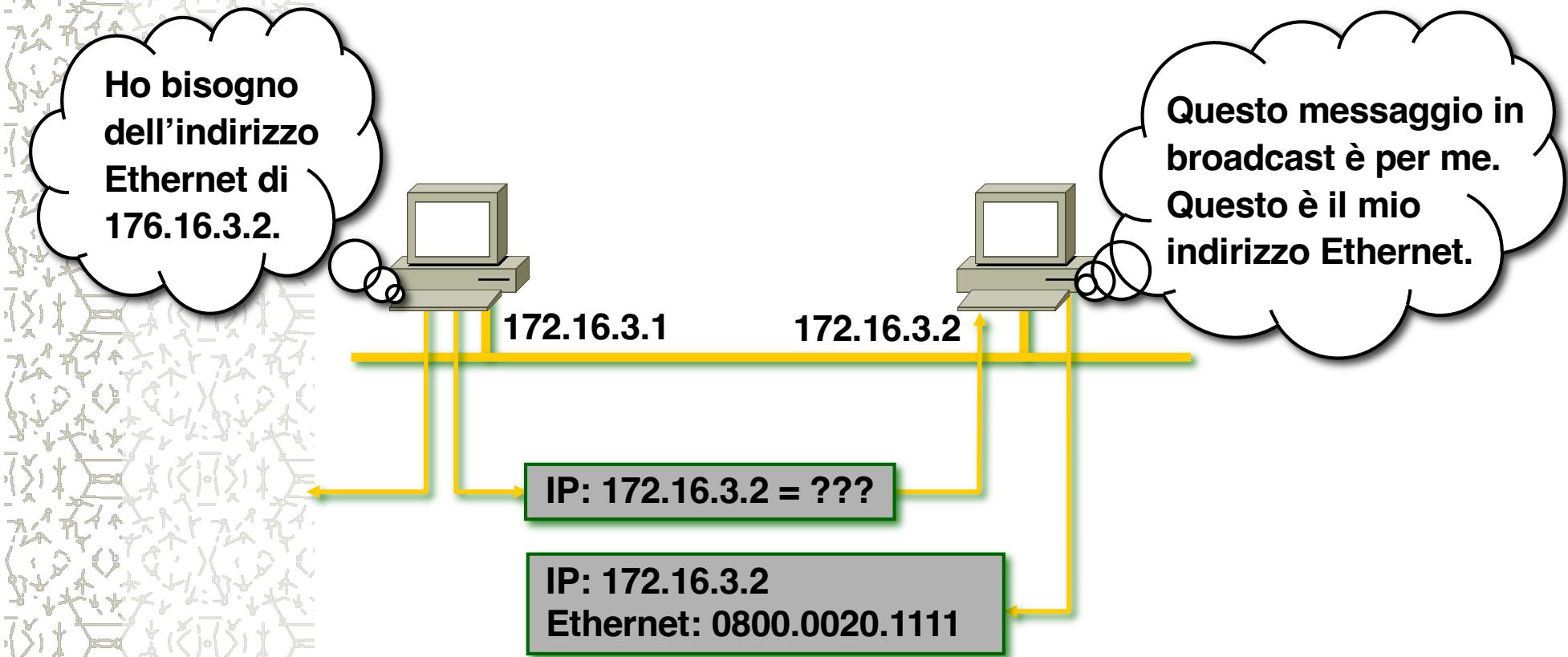
Address Resolution Protocol



Address Resolution Protocol



Address Resolution Protocol



- Map IP \longrightarrow Ethernet
- Local ARP

ARP cache

- Per migliorare le prestazioni, ARP può gestire sull'host locale una **cache** in memoria
- Ogni volta che viene appresa una **nuova** associazione IPaddress-Hwaddress, viene **memorizzata** nella cache
- Quando ARP deve individuare un indirizzo HW, prima **controlla nella cache**: se l'informazione e' presente viene utilizzata **senza inviare pacchetti** sulla rete
- Le entry nella cache di ARP hanno un **tempo di scadenza**, per evitare che eventi quali **sostituzione di schede** di rete o **reindirizzamento** degli host possano rendere impossibile la comunicazione
 - alla scadenza del tempo di validita' l'entry viene **rimossa** dalla cache, ed una successiva richiesta per quell'indirizzo provochera' una nuova emissione di ARP request sulla LAN
- Alcuni sistemi permettono di definire nella cache di ARP delle entry **manuali** prive di scadenza
 - talvolta necessarie, qualora l'host di destinazione **non supporti correttamente** il protocollo ARP
 - questa tecnica puo' essere utilizzata anche per motivi di **efficienza**
 - in ogni caso **difficile da mantenere** aggiornata la cache delle macchine: meglio evitare

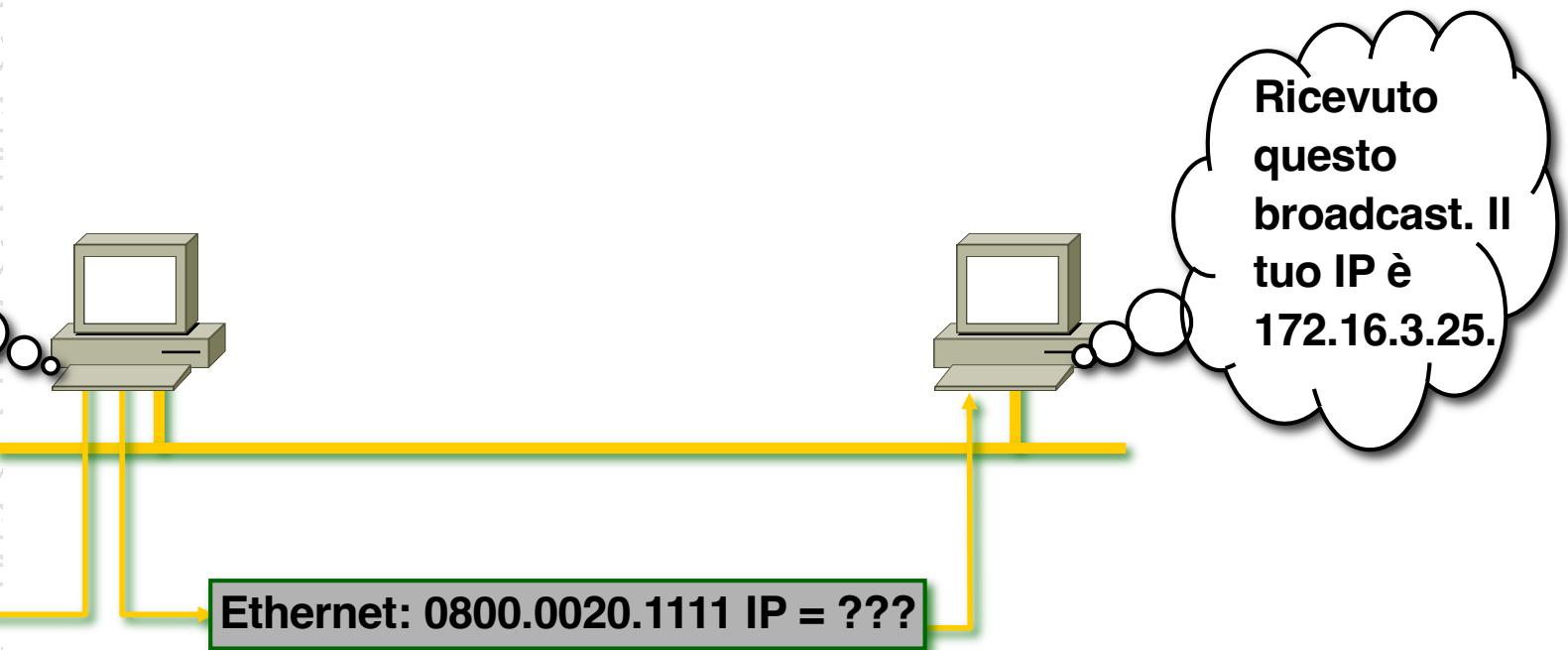
Reverse ARP

- Consente di determinare l'indirizzo IP a partire dall'indirizzo fisico
- Serve quando si accende una workstation diskless
- Ogni rete ha un suo RARP server

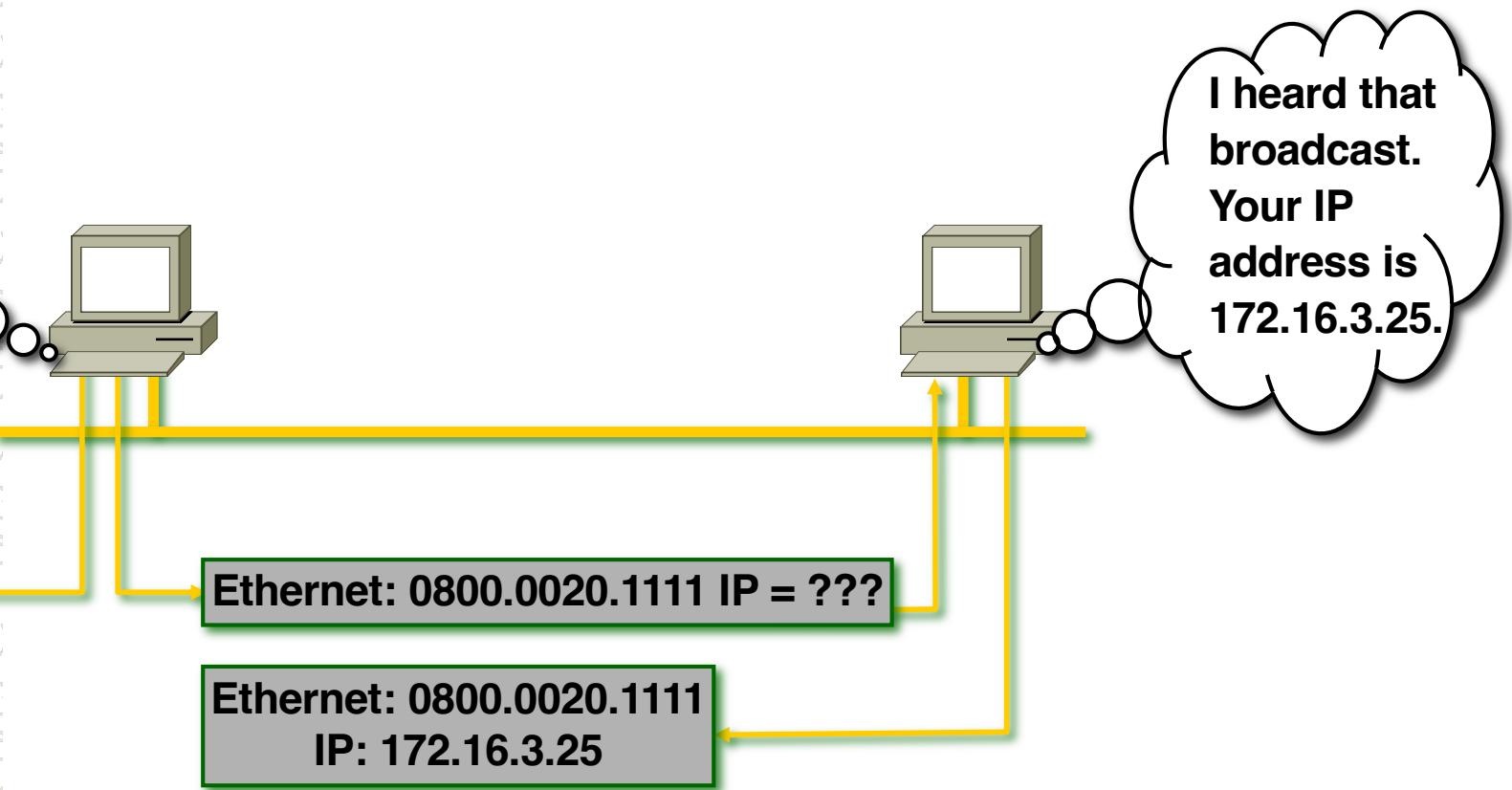
Reverse ARP



Reverse ARP



Reverse ARP



Reverse ARP

