Reti di Telecomunicazioni

Network Layer

Autori

Queste slides sono state scritte da

MicheleMichelotto

michele.michelotto@pd.infn.it

che ne detiene i diritti a tutti gli effetti

Copyright Notice

Network layer

Introduzione
Layer: Modello OSI e TCP/IP
Physics Layer
Data Link Layer
MAC sublayer
Network Layer
Transport Layer
Application Layer

Network layer di internet

- Vediamo prima i principi base che hanno ispirato la progettazione nel passato e reso internet un successo ai nostri giorni
- Consiglio la lettura della RFC 1958 di Brian Carpenter in cui i principi base sono elencati
- Vediamo in ordine di importanza decrescente questi principi

1: Deve Funzionare

- Siamo sicuri che funzioni: migliaia di prototipi devono aver funzionato l'uno con l'altro prima di essere sicuri che lo standard è stato progettato correttamente.
- Inutile scrivere 1000 pagine di standard per accorgersi di un errore e uscire poi con la versione 1.1

2: Keep it Simple

- KISS principle: Nel dubbio scegli la soluzione più semplice
- In pratica: Non aggiungere un sacco di features. Se una feature non è essenziale lasciala fuori, soprattutto se puoi ottenere lo stesso effetto combinando altre features

3: Scelte chiare

- Se ci sono diversi modi per fare una cosa scegline uno e scarta gli altri
- Avere due o più modi per fare la stessa cosa è andare in cerca di guai

4: Sfrutta la modularità

- Questo principio conduce all'idea di avere degli stacks di protocolli
- Ogni layer indipendente dagli altri
- Quando, a causa di nuovi requirements o nuove opportunità tecnologiche, devo cambiare un modulo o un layer, gli altri non vengono colpiti

5: Aspettati eterogeneità

- Diverse tipi di hardware, di strumenti trasmissivi, di applicazioni sono presenti in una rete molto grande
- Per gestire tutti questi oggetti devono avere una rete progettata in modo semplice, generale e flessibile

6: Evita parametri statici

- Se devo inserire un parametro, per esempio una dimensione massima di un pacchetto, è meglio che sender e receiver possano negoziare un valore piuttosto che ci siano delle scelte predefinite.

7: Meglio Buono o Perfetto?

- Bisogna puntare ad un progetto buono, non deve essere perfetto
- Spesso il progettista arriva ad un buon progetto ma non riesce a gestire degli strani casi particolari.
- Piuttosto che rovinare il progetto è meglio accontentarsi di quanto ottenuto e delegare la soluzione tecnica a chi arriva con gli strani casi particolari

8: Rigido o Tollerante?

- Bisogna essere di stretta osservanza alle regole quando si trasmette e molto tolleranti quando si riceve
- I pacchetti che si mandano devono rispettare strettamente gli standard
- Bisogna aspettarsi che i pacchetti che si ricevono siano leggermente non standard e cercare di gestirli in qualche modo

9: Scalabilità

- Se il sistema deve gestire milioni di nodi e miliardi di utenti in modo efficiente, non ci deve essere un database centralizzato
- Il carico deve essere distribuito nel modo più equilibrato possibile su tutte le risorse disponibili

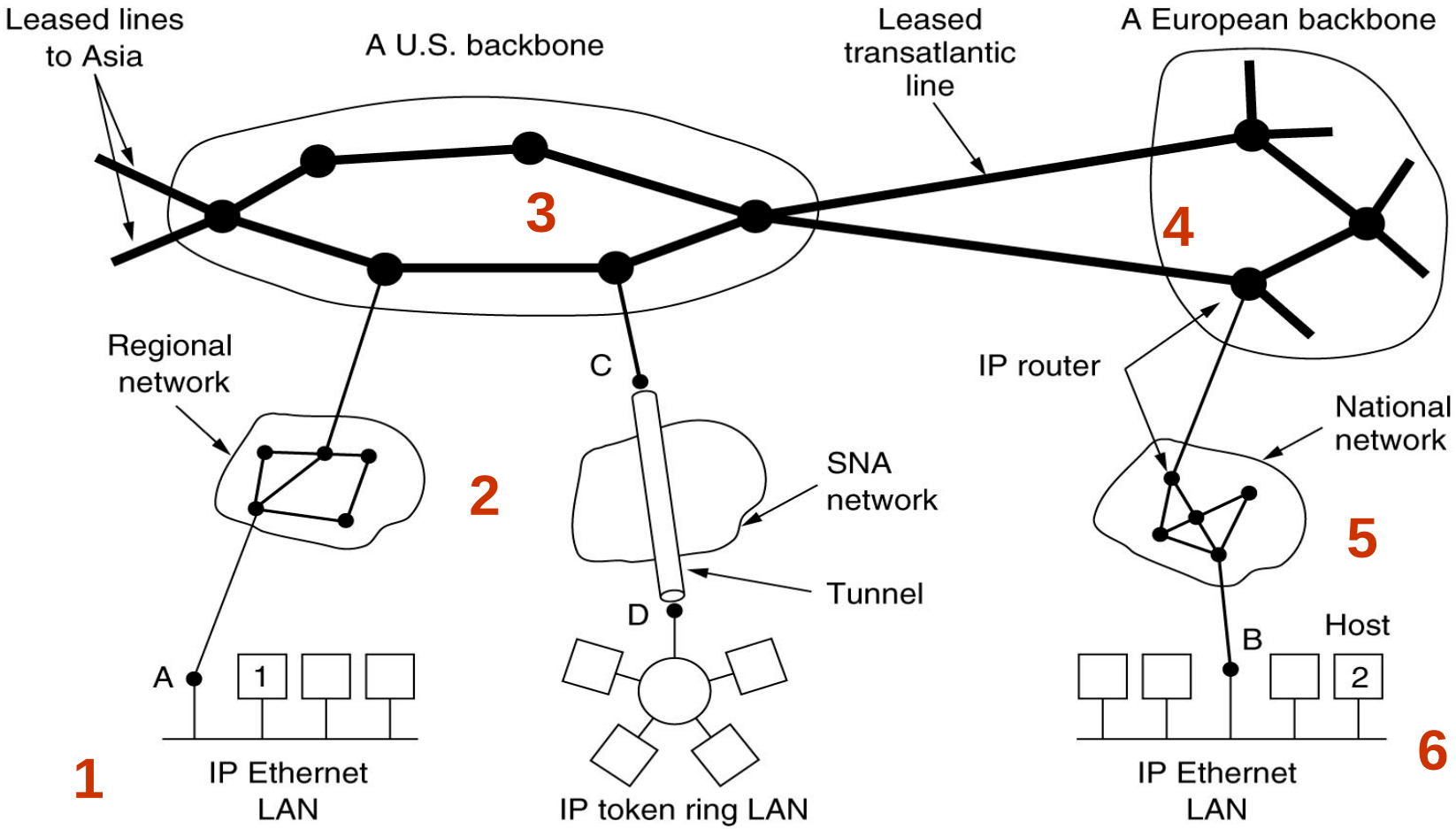
10: Costi e prestazioni

- Se un sistema ha costi troppo elevati o prestazioni troppo deludenti nessuno lo usa.

Autonomous System

- A livello network Internet viene vista come un insieme di sottoreti, dette “Autonomous System”, interconnesse
- Non esiste una vera e propria struttura, ci sono invece dei backbone con linee e router ad alte prestazioni.
- Attaccati ai backbone ci sono reti regionali o di organizzazione
- Attaccate alle reti regionali ci son le diverse LAN di compagnie, università e ISP

internet

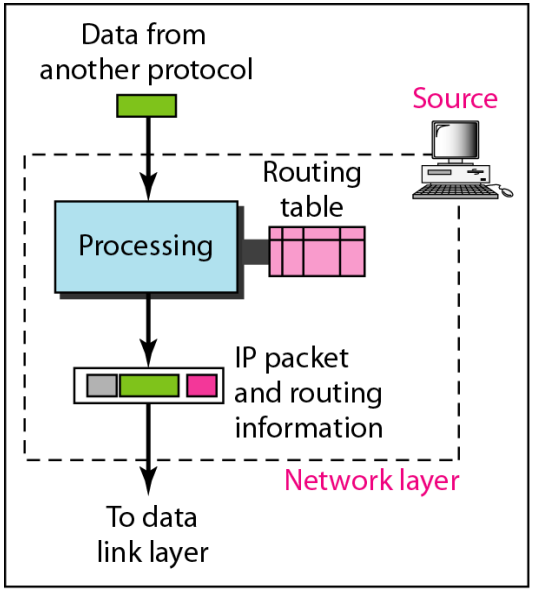

Leased lines to Asia
A U.S. backbone
Leased transatlantic line
A European backbone
3
4
Regional network
C
IP router
National network
2
SNA network
5
Tunnel
D
Host
B
A
1
2
1
IP Ethernet LAN
IP token ring LAN
IP Ethernet LAN
6

IP

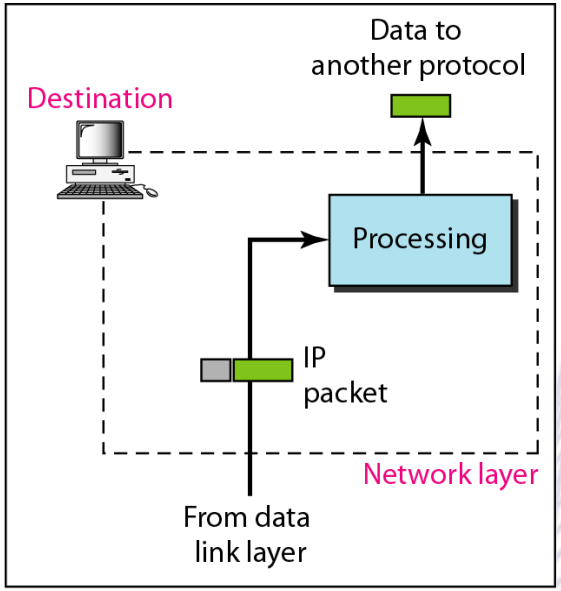
- La colla che tiene tutto insieme è un comune protocollo di livello network chiamato **IP** (**Internet Protocol**)
- A differenza di altri protocolli di livello network, questo è stato progettato fin dall'inizio con l'idea dell'internetworking
- Fornisce un servizio best-effort (non garantito) che trasporta datagram da una sorgente ad un destinatario

source, destination e router

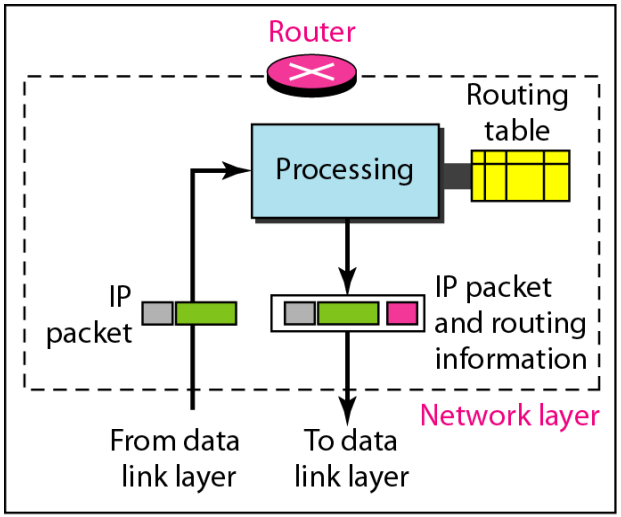
- Il livello network al mittente (a), destinatario (b) e in router intermedio (c)



a. Network layer at source



b. Network layer at destination



c. Network layer at a router

Comunicazione su IP

- Il layer di trasporto prende uno stream di dati e lo spezza in datagrams
- Questi possono essere grandi fino a 64KBytes ma in pratica non sono più grandi di 1500 Bytes (quindi stanno in un frame Ethernet)
- Ogni datagram viene spedito nella Internet, magari frammentato in pezzi più piccoli
- Quando tutti i pezzi arrivano a destinazione vengono riassemblati nel datagram originale
- Vedi l'esempio in cui il datagram attraverso 6 reti per andare dal nodo 1 al nodo 2

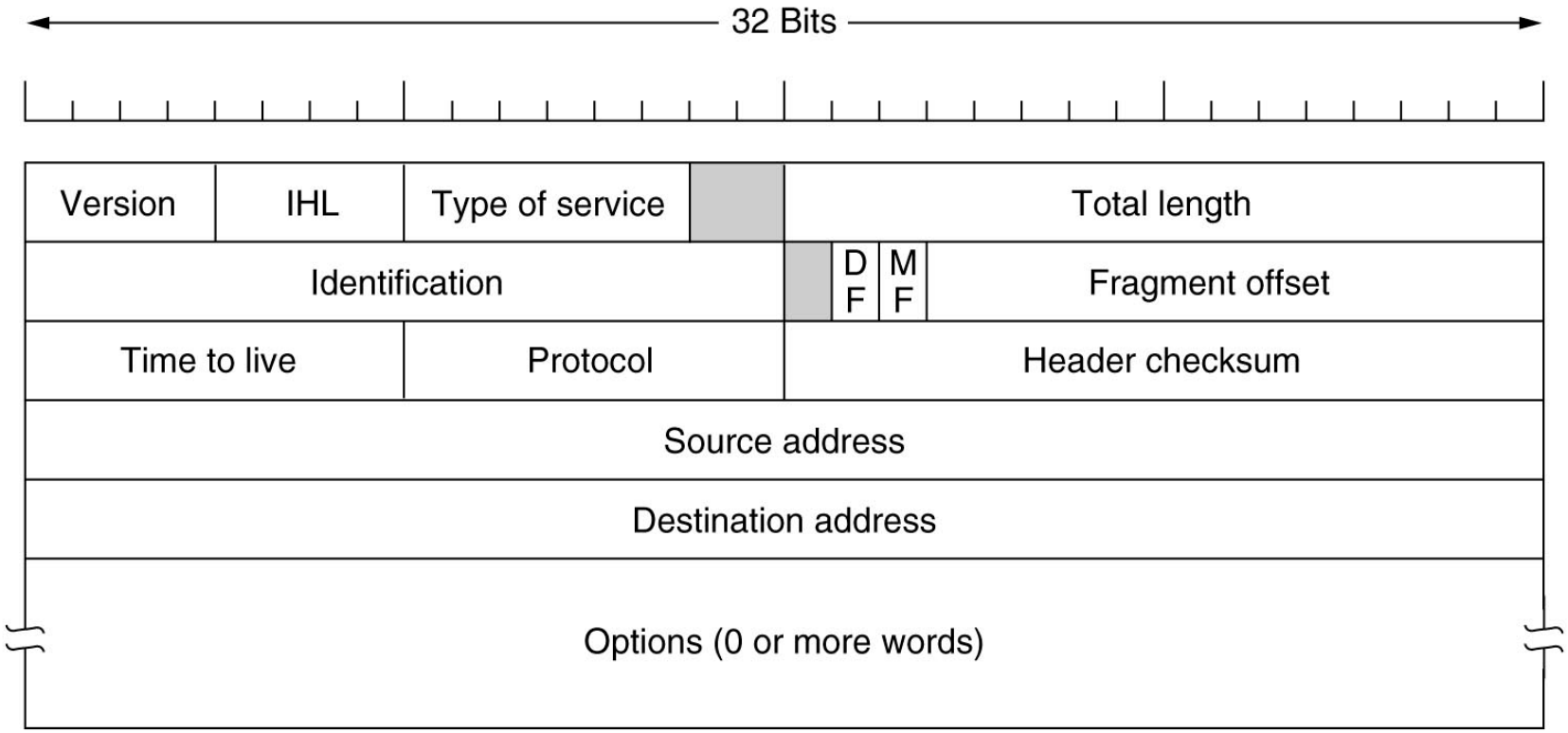
Datagram IP

- È costituito da una parte di header e una parte di dati
- L'header ha una parte fissa di 20 bytes e una parte variabile opzionale
- Il campo "Version" tiene traccia di quale versione del protocollo stiamo usando
  - Questo permette di avere due tipi di protocolli contemporaneamente e di transire da uno all'altro come ora con la transizione IPv4 – IPv6

Header IP

Bit più alti

Bit più bassi



IHL

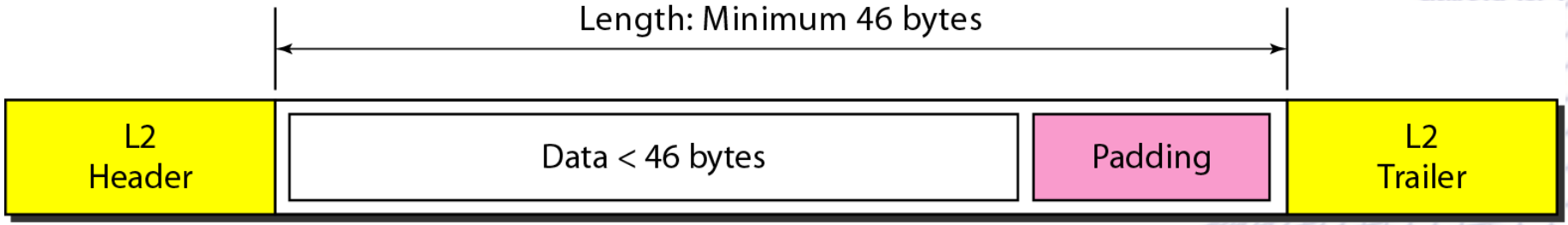
- Il campo IHL ha 4 bit e dice quanto è lungo l'header in termini di parole di 32 bit
- Il valore minimo è 5 per un campo di 20 byte (160bit, 32bit/word) quando nessuna opzione è presente
- Il massimo è 15 per header di 60 Byte per campo opzionale di 40 Byte

TOS

- “Type of Service” è una delle poche campi che hanno cambiato significato nel corso degli anni
- Era stato pensato per distinguere tra diverse classi di servizio
  - Per voce digitalizzata è più importante una consegna veloce che una consegna accurata, al contrario per un file transfer
- All’inizio c’era un campo “Precedence” di 3 bit e 3 flag (**D**,**T**,**R**) per indicare a cosa l’host tenesse maggioramente (**D**elay, **T**hroughput, **R**eliability)
  - In teoria un router potrebbe usare questi campi per decidere se usare un link satellitare al alto throughput ma alto delay o una linea dedicata a basso throughput e basso delay. In pratica i router ignorano questo campo del tutto
- Ora IETF ha deciso di usare questi 6 bit per indicare quale classe di servizio usare per Differentiated Services

Total Length

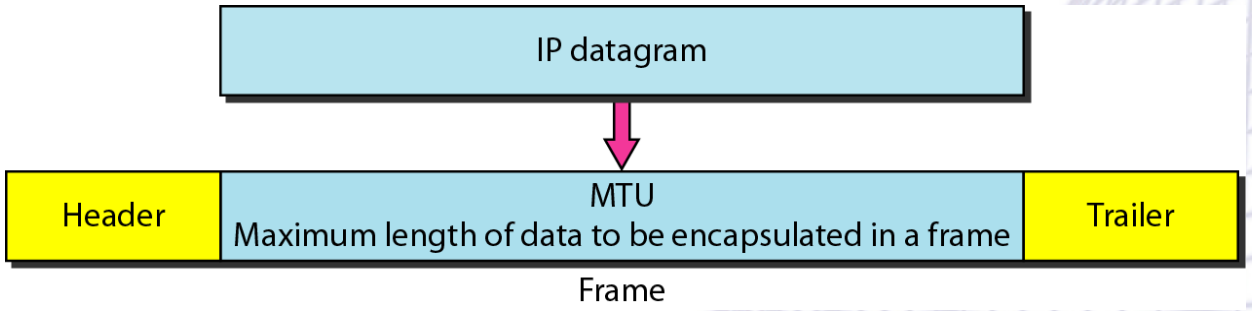
- La lunghezza totale del datagram, compresi header e dati
- Lunghezza massima 65535 Byte
- Per adesso è una lunghezza accettabile, in futuro potrebbero essere necessari datagram più grandi
- NB questo campo è necessario perché a volte i datagram sono riempiti con dati fittizi per accontentare alcune misure minime es. Ethernet



Frammentazione

- Ogni protocollo data link ha una dimensione massima di frame chiamata MTU
- Quando un datagram viene incapsulato in un frame non deve superare questa dimensione che è imposta dalla tecnologia hardware utililizzata e quindi dal protocollo utilizzato nel layer fisico
- Dal momento che quasi tutti i protocolli hanno un MTU più piccolo i datagram devono essere frammentati
- Questa operazione avviene già nel nodo mittente ma potrebbe darsi che il pacchetto poi attraversi link intermedi con MTU più piccola
- Il riassemblaggio poi avviene solo alla destinazione finale
- NB è impossibile garantire che i frammenti viaggino per lo stesso percorso e arrivino in ordine

Protocol	*MTU*
Hyperchannel	65,535
Token Ring (16 Mbps)	17,914
Token Ring (4 Mbps)	4,464
FDDI	4,352
Ethernet	1,500
X.25	576
PPP	296



Identification

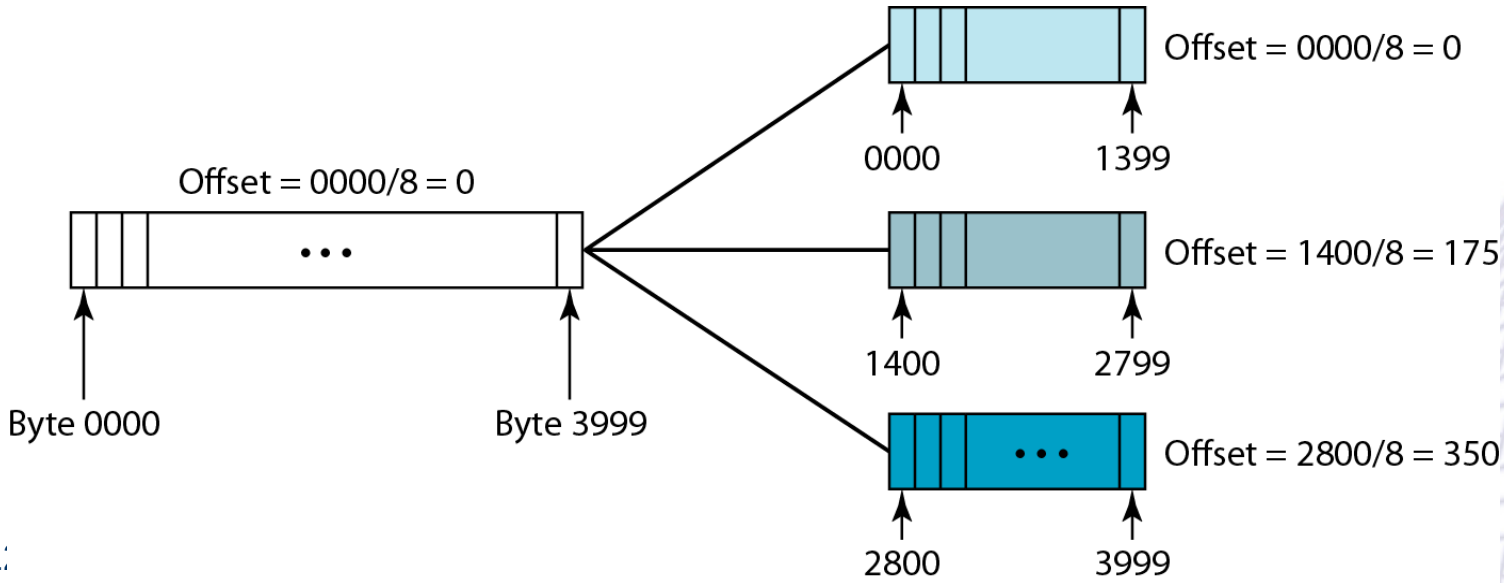
- Permette all'host di destinazione di capire a quale datagram appartiene il frammento che è arrivato
- Tutti i frammenti di un certo datagram contengono lo stesso valore del campo **Identification**
- È lungo 16 bit quindi dopo 2^{16} incrementi si riazzera, comunque un numero molto grande, ci vuole parecchio tempo per cui in una rete lo possiamo considerare come un numero unico

Fragment Offset

- Dopo un bit non usato si trova un bit **DF** (**Don't Fragment**) che ordina al router di non frammentare il pacchetto perché il ricevente non lo sa riassemblare
  - Vuol dire che per arrivare a destinazione potrebbe passare per una route non ottimale. Comunque tutte le macchine devono accettare come minimo un pacchetto di 576 Byte
  - Se non ci riesce deve scartarlo e mandare un messaggio con ICMP
- Poi c'è un bit **MF** (**More Fragment**).
  - Tutti i frammenti escluso l'ultimo hanno questo bit settato. Serve per capire quando sono arrivati tutti i frammenti
- **Fragment Offset**:
  - Serve per capire all'interno del datagram dove va situato il corrente frammento.
  - Tutti i frammenti a parte l'ultimo devono essere multipli di 8 Byte che è l'unità fondamentale di frammentazione. Dal momento che ho 13 bit segue che ho al massimo 8192 frammenti per datagram e un lunghezza totale di 65536 bytes

Esempio di frammentazione

- Datagram di 4000 byte frammentato in 3 frammenti
- Devo numerare da 0 a 3999
- Il primo da 0 a 1399 per cui l'offeset è 0/8 = **0**
- il secondo da 1400 a 2799: l'offset vale 1400/8 = **175**
- il terzo 2800 a 3999: l'offset vale 2800/8 = **350**



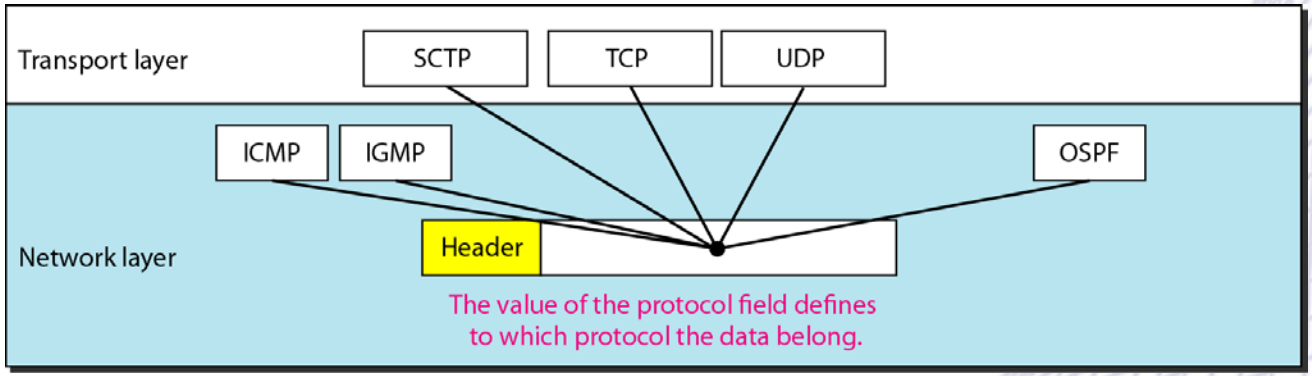
Time to live

- Un contatore che limita la durata di vita del pacchetto.
- Dovrebbe contare in secondi con un massimo di 255 secondi
  - Ma questo richiede che tutti i router abbiano i clock sincronizzati e che siano in grado di calcolare il tempo che il datagram impiega per essere trasferito
  - Invece viene decrementato di uno ad ogni hop e decrementato più volte quando il pacchetto è accodato per lungo tempo in un router
  - In pratica conta gli hops
  - TTL iniziale spesso 64 ma cambia da s.o. di host mittente
- Quando il contatore arriva a zero il pacchetto viene scartato e viene mandato un warning al mittente. Questo accorgimento impedisce ad un datagram di viaggiare all'infinito a causa per esempio di un loop nel routing

Protocol

- Cosa devo fare quando ho riassemblato il pacchetto?
- Devo consegnarlo al livello di trasporto. In questo campo posso specificare che vada a TCP oppure a UDP o ad un altro protocollo definito da IANA e in origine definito nella RFC 1700

Value	*Protocol*
1	ICMP
2	IGMP
6	TCP
17	UDP
89	OSPF



Header Checksum

- Verifica solo l'header
- I dati sono eventualmente verificati dai livelli superiori
- Utile per rivelare errori dovuti a disfunzioni della memoria all'interno del router
- Se tutto è ok il valore è zero
- Questo valore deve essere ricalcolato ad ogni hops dal momento che ogni volta cambia almeno il TTL

Source (Destination) Address

- Indicano il numero di rete e di host del mittente e del destinatario
- Li vedremo in dettaglio in seguito

Campo Options

- Sono state pensate per permettere a future versioni del protocollo di includere informazioni non presenti nell'idea originale
- Per sperimentare nuove idee senza riservare nell'header dei bit che sarebbero stati usati raramente
- Hanno lunghezza variabile e ognuna inizia con un codice identificativo di un 1byte
- Alcune hanno anche un byte di lunghezza e poi uno o più data byte
- All'inizio erano 5 e le vedremo in dettaglio. La lista aggiornata viene mantenuta da IANA

Le 5 Opzioni

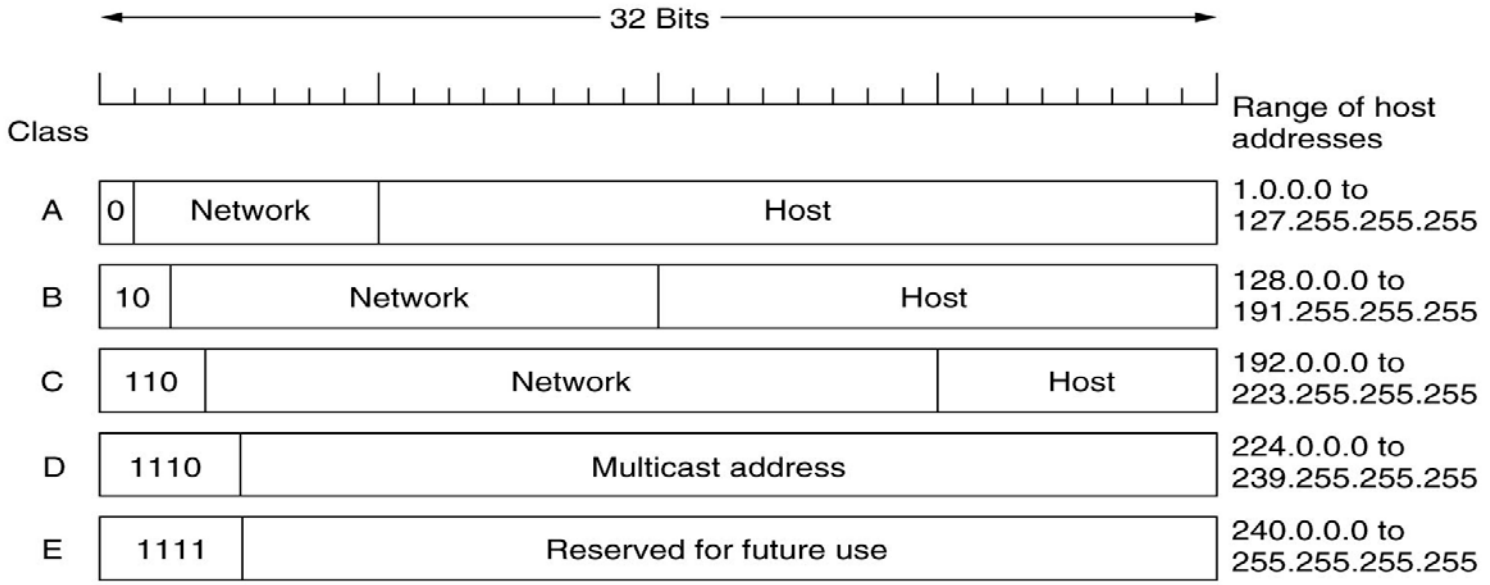
- Security:
  - Dice quanto sono sicure le informazioni:un router potrebbe usarlo per decidere di non instradare per una route insicura. Mai usato
- Strict Source Routing:
  - Specifica la lista di IP address attraverso cui routare. Per fare misure di timing o per forzare pacchetti di emergenza
- Loose Source Routing
  - Specifica la lista di alcuni router da usare e nell'ordine specificato ma non tutti i router. Utile per suggerire una route invece che un'altra
- Record Router
  - Dice ai Router lungo il path di loggare il loro indirizzo per capire se ci sono bug nel routing
- Timestamp
  - Come sopra ma oltre all'indirizzo a 32bit ogni router scrive anche una timestamp a 32 bit

Indirizzi IP

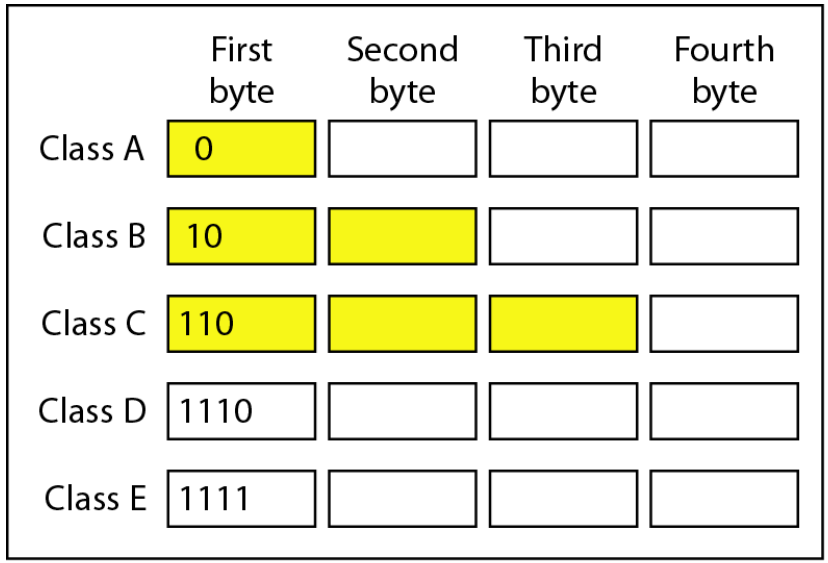
- Ogni host (end node o router) ha un indirizzo IP che codifica il numero di rete e il numero di host
  - Gli indirizzi IP sono in teoria unici e sono lunghi 32 bit
  - 2^{32} = 4.294.967.296 indirizzi diversi
  - I 32 bit si indicano come sequenza di bit di solito raggruppati in byte
  - Es 01110101 10010101 00011101 00000010
  - Ma più spesso ogni byte viene convertito in notazione decimale
  - Es 117.149.29.2
- L'indirizzo IP non è propriamente dell'host
  - ma della sua scheda di rete per cui un router ha un indirizzo IP per ogni porta di rete
  - una porta di un host o di un router ha un indirizzo per ogni rete su cui è presente.

Classful Addressing

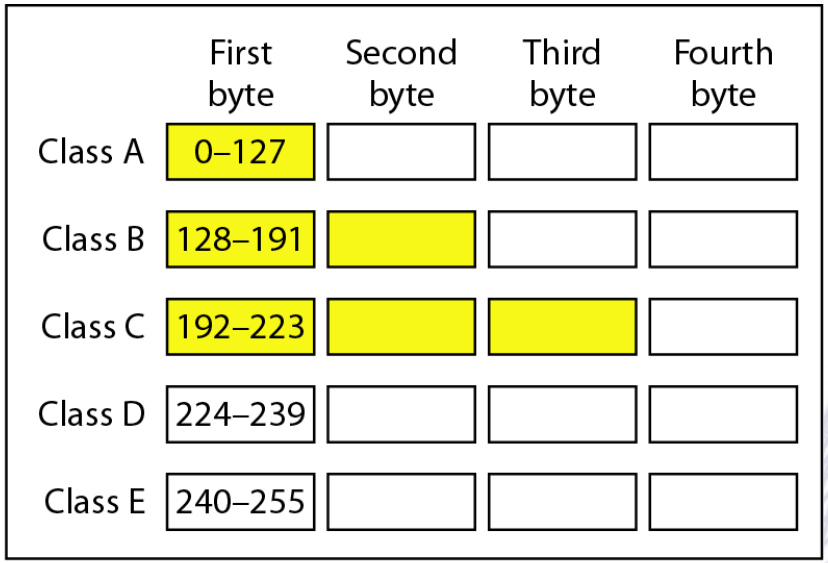
- Classe A: 128 networks con 16M indirizzi ciascuna
- Classe B: 16384 networks con 64K indirizzi ciascuna
- Classe C: 2M networks con 256 indirizzi ciascuna
- Classe D: Indirizzi Multicast (enorme spreco 268 M indirizzi)
- Classe E: Riservata (di nuovo 268M indirizzi)



Classful addressing



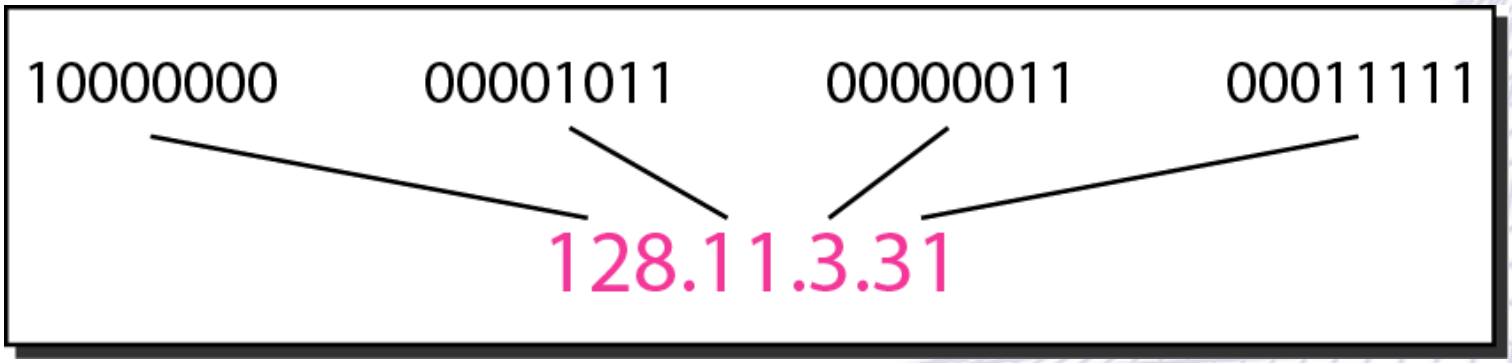
a. Binary notation



b. Dotted-decimal notation

Numerazione

- I numeri sono assegnati da un'associazione no-profit chiamata ICANN per evitare conflitti (due nodi con lo stesso numero)
- ICANN delega poi gerarchicamente l'assegnazione delle reti ad altri enti
- Vediamo di nuovo la notazione "dotted decimal" per cui C0290614 viene scritto 192.41.6.20
- Altro esempio in figura 800B031F diventa 128.11.3.31



Indirizzi di rete e di host

- Con l'indirizzamento a classi un indirizzo di classe A,B o C viene diviso in un **indirizzo di rete** e un **indirizzo di host**
- Classe A: 7 bit di indirizzi per la rete (il primo identifica la classe), 24 bit per gli host
- Classe B: 14 per la rete (i primi due per la classe) e 16 per gli host
- Classe C: 21 bit per la rete (primi 3 per la classe) e 8 bit per gli host
- Il concetto di host e rete non esiste per le classi D e E

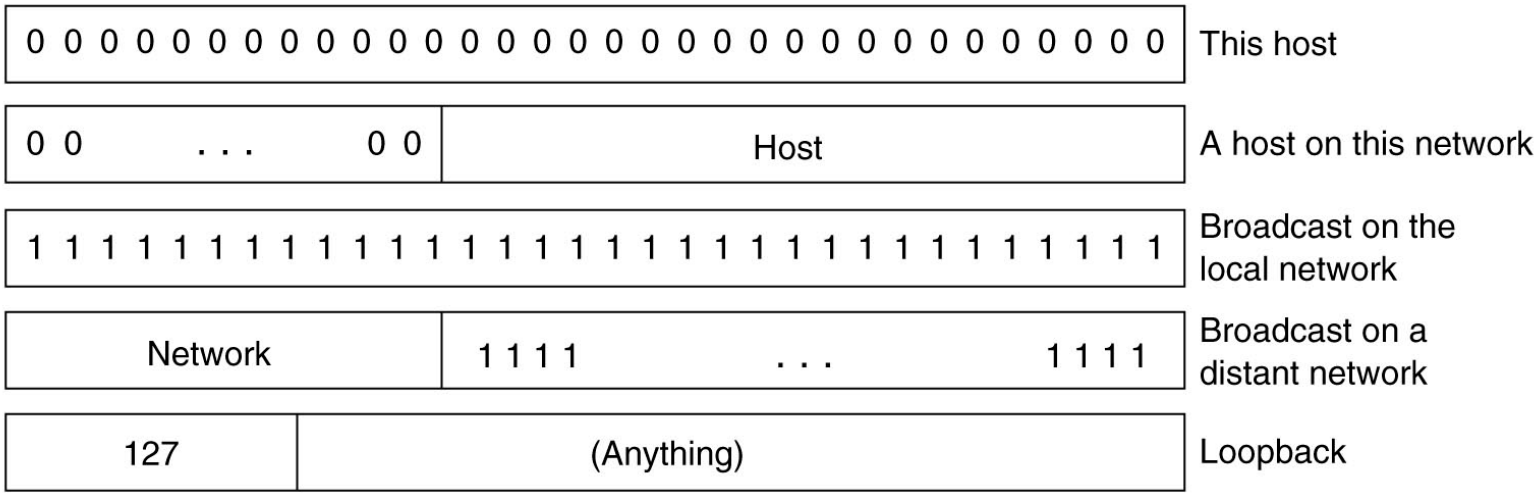
maschere

- Per estrarre rapidamente da un indirizzo IP l'indirizzo di rete o quello di host si usano delle maschere di bit
- Per avere l'indirizzo di rete faccio un AND (bit a bit) tra l'indirizzo e la sua maschera
- Per avere l'indirizzo di host faccio un AND tra l'indirizzo e il complemento della maschera

Class	*Binary*	*Dotted-Decimal*	*CIDR*
A	**11111111** 00000000 00000000 00000000	**255**.0.0.0	/8
B	**11111111 11111111** 00000000 00000000	**255.255**.0.0	/16
C	**11111111 11111111 11111111** 00000000	**255.255.255**.0	/24

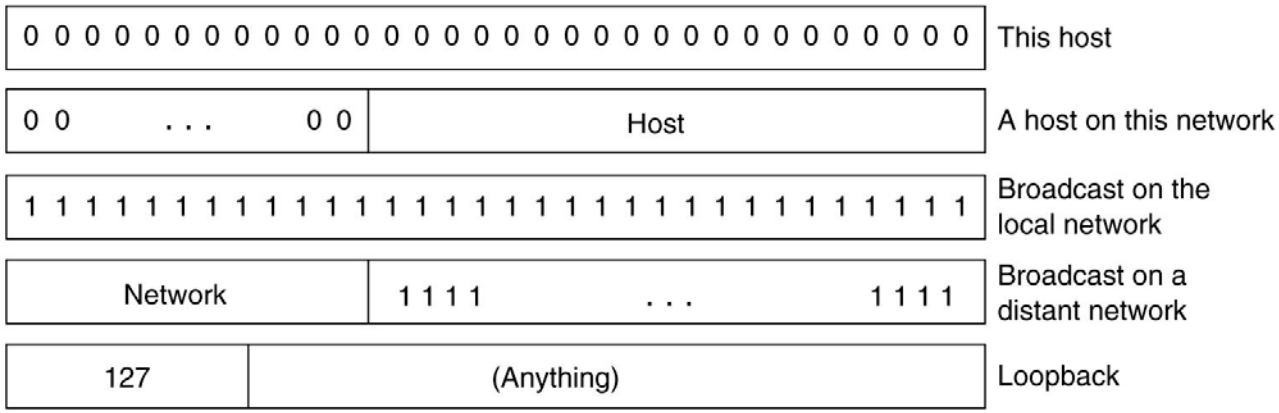
Indirizzi speciali

- L'indirizzo 0.0.0.0 viene usato dagli host nella fase di boot
- Indirizzi con la parte network a zero indicano che si vuole indirizzare una macchina nella stessa rete
- Indirizzo con tutti i bit a 1 → 255.255.255.255 significa "tutti i nodi", un broadcast nella LAN a livello network



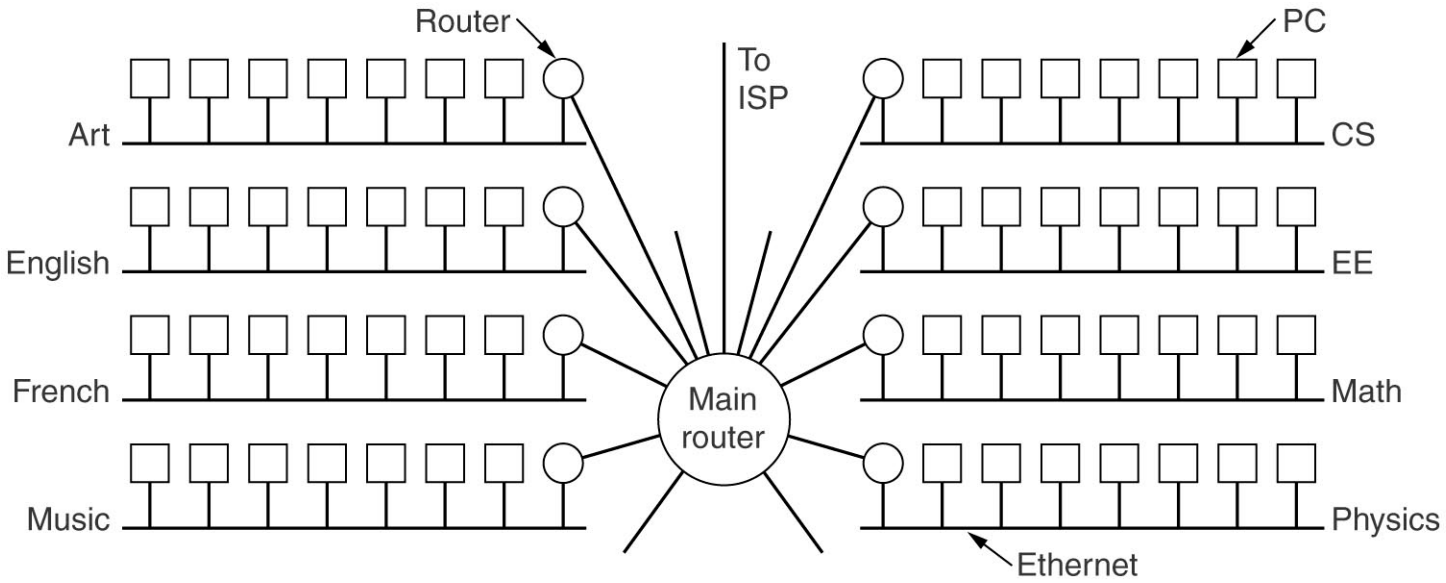
Indirizzi speciali

- Indirizzi con una parte di rete normale e solo la parte di host a 1 significa un broadcast su quella rete remota (di solito l'amministratore del router blocca questi pacchetti)
- Indirizzi che cominciano con 127 sono per il test del loopback. I pacchetti inviati a questo indirizzo non vengono messi sul cavo ma sono processati localmente e trattati come pacchetti in ingresso. In questo modo posso mandare pacchetti nel protocollo di LAN senza sapere il mio indirizzo



Subnetting

- Come abbiamo visto tutti gli host di una rete hanno lo stesso indirizzo di network
- Si potrebbe voler separare una rete in parti diverse per uso interno ma apparendo sempre come un'unica rete verso l'esterno
- Per esempio ho un router di bordo verso l'esterno di un campus ma poi ho diverse LAN ognuna con il suo router verso l'interno e ho quindi bisogno di routing tra le diverse LAN

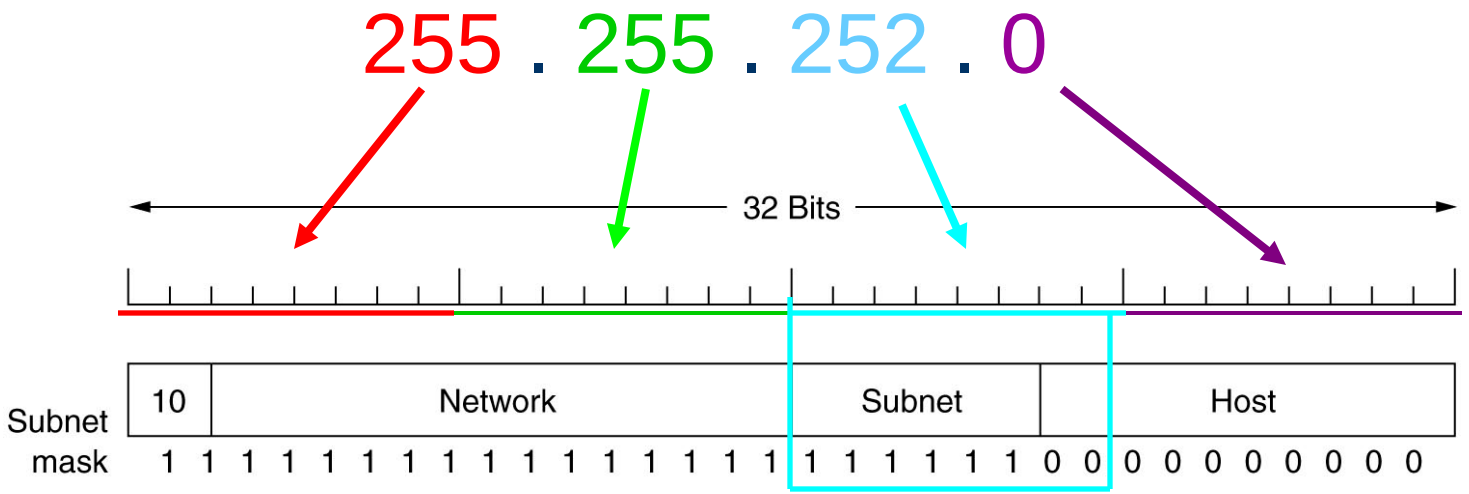


subnet

- In Internet queste parti si chiamano subnet
  - Noi chiamavamo subnet l'insieme dei router e linee di comunicazione in una rete. Attenzione a non confondere i due usi della parola
- Quando arriva un pacchetto ad una rete B come fa a sapere a chi mandarlo?
  - Ha una tabella con 65536 entries che indica quale router usare per ogni host?
  - No! invece di usare 14 bit per la rete e 16 per gli host uso alcuni numeri della parte host per creare una sottorete. Per esempio se ho 35 dipartimenti uso 6 bit per le sottoreti, ognuna con 10 bit per gli host

Subnet mask

- Per fare questo uso una subnet mask che scrivo o in notazione dotted decimal con uno slash che indica i bit della parte di rete + sottorete oppure in esadecimale
  - Esempio 255.255.252.0 oppure /22 per indicare che sto usando 22 bit di subnet mask
  - Come vedete abbiamo una classe B con $2^6=64$ subnet da 1024



Nessun conflitto

- All'esterno del main router questo non si vede quindi non devo dirlo all'ICANN
- La prima subnet potrebbe usare l'indirzzo 130.50.4.1, la seconda 130.50.8.1, poi 130.50.12.1, di quattro in quattro come si vede contando in binario
- Subnet1 10000010 . 00110010 . **000001**|00 . 00000001
- Subnet2 10000010 . 00110010 . **000010**|00 . 00000001
- Subnet3 10000010 . 00110010 . **000011**|00 . 00000001
- Ogni router ha una tabella con entries indirizzi IP tipo (network,0) e altri del tipo (questa-rete, host)
- I primi si riferiscono a reti lontane mentre i secondi dicono come raggiungere i nodi locali. Ogni tabella è associata all'interfaccia da usare per raggiungere quella rete

Default gateway

- Quando arriva un pacchetto IP si cerca il suo destination address nella tabella.
  - Se è in una rete distante il pacchetto viene forwardato al prossimo router usando l'interfaccia indicata nella tabella
  - Se è un nodo locale viene mandato direttamente a destinazione
  - Se non c'è la parte di rete viene mandato ad un default router
  - In questo modo ogni router deve vedere solo altre reti oppure host locali non tutte le coppie (network,host) quindi posso avere routing table piccole
  - Quando inserisco subnetting compaiono entries del tipo (questa-rete, subnet, 0) e (questa-rete, questa-subnet, 0) per cui un router della sottorete **k** sa come raggiungere anche le altre subnet e anche gli host nella propria sottorete **k** ma non deve conoscere i dettagli delle altre sottoreti diverse da **k**

Esempio di subnet

- In pratica il router fa un AND Boolean con la subnet mask mettendo a 1 tutti i bit della parte host che invece fanno parte della subnet e cerca questo indirizzo nelle sue tabelle
- Es arriva pacchetto con indirizzo 130.50.**15**.6, lo metto in AND con 255.255.**252**.0/22 e ottengo 130.50.**12**.0 e capisco che devo andare nella **terza** subnet

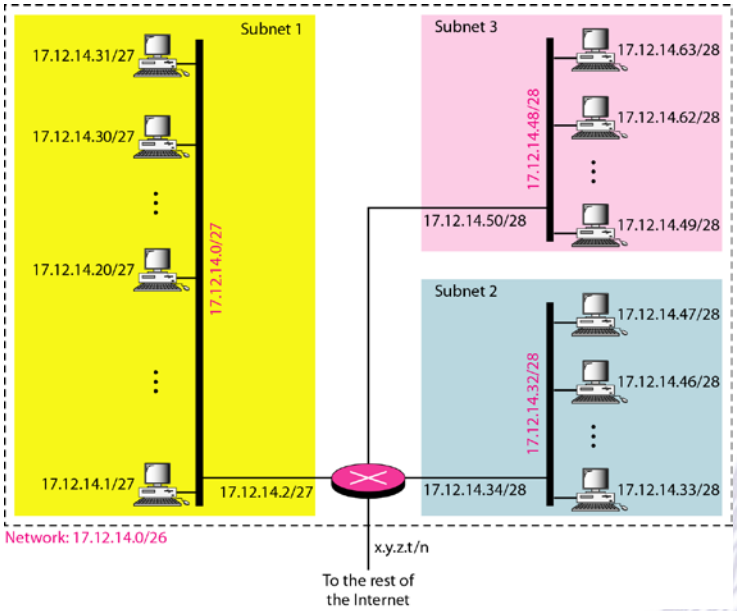
015	→	00001111	AND
252	→	11111100	=
012	→	00001100	

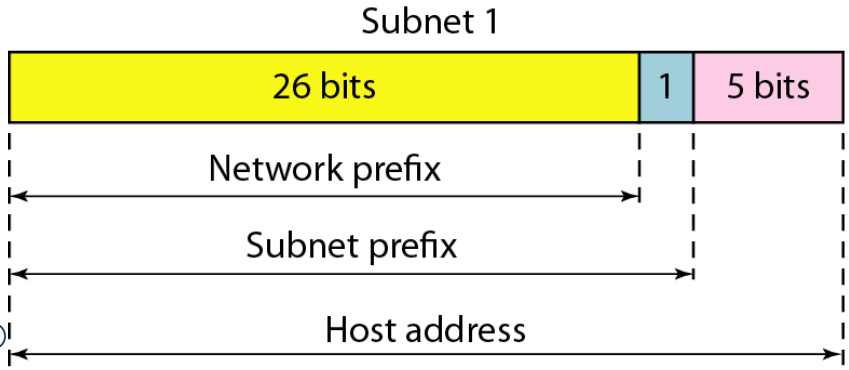
Esempio di subnet

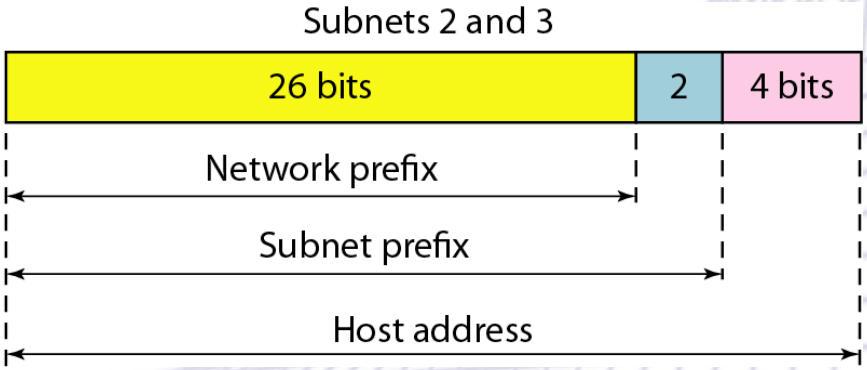
- In questo modo riduco le dimensioni della parte locale della tabella
- Mi sposto da una visione “questa-rete” e “216 = 65536 nodi” quindi una gerarchia a due livelli di 16+16
- ad una gerarchia a 3 livelli di rete-sottorete-nodo 16+6+10

Altro esempio

- avevo una rete di 26bit e 64 host in una gerarchia a due livelli
- Ora ho 3 subnet, una con 32 e due con 16 host con una gerarchia a 3 livelli, net, subnet, host







Carenza di indirizzi

- La separazione in classi è molto rigida.
  - È stata pensata all'inizio di internet quando aveva senso pensare che non ci sarebbero mai stati più di 16000 università attaccate a internet. Nessuno pensava a nodi internet in ogni casa
  - 256 nodi per una classe C sono pochi se non per piccole organizzazioni
  - Le classi B invece sono troppo grandi ma d'altra parte anche splittare (in modo meno intuitivo) a 20 bit invece che a 16 avrebbe portato ad una esplosione del numero delle classi B

Carenza di indirizzi

- Se ci fosse mezzo milione di classi C i router dovrebbero avere tabelle di queste dimensioni perché i router non vogliono sapere nulla degli host remoti ma vogliono conoscere le reti remote.
- Tabelle di queste dimensioni si corromperebbero molto facilmente anche pensando al fatto che gli algoritmi dinamici richiedono ai router di trasmettere periodicamente le tabelle

CIDR

- Classless InterDomain Routing (RFC 1519)
- Permette di dare a Internet un po’ di respiro per quanto riguarda la carenza di indirizzi allocando indirizzi senza badare alle classi
- Se un sito ha bisogno di 2000 indirizzi gli diamo un blocco di 2048 indirizzi
- Questo rende il forwarding più complicato

Cosa cambia?

- Come era il routing?
  - Con il vecchio sistema a classi quando un pacchetto arriva leggo la classe poi ho un ***if*** a 16 vie che mi mette i pacchetti in A,B,C e D con 8 vie per la A, 4 per la B e due ciascuna per C e D. Il codice di ogni ***if*** maschera gli 8- o 16- o 24-bit della parte di rete e poi cerca nelle tabelle A B e C la linea di uscita e il pacchetto viene mandato su quella linea
  - Magari essendo A e B più piccole sono tabelle indicizzate per indirizzo di A e B mentre per la C usa una ricerca ad hashing

Esempio di routing table

	Indirizzo	Maschera
Ca.	11000010 00011000 **00000000** 00000000	11111111 11111111 11111**000 00000000**
Ed.	11000010 00011000 **00001000** 00000000	11111111 11111111 11111**00 00000000**
Ox.	11000010 00011000 **00010000** 00000000	11111111 11111111 1111**0000 00000000**

University	First address	Last address	How many	Written as
Cambridge	194.24.0.0	194.24.7. 255	2048	194.24.0.0/21
Edinburgh	194.24.8.0	194.24.11.255	1024	194.24.8.0/22
(Available)	194.24.12.0	194.24.15.255	1024	194.24.12/22
Oxford	194.24.16.0	194.24.31.255	4096	194.24.16.0/20

Routing associato

- Arriva un pacchetto per 194.24.17.4 → 11000010.00011000.00010001.00000010
- Lo metto in AND con la maschera di Cambridge e ottengo 11000010.00011000.00010**000.00000000** che non matcha con l'indirizzo base di Cambridge per cui
- lo metto in AND con la mask di Edinburgh e ottengo 11000010.00011000.000100**00.00000000** che non matcha con l'indirizzo di Edinburgh
- per cui provo con la mask di Oxford e ottengo 11000010.00011000.0001**0000.00000000** che matcha l'indirizzo base di Oxford. Se non ne trovo altre uso la entry di Oxford e mando il pacchetto verso la linea di Oxford

Punto di vista esterno

- Come vedo questi tre siti dal punto di vista di un router in Omaha, Nebraska, USA che ha solo 4 linee di uscita verso Minneapolis, New York, Dallas e Denver?
- Quando al router di Omaha arrivano queste 3 entries vede che le può combinare in una unica entry: **194.24.0.0/19**
- Questa entry manda tutti i pacchetti per le 3 università a New York. NB il router di Omaha ha ridotto la sua routing table di due entries (da 3 a 1)

Punto di vista esterno

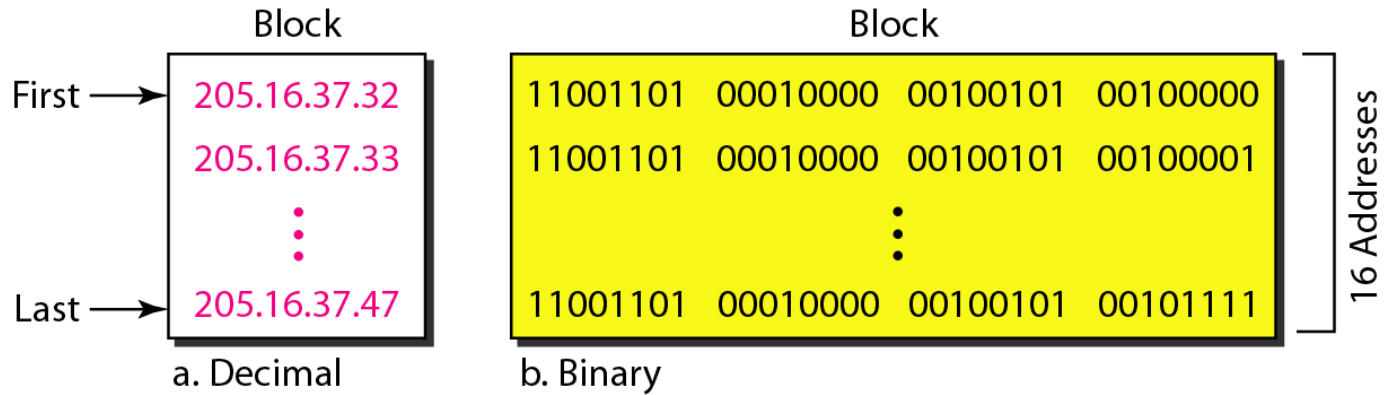
- Se New York ha un'unica linea verso Londra per tutto il traffico UK anche il router di New York può ridurre la sua routing table
- Se invece ha due linee separate una per Londra e una Oxford allora deve tenere le entries separate
- Questa tecnica di riduzione delle routing table viene usata pesantemente per cui si danno numeri di rete contigui a siti che sono topologicamente vicini
- NB il router di Omaha manda anche i pacchetti della rete non assegnata verso New York. Nessun problema se la rete è veramente non assegnata, ma se in futuro dovesse essere assegnata ad una ditta in California, dovremmo aggiungere una entry 194.24.12.0/22 per gestire questo caso

Come cambia il routing

- Com’è ora con CIDR?
  - Ora c’è una unica routing table che consiste in un array di (indirizzo IP, subnet mask, linea di uscita)
  - Quando il pacchetto arriva cerco il suo indirizzo IP
  - La routing table viene cercata brutalmente per trovare un match. Se ne possono trovare diverse con diverse lunghezze di subnet mask in questo caso viene presa la subnet più lunga. Es se trovo un match per una maschera /20 e una /24 prendo la /24
  - Ci sono diverse algoritmi per velocizzare questo scan. I router in commercio hanno appositi chip VLSI o ASIC per eseguire questi algoritmi in hardware

Esercizio

- Siamo una piccola organizzazione, ci hanno dato una rete 205.16.37.39/28
- Quanti indirizzi sono? Qual è il primo? E l'ultimo?
  - /28 quindi, visto che 32-28 = 4 ho 2^4 = 16 indirizzi
  - Il primo lo trovo mettendo a **0** i 4 bit a dx della parte host dell'indirizzo dato 205.16.37.0010**0111** quindi 205.16.37.0010**0000** = 205.16.37.32
  - L'ultimo lo trovo mettendoli tutti a **1** quindi 205.16.37.0010**1111** = 205.16.37.47



continua

- Oppure uso le maschere
  - La maschera di una /28 è fatta di 28 volte bit 1 e 32-28 volte bit 0
  - 11111111 11111111 11111111 11110000
  - Come trovo il primo indirizzo? AND tra mask e l'indirizzo del blocco!
  - E l'ultimo? OR tra l'indirizzo del blocco e il complemento della mask

Address:	11001101	00010000	00100101	00100111
Mask:	**11111111**	**11111111**	**11111111**	**11110000**
First address:	11001101	00010000	00100101	00100000

Address:	11001101	00010000	00100101	00100111
Mask complement:	**00000000**	**00000000**	**00000000**	**00001111**
Last address:	11001101	00010000	00100101	00101111

Network address

- Il concetto di **indirizzo di rete** è molto importante
- Quando una organizzazione ha un blocco di indirizzi lo usa come vuole
- Il primo indirizzo tuttavia viene normalmente (ma non sempre) trattato come un indirizzo speciale per identificare l'organizzazione all'interno di internet

Limited broadcast address

- Anche l'ultimo indirizzo con tutti i bit della parte host a 1 ha un significato particolare, infatti i messaggi inviati a questo indirizzo vengono mandati dal router a tutti i nodi della rete x.y.z.w/
- Si chiama indirizzo di Limited Broadcast perché è un broadcast limitato alla rete

32 bit sono pochi

- Abbiamo visto che Internet ha avuto un successo che i progettisti non si aspettavano
- La scarsità di indirizzi è in parte dovuta all'indirizzamento a classi che come abbiamo visto si risolve con CIDR
- Tuttavia la diffusione di internet da ambienti universitari alle grandi imprese e poi alle piccole imprese e ai privati ha portato a richieste di miliardi di indirizzi
- La prima risposta degli ISP è stata quella di dare indirizzi dinamici allocati solo per il tempo della connessione.
- Anche questa risposta si rivela insufficiente
  - Piccole aziende vogliono essere sempre online
  - Privati che usano ADSL o linee modem con tariffe flat, possono stare quasi sempre online

Moltiplicazione degli indirizzi

- La soluzione consiste nell'aumentare lo spazio degli indirizzi.
- Vedremo IPv6 che permette indirizzi a 128 bit, miliardi di indirizzi per ogni essere umano
- IPv6 da anni cerca di sostituirsi a IPv4 ma la transizione è molto dolorosa
- La soluzione temporanea e poco elegante si chiama **N**etwork **A**ddress **T**ranslation (**NAT**)

Network Address Translation

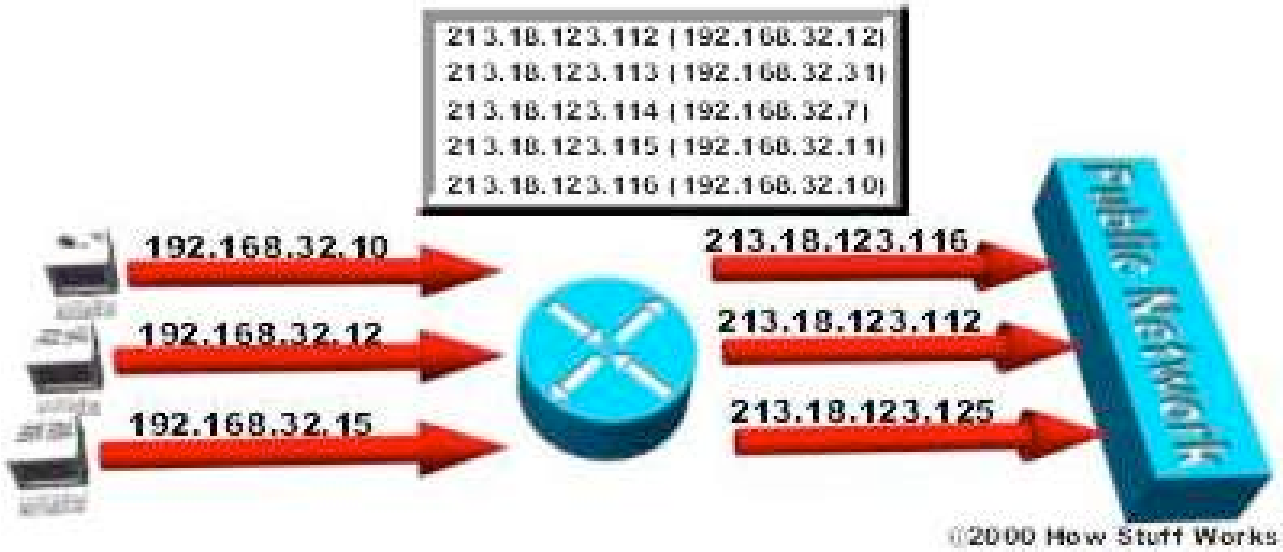
- Viene descritta nella RFC 3022
- Ad una ente o privato viene assegnato un numero un unico numero IP oppure un range molto piccolo di numeri per il traffico Internet
- All'interno dell'ente ogni computer riceve un numero IP univoco necessario per il **routing all'interno**.
- Quando un pacchetto deve uscire e andare verso l'ISP l'indirizzo viene "tradotto" al volo

Diversi tipi di NAT

- Static NAT
  - Mappa un indirizzo IP privato ad un indirizzo IP pubblico
  - Utile quando un dispositivo deve essere accessibile dall'esterno della rete
  - il computer con indirizzo 192.168.32.10 si traduce sempre con **213.18.123.110:**

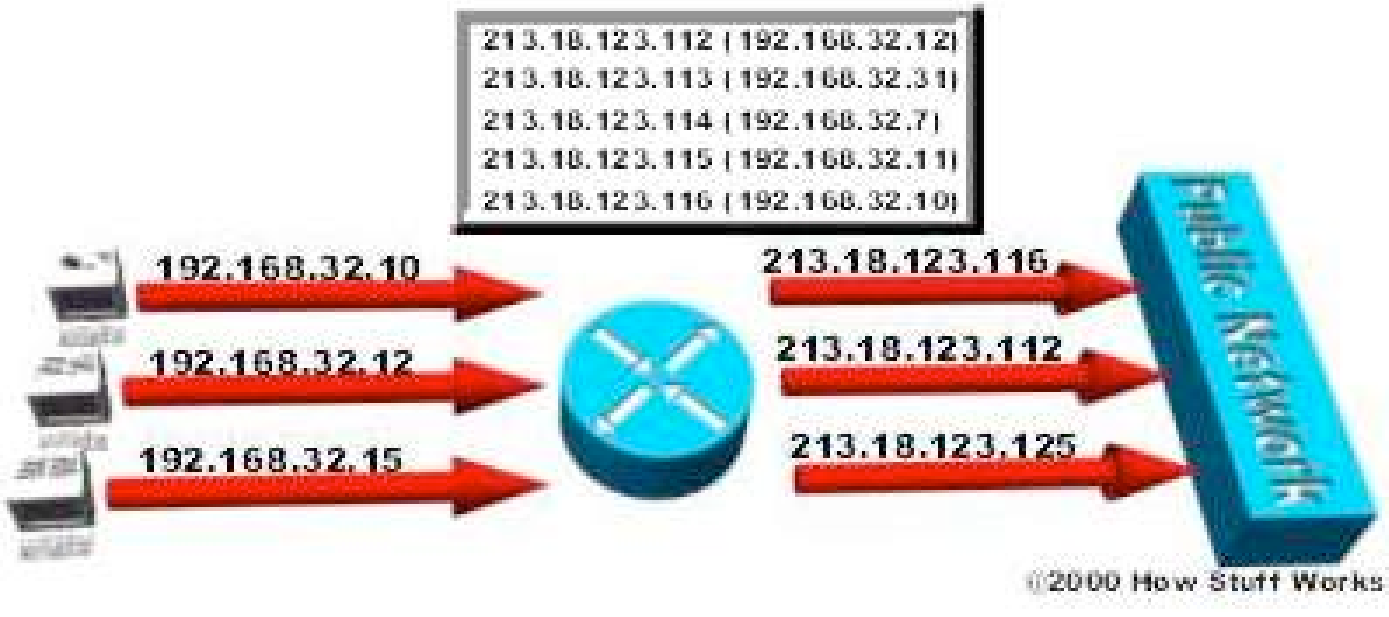
Diversi tipi di NAT

- Dynamic NAT
  - Mappa un indirizzo IP privato su di un indirizzo IP pubblico preso da una gruppo di indirizzi.
  - Dynamic NAT stabilisce una mappatura one-to-one ma la mappatura dipende da quali siano gli indirizzi del pool siano disponibili
  - Nel dynamic NAT, il computer con 192.168.32.10 si traduce nel primo disponibile nel pool tra 213.18.123.100 e 213.18.123.150



Overloading NAT

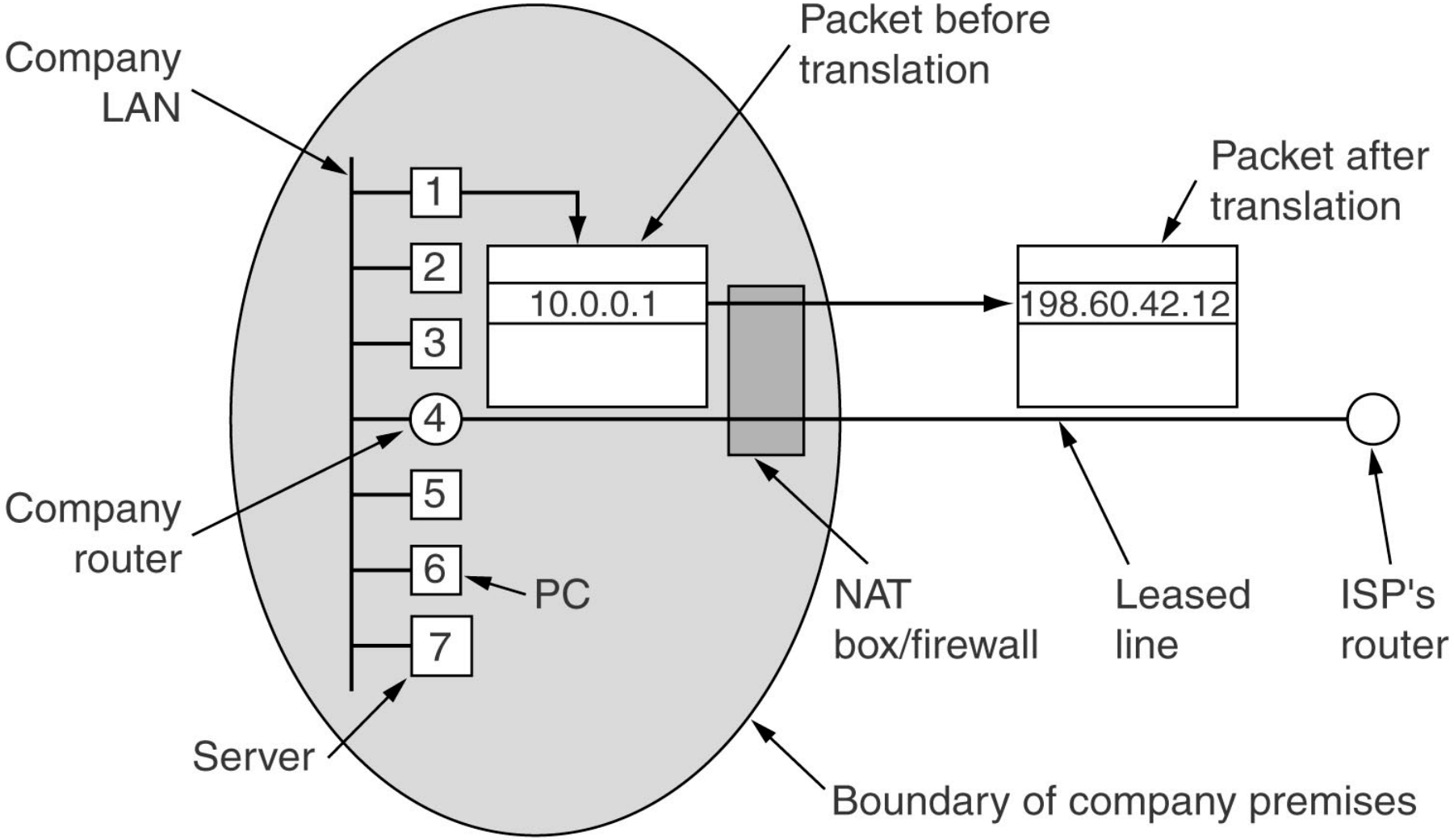
- Overloading NAT detto PAT
  - Una versione di NAT dynamic in cui diversi nodi interni sono mappati ad unico indirizzo pubblico usando le porte
  - Noto come PAT (Port Address Translation), single address NAT o anche port-level multiplexed NAT.
  - Quello che ci serve per rimediare al problema della scarsità di indirizzi



Indirizzi privati

- Questo è possibile dichiarando tre ranges di indirizzi come privati
  - Nel senso che possono essere usati liberamente all'interno degli enti stessi, nelle LAN, ma non possono essere annunciati su **Internet**
  - 10.0.0.0 10.255.255.255/8 (16.777.216 nodi)
  - 172.16.0.0 172.31.255.255/12 (2.048.576 nodi)
  - 192.168.0.0 192.168.255.255/16 (65.536 nodi)
  - Quando il pacchetto esce dall'ente, passa attraverso una NAT box che converte l'indirizzo interno es. 10.0.0.0 nel vero indirizzo IP pubblico 198.60.42.12

NAT box


Company LAN
Packet before translation
Packet after translation
1
2
3
4
5
6
7
10.0.0.1
198.60.42.12
Company router
PC
NAT box/firewall
Leased line
ISP's router
Server
Boundary of company premises

Come risponde?

- Il pacchetto 10.0.0.1 viene tradotto nell'indirizzo pubblico
- Quando il pacchetto di reply torna come fa la NAT box a mandarlo all'indirizzo interno giusto?
- **Questo è il problema del NAT**
- Non c'è un campo libero nell'header di IP per tenere traccia del vero mittente. C'è solo un unico bit libero!

Uso TCP/UDP

- I progettisti del NAT hanno visto che la maggior parte del traffico trasporta all'interno pacchetti TCP o UDP
- Vedremo che questi protocolli usano nei loro header un numero di porta mittente e porta destinazione.
- Serve per indirizzare i processi all'interno della macchina.
- Vediamo l'esempio TCP ma UDP si comporta in modo analogo

Uso delle porte

- Le porte sono interi a 16 bit
- Quando un processo vuole comunicare con un processo remoto usa una porta libera nella propria macchina (SOURCE port) e specifica anche una porta al lato remoto (DESTINATION port)
- Le porte 1-1023 sono di solito riservate per servizi noti (well known services)
- Analogia con le extension per un centralino telefonico

Traduzione indirizzo porta

- Quando il pacchetto destinato all'esterno entra nel NAT, l'indirizzo 10.x.y.z viene rimpiazzato da un indirizzo pubblico dell'ente
- **Anche la source port viene rimpiazzata** da una entry nella tabella di traduzione con 65536 entry del NAT
  - La tabella punta all'indirizzo e alla porta originali
- IP e checksum vengono ricalcolati e scritti nel nuovo pacchetto tradotto

…e al ritorno

- Quando un pacchetto arriva da internet, la **porta Destination** viene estratta e presa come un indice dentro la tabella della NAT box
- L'indirizzo e la porta originari vengono estratti e sostituiti, rifatti gli header e i checksum e il pacchetto viene mandato al host con l'indirizzo privato originario (es 10.x.y.z) alla porta originaria

Abominio

Questo schema, molto usato, risolve il problema ma viene visto come un “abominio” dai puristi IP:

- Prima di tutto viene violato il modello architetturale di IP che dice che ogni indirizzo IP identifica univocamente una macchina. Con NAT migliaia di macchine usano per es 10.0.0.1
- Il NAT cambia IP da rete connectionless a rete connection oriented dal momento che ogni NAT box deve tenere informazioni per ogni connessione apera attraverso di essa.
- Se il NAT crasha, la tabella si perde e tutte le connessioni distrutte
- Senza NAT se un router crasha non ci sono effetti sul TCP, il processo mittente aspetta qualche secondo e ritrasmette tutti i pacchetti che non sono stati acknowledged

Abominio (cont)

- Il NAT viola la regola più fondamentale del layering, il layer k non deve basarsi su cosa il layer k+1 ha messo nel payload. Deve essere indipendente da esso. Invece se un futuro TCP-2 avesse un **header** diverso (es 32 bit /porta) NAT smetterebbe di funzionare. **NAT distrugge l'indipendenza dei layers**
- I processi su internet non devono per forza usare TCP o UDP. Se un utente vuole usare un nuovo protocollo che non usa numeri di porta, non passerebbe il NAT

Abominio (cont.)

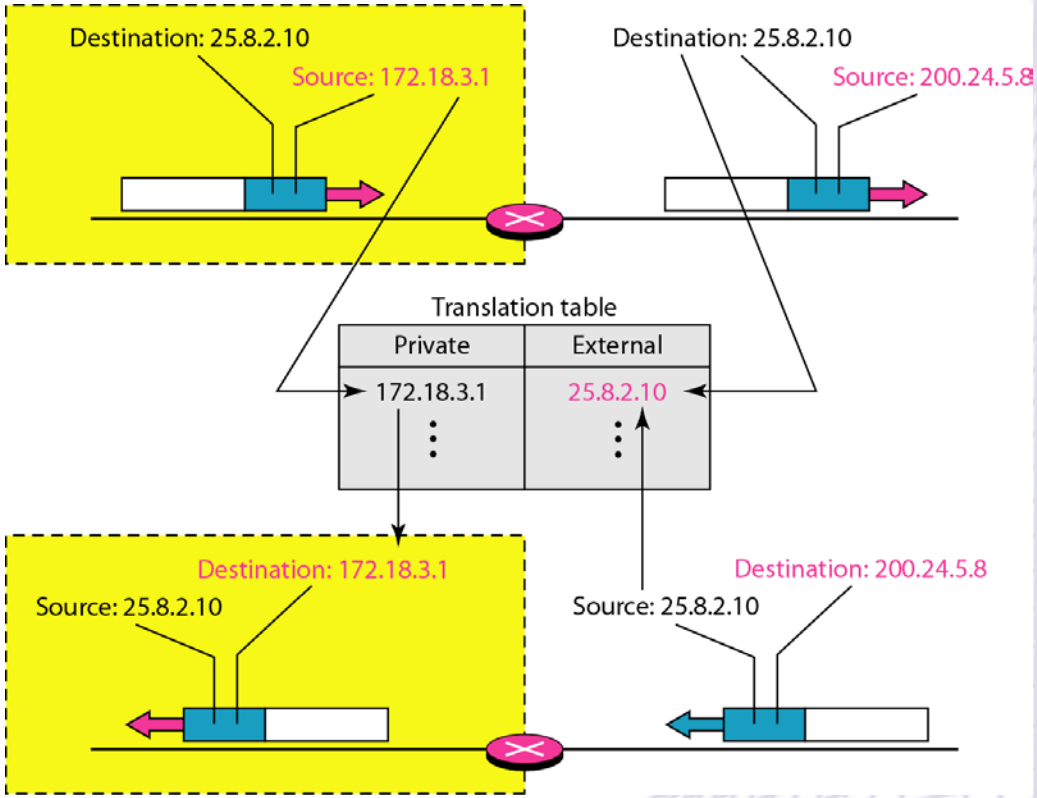
- Alcune applicazioni inseriscono il numero IP dentro il body del testo. Il ricevente estrae queste informazioni e le usa. Il NAT non sa nulla di questi indirizzi, non li traduce, quindi ogni tentativi di usarli remotamente fallisce.
  - Es FTP (File Transfer Protocol) fallisce in presenza di NAT a meno di non prendere precauzioni
  - H.323, il protocollo per telefonia su IP ha problemi e non funziona bene dietro un NAT. Bisognerebbe patchare il NAT per ognuno di questi protocolli.

Altri problemi…

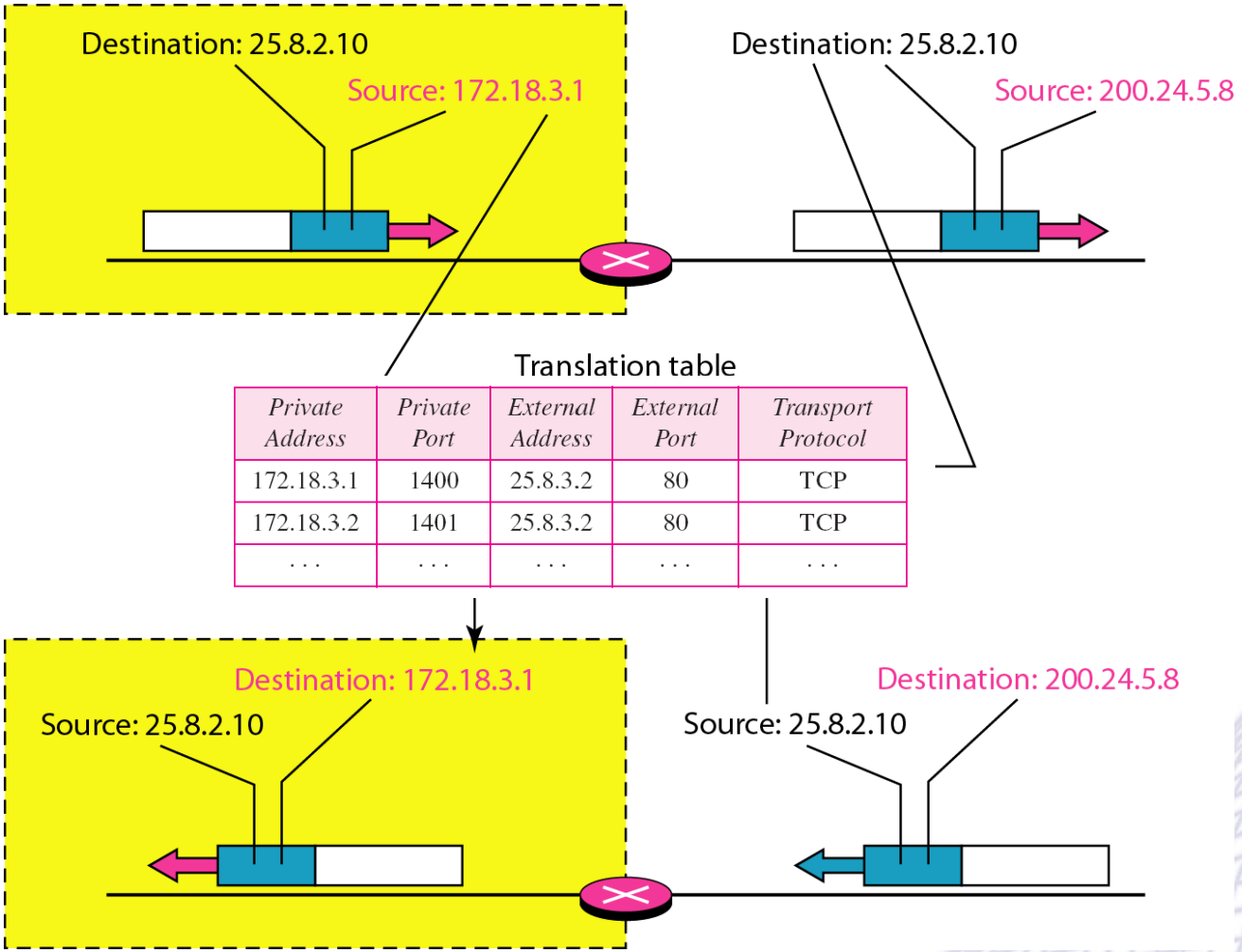
- Il campo TCP ha 16 bit, e quindi permette di mappare 65536 macchine in un indirizzo IP
  - Alcune (4096) di queste porte sono riservate per usi speciali
  - Il numero di porte disponibili e quindi di indirizzi NATtabili è al massimo 61440.
- Tutti questi problemi sono discussi nella RFC 2993
  - Gli oppositori al NAT dicono che risolvere il problema della mancanza di indirizzi in questo modo orribile deve essere temporaneo e che la soluzione deve venire da IPv6
  - Purtroppo questa soluzione ha di fatto ridotto la spinta ad una rapida transizione verso IPv6

NAT senza porte

- Gli indirizzi privati servono solo all'interno di una organizzazione.
- Se solo poche macchine avessero bisogno della connessione esterna e avessi un numero equivalente di indirizzi pubblici non sono costretto ad usare le porte
- Quindi associo in una tabella N indirizzi interni a N indirizzi esterni e traduco solo il numero



Con le porte


Destination: 25.8.2.10
Source: 172.18.3.1
Destination: 25.8.2.10
Source: 200.24.5.8
Translation table
Private Address | Private Port | External Address | External Port | Transport Protocol
172.18.3.1 | 1400 | 25.8.3.2 | 80 | TCP
172.18.3.2 | 1401 | 25.8.3.2 | 80 | TCP
. . . | . . . | . . . | . . . | . . .
Destination: 172.18.3.1
Source: 25.8.2.10
Source: 25.8.2.10
Destination: 200.24.5.8

Oltre IPv4

- Il problema della scarsità di indirizzi IP univoci è stato alleviato ma non risolto da CIDR e NAT
  - Servono quindi spazi di indirizzamento più vasti
- Ci sono anche altri problemi
  - Internet nasceva nelle Università, aziende Hi-Tech, Siti governativi.
  - Ora viene usato da gente più eterogenea
    - Gente che usa portatili wireless per tenersi in contatto con la sede
    - D'altra parte si assiste ad una convergenza tra computer con telefoni e televisioni per cui saranno necessari miliardi di numeri IP
- Nel 1990 IETF ha cominciato a lavorare su di una nuova versione di IP

IP Next: Requirements

- Supporto per miliardi di host, anche con allocazioni inefficiente dello spazio di indirizzamento
- Ridurre le dimensioni delle routing tables
- Semplificare i protocolli così che i router processino i pacchetti più velocemente
- Migliorare la security (autenticazione e privacy)
- Prestare più attenzione ai tipi di servizio (dati real time)
- Migliorare il multicasting
- Rendere possibile per un host di fare roaming senza cambiare indirizzo
- Permettere al protocollo di evolvere in futuro
- Permettere al nuovo protocollo di coesistere con il vecchio per tanti anni

Proposte

- IETF fece una RFC 1550 per la quale arrivarono 21 risposte, non tutte complete
  - A Dicembre 1992 rimasero 7 risposte serie in competizione, da alcune che erano minime pezze a IP ad altre molto radicali
  - Una proponeva di mettere TCP su CLNP che aveva indirizzi a 160 bit e avrebbe permesso di unificare i due maggiori protocolli di livello rete
  - A molti questo sembrava un'ammissione che il mondo OSI aveva fatto qualcosa di buono, affermazione poco Politically Correct nella cerchia Internet
  - CLNP era in realtà costruito simile a IP, quindi non molto differente. Infatti il protocollo scelto alla fine assomiglia più a CLNP che a IPv4
  - Inoltre mancava anche di supporto per i tipi di servizio, richiesto per trasmettere efficacemente traffico multimediale

La scelta

- Nel 1993 tre proposte (“Deering”, “Francis” e “Katz and Ford” vennero pubblicate su IEEE Network
- Dopo diverse discussioni venne scelto un misto di Deering e Francis chiamato **SIPP** (**Simple Internet Protocol Plus**) che divenne quindi **IPv6**

IPv6

- IPv6 mantiene le migliori features di IP, scarta o dissinnesca quelle peggiori e ne aggiunge di nuove
- Risponde bene a tutti i requirements
- Non è in generale compatibile con IPv4 ma è compatibile con gli altri protocolli (TCP, UDP, ICMP, OSPF, BGP, DNS) a meno di minimi cambiamenti
- Si vedano le RFC da 2460 a 2466

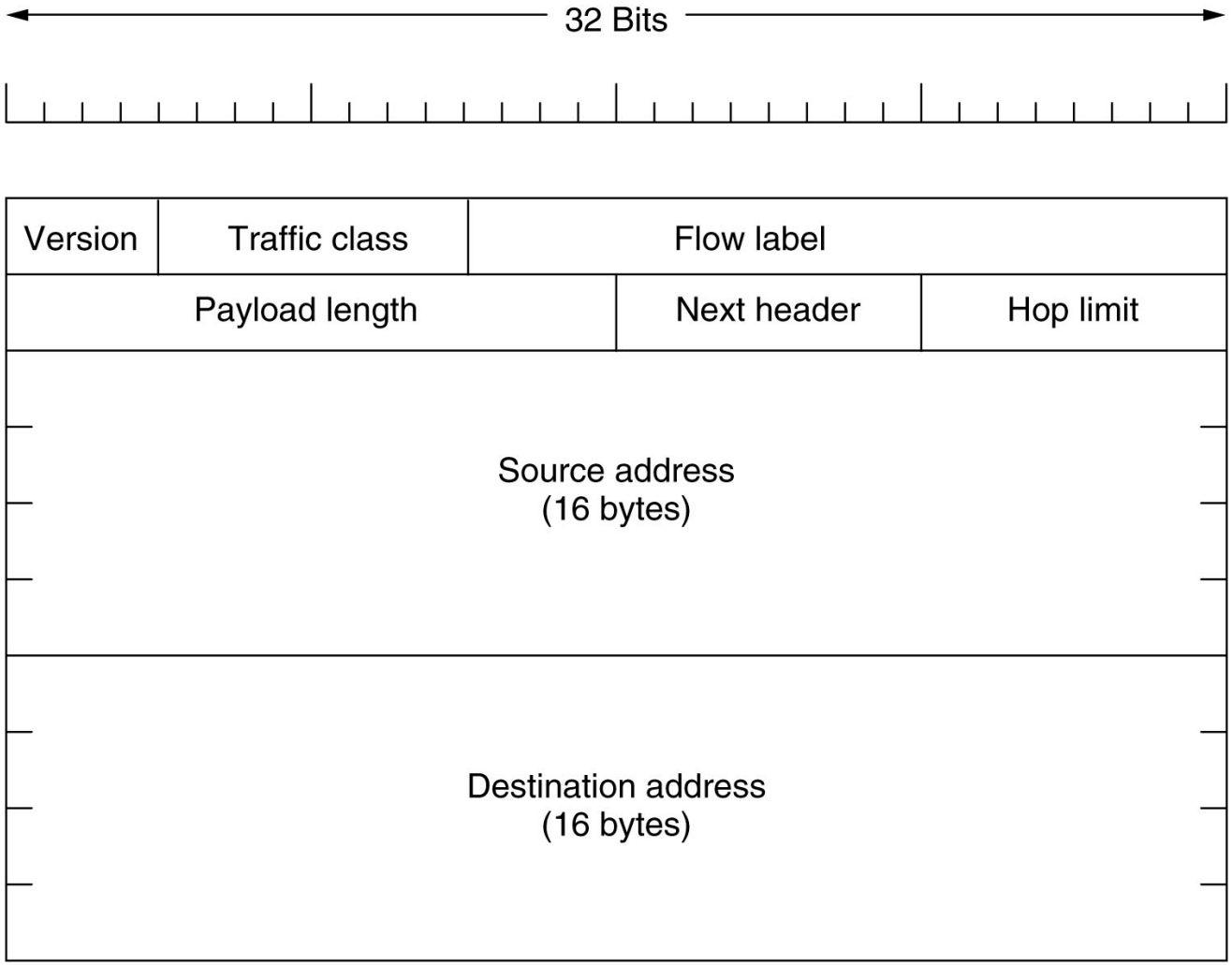
Miglioramenti

- Spazio di indirizzamento enorme.
  - Ci sono 16 bytes. Riserva illimitata di indirizzi internet
- Header semplificato
  - Contiene solo 7 campi rispetto ai 13 di IPv4. Miglior processing dei pacchetti migliorando bandwidth e delay
- Supporto per le option
  - Molti campi prima richiesti ora sono opzionali
  - Inoltre molte opzioni sono rappresentate in modo diverso, permettendo ai router di saltare le opzioni che non sono per loro. Migliora il processing dei pacchetti

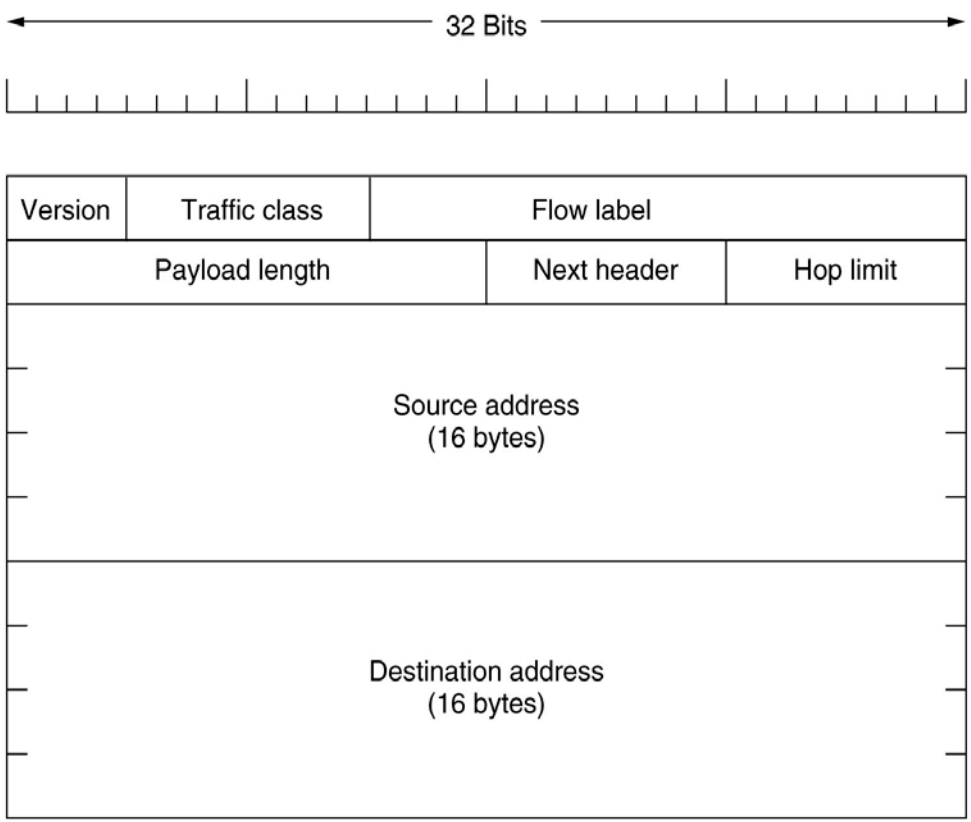
Miglioramenti

- Migliore Security
  - Autenticazione e Security che poi sono stati messi anche in IPv4 per cui ora non si vedono grandi differenze
- Quality of Service

Header IPv6



Version



- Il campo Version vale sempre 6 (e 4 per IPv4)
  - Nel periodo di transizione un router potrà esaminare questo campo per distinguere i tue tipi di pacchetti
  - Questo potrebbe rallentare il processing e spingere alcune implementazioni ad usare un campo nel **header data link** per passare il pacchetto al network layer corretto. Questo tuttavia viola il principio di progettazione secondo cui un layer non devono sapere il significato dei bit del livello superiore
  - Discussioni tra sostenitori di “make it right” e “make it fast”

Traffic Class

- Traffic Class
  - Distingue pacchetti con diversi requirements real time. C’era qualcosa di simile anche in IPv4 ma era poco usato dai router
  - Ora si sta sperimentando come usarlo soprattutto per consegna di contenuti multimediali

Flow label

- Flow Label
  - Sperimentale. Permette a source e destination di costruire una pseudo-connessione con particolari requirements
  - Es. Un certo flusso di pacchetti potrebbe aver bisogno di un certo delay e quindi avere banda riservata. Posso quindi creare un flow e dargli un numero. Se al router arriva un pacchetto con Flow non zero, può cercare in una sua tabella interna che trattamento speciale deve riservare
  - In questo modo ho sia una rete a datagram che una specie di di circuito virtuale
  - Ogni rotta è definita da indirizzo sorgente, destinazione e numero di flow per cui posso avere diversi flussi attivi allo stesso momento tra una coppia di indirizzi IP

Payload length

- Dice quanti bytes seguono i 40 byte dell'header
  - In IPv4 si chiamava Total Lenght e comprendeva anche l'header

Next Header

- Il motivo per cui l'header si è potuto semplificare è che ci possono essere header di estensioni addizionali
  - Questo campo dice quali degli attuali sei header di estensione segue questo header
  - Se questo header è l'ultimo il campo viene usato per indicare a quale gestore di protocollo di trasporto (TCP, UDP) passare il pacchetto

Hop limit

- Limita la vita del pacchetto
  - Come il Time to Live di IPv4
  - Viene decrementato di uno ad ogni hop
  - In teoria IPv4 era il tempo in secondi ma nessun router lo usa in quel modo per cui è stato cambiato in modo da rispecchiare l'uso effettivo

Indirizzi

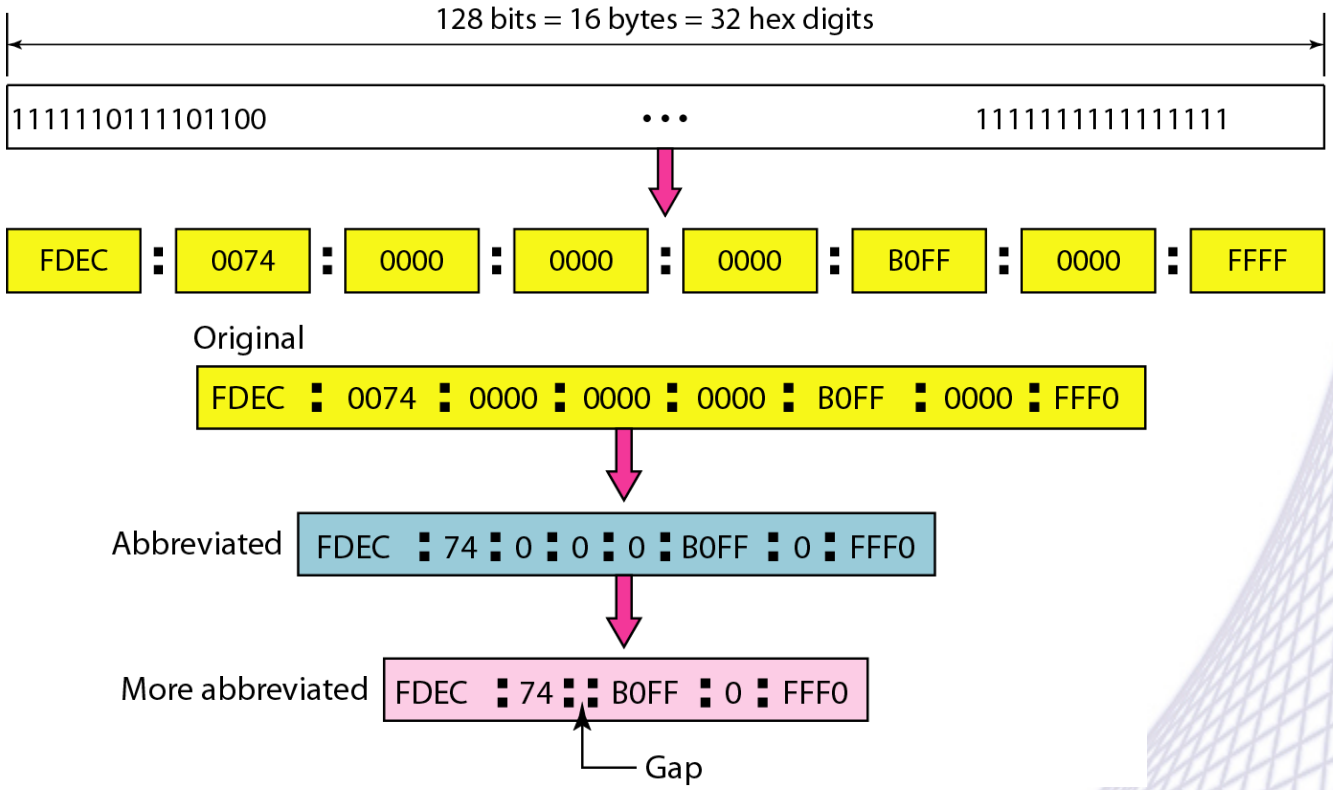
- I campi successivi sono Source Address e Destination Address
  - La proposta di Deering era di 8 byte ma nella discussione si pensò che in poche decine di anni si sarebbero esauriti di nuovo tutti gli indirizzi per cui si decise per 16 byte
  - Ad alcuni sembrò un numero esagerato ma altri volevano addirittura 20 per esseri compatibili con OSI
  - Altri volevano indirizzi di dimensione variabile

Notazione

- Gli indirizzi vengono scritti come 8 gruppi di quattro cifre esadecimali separati da “:”
  - Es: 8000:0000:0000:0000:0123:4567:89AB:CDEF
- Molti indirizzi contengono zeri. Per semplificare
  - Gli zero iniziali di un gruppo possono essere omessi per cui 0123 diventa 123
  - Gruppi di 16 bit zero possono essere omessi e sostituiti da “:”
  - Es precedente: 8000::::123:4567:89AB:CDEF
  - Gli indirizzi IPv4 si possono scrivere ::192.31.20.46

Notaz. hex (e abbreviata)

- Indirizzi IPv6 in notazione binaria e esadecimale
- In diverse notazioni abbreviate



Quanti sono?

- Con 16 byte ho 128 bit
- Quindi ho 2^{128} indirizzi, che corrisponde a $3 \cdot 10^{38}$
- Ho 2^{96} indirizzi in più rispetto a IPv4 !
- Se tutta la terra e le acque fossero coperta di computer avrei $7 \cdot 10^{23}$ indirizzi IP per ogni metro quadro, più del numero di Avogadro !
- In pratica lo spazio di indirizzamento non viene usato in modo efficace (come anche lo spazio telefonico, non tutti i prefissi di area sono riempiti in modo uguale)
- In gran parte sono usati per diversi scopi o sono riservati, a noi interessano quelli di tipo Unicast, assegnati dall'internet provider (sono 1/8 del totale e iniziano con 010

Prefissi IPv6

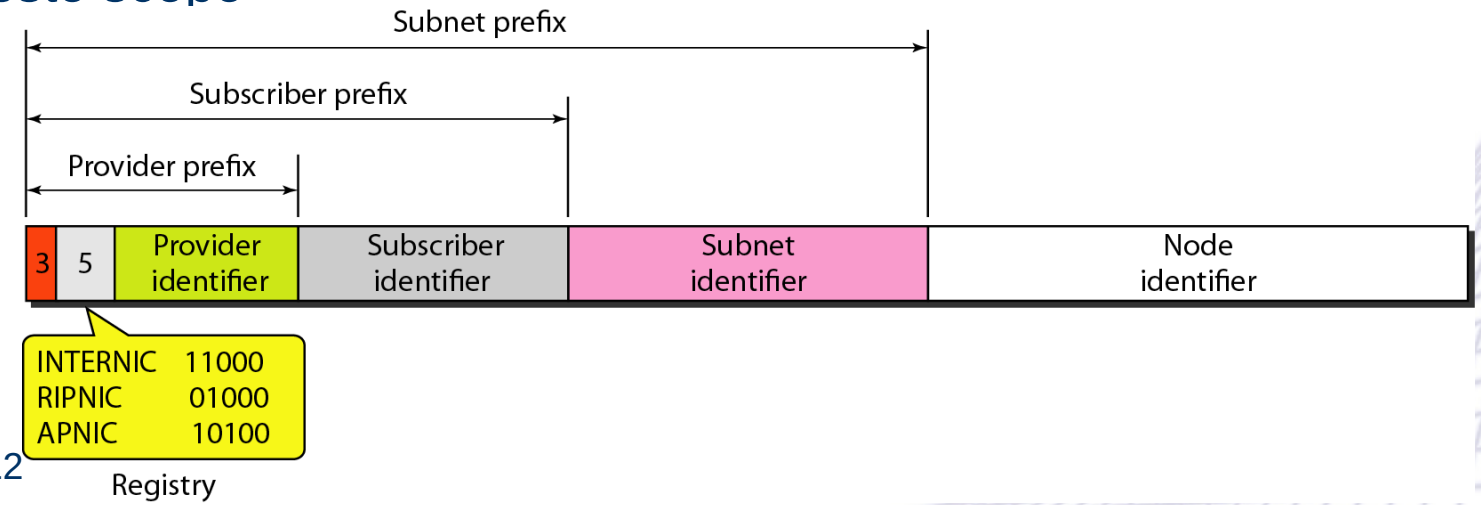
Type Prefix	*Type*	*Fraction*
0000 0000	Reserved	1/256
0000 0001	Unassigned	1/256
0000 001	ISO network addresses	1/128
0000 010	IPX (Novell) network addresses	1/128
0000 011	Unassigned	1/128
0000 1	Unassigned	1/32
0001	Reserved	1/16
001	Reserved	1/8
010	**Provider-based unicast addresses**	**1/8**

Prefissi IPv6

Type Prefix	*Type*	*Fraction*
011	Unassigned	1/8
100	Geographic-based unicast addresses	1/8
101	Unassigned	1/8
110	Unassigned	1/8
1110	Unassigned	1/16
1111 0	Unassigned	1/32
1111 10	Unassigned	1/64
1111 110	Unassigned	1/128
1111 1110 0	Unassigned	1/512
1111 1110 10	Link local addresses	1/1024
1111 1110 11	Site local addresses	1/1024
1111 1111	Multicast addresses	1/256

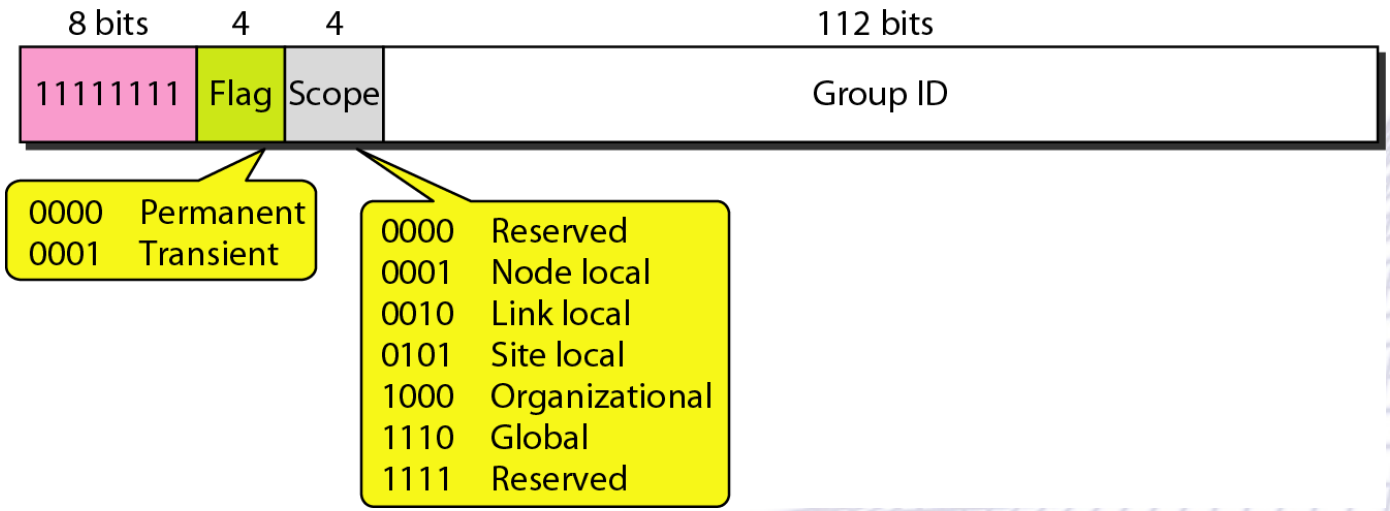
Formato IPv6 unicast

- Dopo **010** vengono **5** bit che identificano l'agenzia che ha registrato gli indirizzi (internic per Nord America, RipNic per l'Europa e ApNic per Asia e Pacifico
- **16** bit per identificativo dell'ISP
- **24** bit identificato dell'utente che ha sottroscritto il contratto di accesso con il provider
- Ogni utente ha il diritto di usare diverse sottoreti per le quali ha **32** bit
- Infine **48** bit per il nodo. Si raccomanda 48 bit per essere compatibile con l'indirizzo di rete, anzi in futuro potrebbe essere davvero usato per questo scopo



Multicast

- Individua un gruppo di host nella rete
- I pacchetti devono essere indirizzati a tutti i nodi nel gruppo identificato dall'indirizzo
- Un indirizzo permanente è sempre attivo, uno temporaneo viene attivato per esempio per partecipare ad una videconferenza per la durata della conferenza
- Il terzo è lo scope (la visibilità)

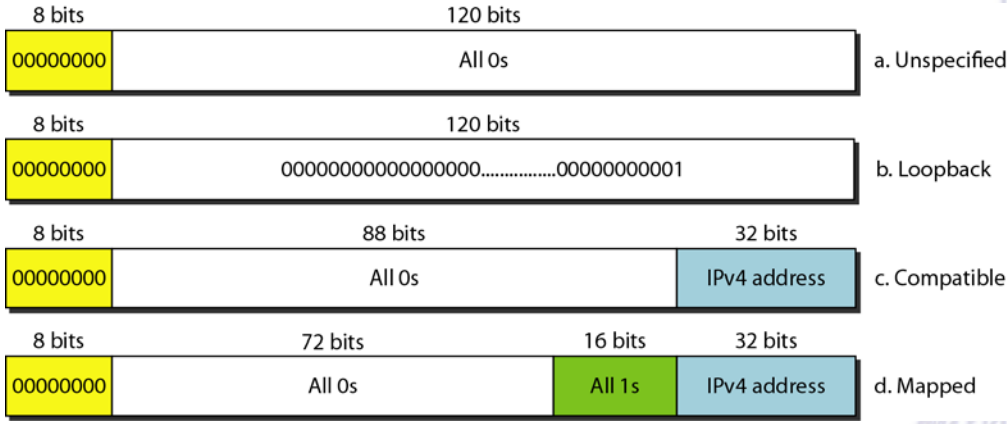


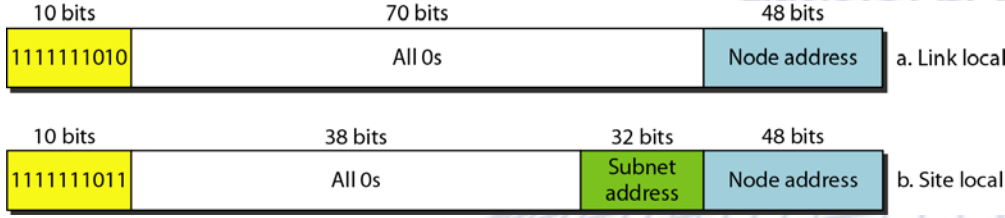
Anycast

- Identifica un gruppo di nodi. Un pacchetto indirizzato ad un indirizzo anycast deve essere indirizzato ad uno qualsiasi dei nodi, tipicamente il più raggiungibile
- Possibile utilizzo: per dare a tutti i router di un ISP un indirizzo di anycast in modo che un router esterno riesca a raggiungere almeno uno

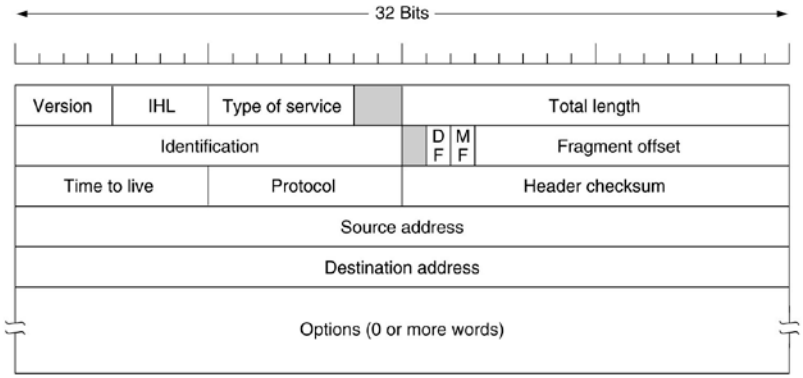
Indirizzi riservati

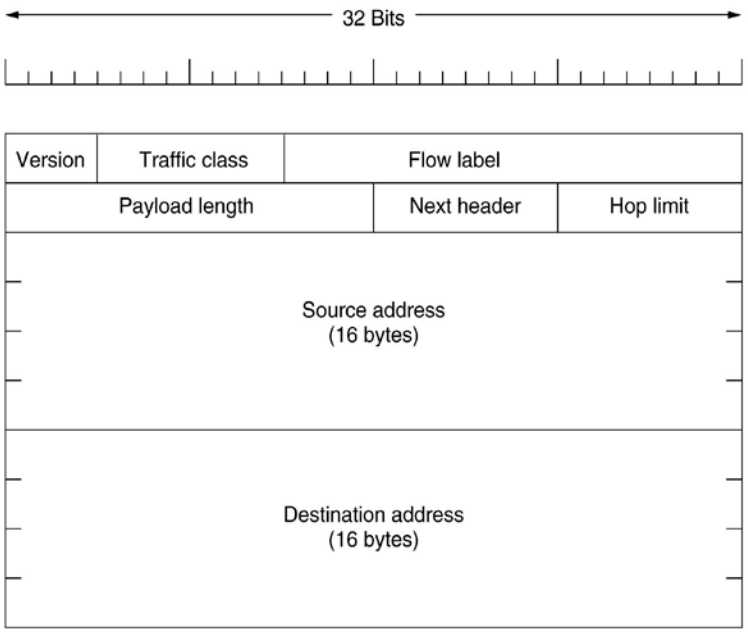
- **Unspecified**: quando un nodo non conosce il proprio indirizzo e deve mandare pacchetti per scoprirlo
- **Loopback**: i pacchetti percorrono lo stack ma non vanno sulla rete e vengono consegnati al nodo mittente
- **Compatible**: da utilizzare nella fase di transizione IPv4 – IPv6 per mandare messaggi da IPv6 a IPv6 via router IPv4
- **Mapped**: per transizione da usare quando un nodo IPv6 vuole parlare con un nodo IPv4
- Indirizzi locali di **sottorete** o di **rete** per indirizzi privati da usare all'interno dell'organizzazione





Cosa resta fuori?





- Il campo IHL è rimasto fuori perché l'heaser IPv6 ha lunghezza fissa
- Il campo Protocol non serve perché il campo Next Header dice cosa viene dopo l'ultimo header IP (un segmento TCP o UDP)
- Tutti i campi relativi alla frammentazione
- Inoltre il checksum non viene fatto per non ridurre le prestazioni. Con le reti affidabili in uso e il fatto che di solito il livello data link e trasporto hanno i loro checksum non vale la pena calcolarne un altro

Frammentazione

- La frammentazione viene gestita in modo diverso
  - I nodi IPv6 devono essere in grado di deteminare dinamicamente le dimensioni di datagram da usare
  - Questo rende più difficile che succeda la frammentazione
  - Il minimo è stato portato da 576 a 1280 byte permettendo 1024 byte di dati e diversi header
  - Inoltre se un host manda un pacchetto IPv6 troppo grande, invece di frammentarlo il router che non riesce a forwardarlo lo rimanda indietro con un messaggio di errore
  - Il fatto che gli host mandino i pacchetti di dimensione giusta è in ultima analisi più efficiente che se i router devono frammentarli al volo.

Estensioni

- Alcuni dei campi mancanti di IPv4 a volte sono necessari. IPv6 introduce il concetto di header di estensione
- Questi header forniscono informazioni extra codificate in maniera efficiente
- Ci sono sei tipi di estensione, ognuno opzionale ma se più di uno è presente devono apparire dopo l'header fisso e nell'ordine in cui sono listati

Sei tipi

Extension header	Description
Hop-by-hop options	Miscellaneous information for routers
Destination options	Additional information for the destination
Routing	Loose list of routers to visit
Fragmentation	Management of datagram fragments
Authentication	Verification of the sender's identity
Encrypted security payload	Information about the encrypted contents

- Alcuni hanno un formato fisso, altri hanno un numero variabile di campi di lunghezza variabile
- Per questi ogni entry viene codificato come una tupla (Type, Length, Value)

Type, Length, Value

- Type è lungo un byte e indica di quale opzione si tratta.
  - I valori di Type sono stati scelti in modo che i primi due bit dicono al router cosa fare con le opzioni che non sanno trattare:
    - Salta l'opzione
    - Scarta il pacchetto
    - Scarta il pacchetto e manda indietro un pacchetto ICMP
    - Scarta il pacchetto e manda indietro un pacchetto esclusi gli indirizzi multicast
- Length è lungo un byte dice quanto è lungo il Value (da 0 a 255)
- Value contiene eventuali informazioni richieste fino appunto a 255 byte

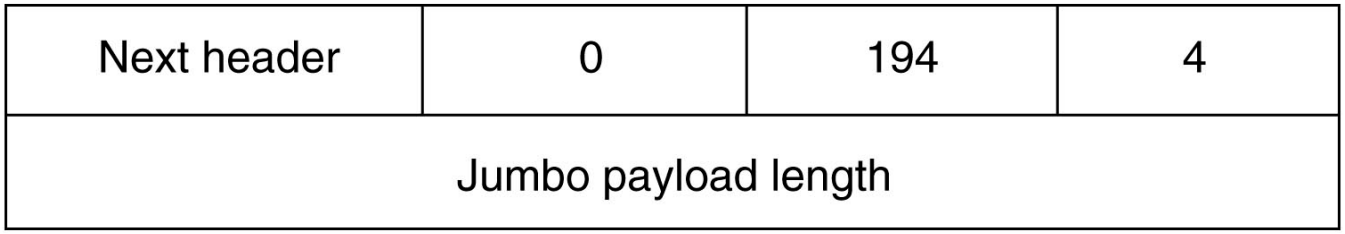
Hop by Hop

- L'header Hop-by-hop viene usato per informazioni che tutti i router nel path devono esaminare
  - Al momento è stata definita un'opzione: supporto di datagrams che eccedono i 64 KB
  - Quando viene usato il Payload length nell'header fisso viene settato a zero
  - Come tutti gli extension ho un byte che mi dice il tipo del prossimo header, poi un byte che dice quanto è lungo questo header esclusi i primi 8 bytes (0)

Next header	0	194	4
Jumbo payload length			

Jumbogram

- Poi ho due byte, uno dice che sto specificando le dimensioni del datagram (codice 194) e poi che le dimensioni sono un numero di 4 bytes.
- Gli ultimi 4 bytes sono le dimensioni del datagram
- Deve essere un numero superiore a 65536, altrimenti il primo router scarta il pacchetto e restituisce un ICMP
- Datagram che usano questa estensione si chiamano jumbograms, e sono usati per trasferire grosse quantità di dati, soprattutto tra supercalcolatori

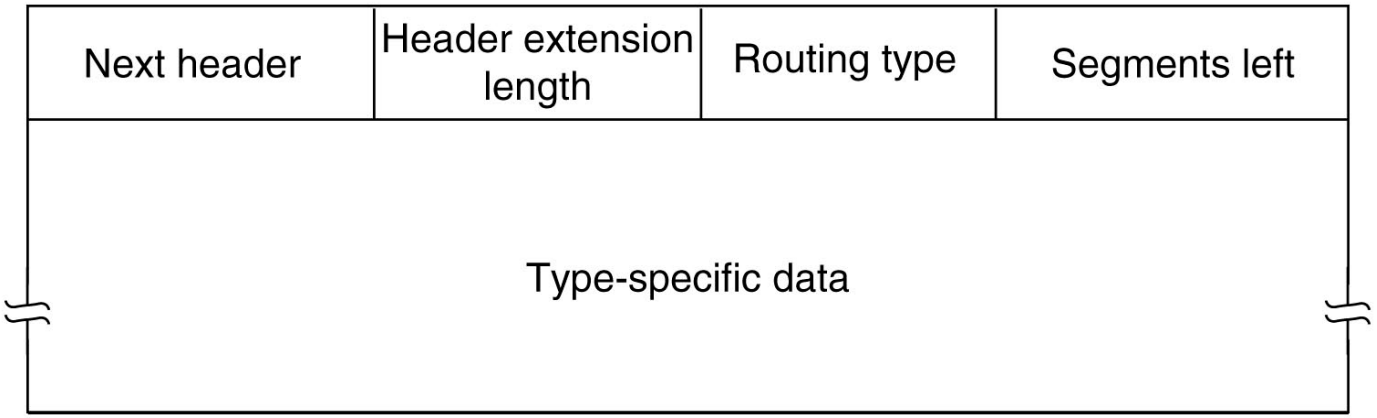


Destination Host

- Destination Host Header
  - Per campi che devono essere interpretati solo nell'host di destinazione
  - Non viene usata per il momento ma i router e gli host la devono implementare nel caso a qualcuno venga in mente un uso sensato nel futuro

Routing Header

- Contiene la lista dei router che devono essere visitati tra source e destination
  - Simile al loose source routing di IPv4
- I primi 4 bytes contengono 4 campi da un byte
  - **Next header** e **Header Extension Lenght** li abbiamo visti prima
- **Routing type** da il formato del resto dell'header
  - Type 0 indica che una word di 32 byte segue la prima word ed è seguita da alcuni numeri di indirizzi IPv6
- **Segment left** tiene traccia di quanti indirizzi nella lista non sono ancora stati visitati
  - Quando arriva a 0 il pacchetto non ha più nessuna guida e deve trovare la route da solo. Di solito a questo punto siamo vicini alla destinazione e la best route è ovvia



Fragmentation

- Fragmentation header
  - Gestisce la frammentazione come IPv4
  - Nell'header ci sono gli identificatori del datagram, il numero di frammento e un bit che indica se altri frammenti seguono
  - In IPv6 al contario di IPv4 solo l'host mittente può frammentare un pacchetto, i router lungo la via non possono farlo
  - Questo è un cambiamento filosofico rispetto al passato ma semplifica la vita ai router e rende il routing più veloce. Se un router non riesce a mandare un pacchetto manda indietro un ICMP di errore e il mittente frammenta il pacchetto usando questo header e riprova

Authentication Encryption

- Authentication header
  - Fornisce un meccanismo per cui il ricevente può essere sicuro dell'identità del mittente
- Encryption header
  - Encrypted security payload permette di criptare il contenuto di un pacchetto in modo che solo il destinatario legittimo possa leggerlo
- Entrambi questi header usano teniche crittografiche per svolgere questo task

Controversie

- Molte scelte sono state molto controverse
  - Abbiamo già visto la discussione che ha portato all'indirizzo fisso a 16 byte
- Il campo Hop Limit
  - Qualuno sentiva 255 (8 bit) come troppo limitato, visto che già oggi si usano path a 32 hop e voleva 16 bit.
  - Ma la funzione di Hop Limit è quella di impedire al pacchetto di vagabondare troppo e 65636 sarebbe troppo. Se Internet cresce, devono crescere anche i link a lunga distanza e deve essere possibile andare ovunque in una dozzina di hops al massimo. Se ne servono più di 125 c'è qualcosa di sbagliato nei backbone nazionali.

Max packet size

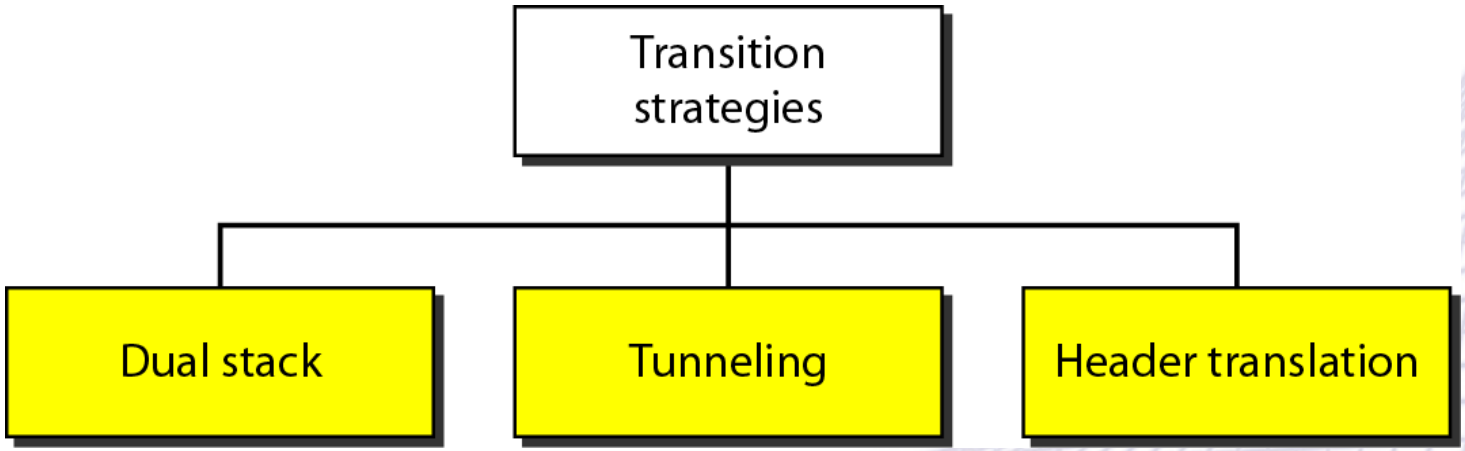
- La gente del supercomputing voleva pacchetti superiori a 64 KB
  - Quando un supercomputer comincia a trasferire non vuole essere interrotto ogni 64KB
  - D'altra parte pacchetti di 1MB su un link a 1.5 Mbps mi tengono la linea impegnata per oltre 5 secondi
- Compromesso:
  - Pacchetti normali sono limitati a 64 KB ma l'estensione hop-by-hop permette i jumbograms

Togliere il checksum

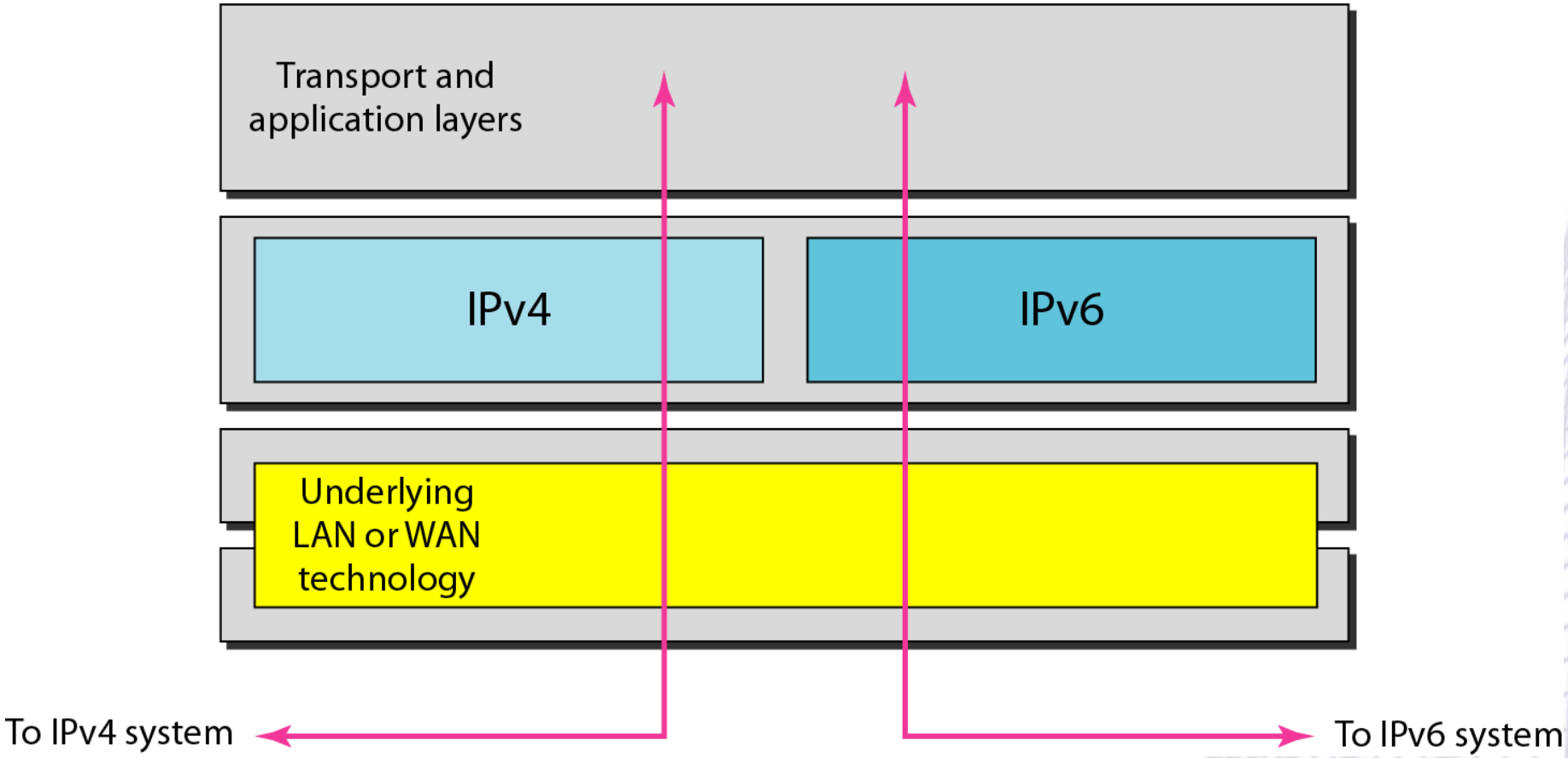
- Secondo alcuni era come togliere i freni ad una macchina. Va più veloce ma in caso di eventi inaspettati ci sono problemi
- L'argomento contro il checksum era che se un'applicazione tiene veramente all'integrità dei dati deve avere comunque un controllo a livello di trasposto
- Metterne un altro a livello IP è inutile e l'esperienza dice che è anche molto dispendioso in termini di risorse di calcolo
- Qui vincono gli anti-checksum

Transizione

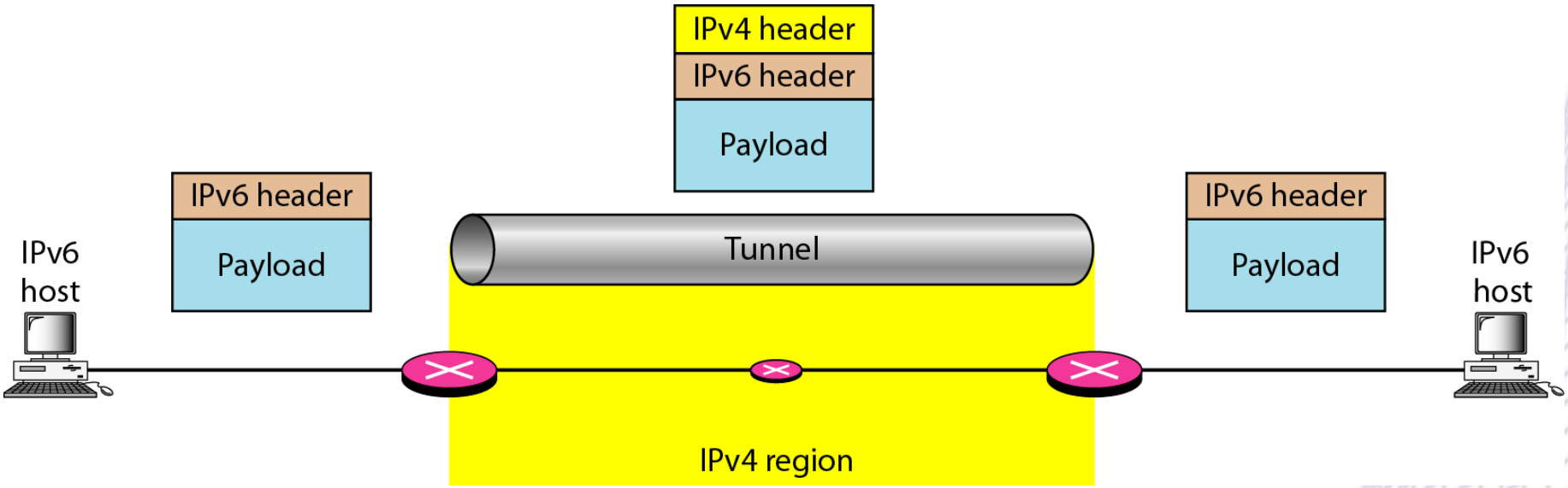
- La transizione è iniziata molto lentamente.
- All'inizio isole IPv6 potranno comunicare via tunnel su IPv4
- Alla fine le isole si fonderanno e rimarranno solo isole IPv4 che scompariranno
- Gli enormi investimenti in router IPv4 porteranno ad una transizione molto lenta che potrebbe durare più di una decina di anni
- Rumours di una possibile fine degli indirizzi IPv4 tra 2010 e 2016, ultimo blocco assegnato a Febbraio 2011



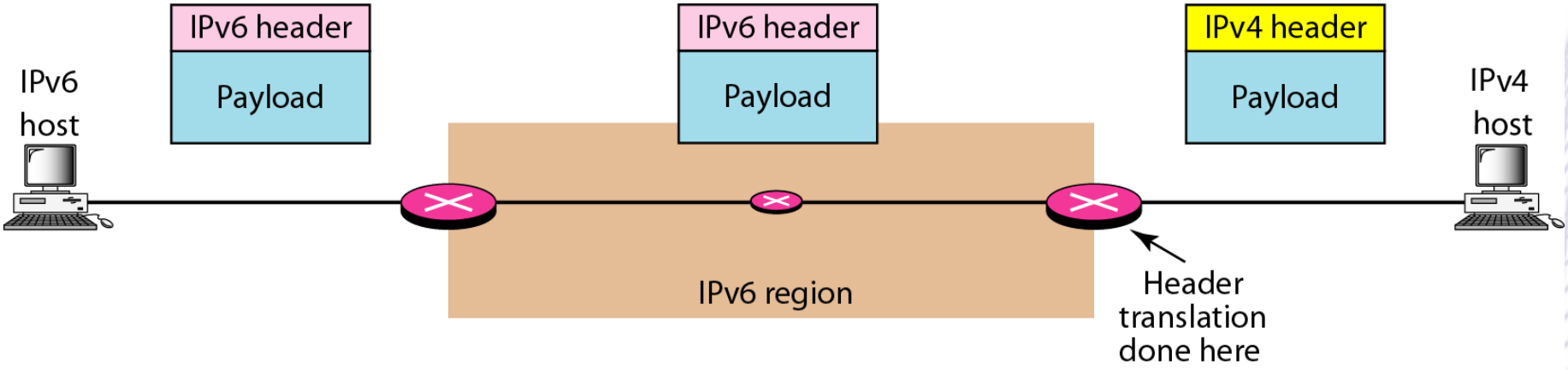
Dual stack


Transport and application layers
IPv4
IPv6
Underlying LAN or WAN technology
To IPv4 system
To IPv6 system

tunneling

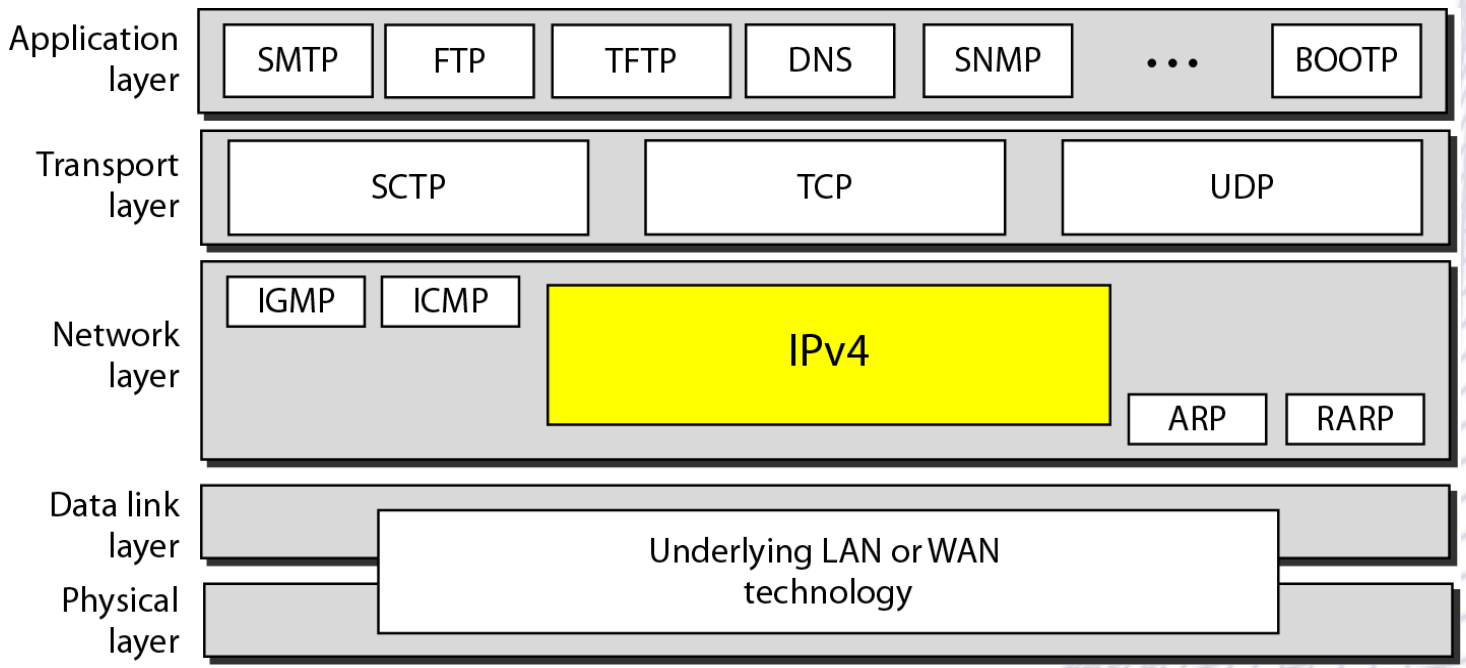

IPv4 header
IPv6 header
Payload
IPv6 header
Payload
Tunnel
IPv6 header
Payload
IPv6 host
IPv6 host
IPv4 region

Address translation



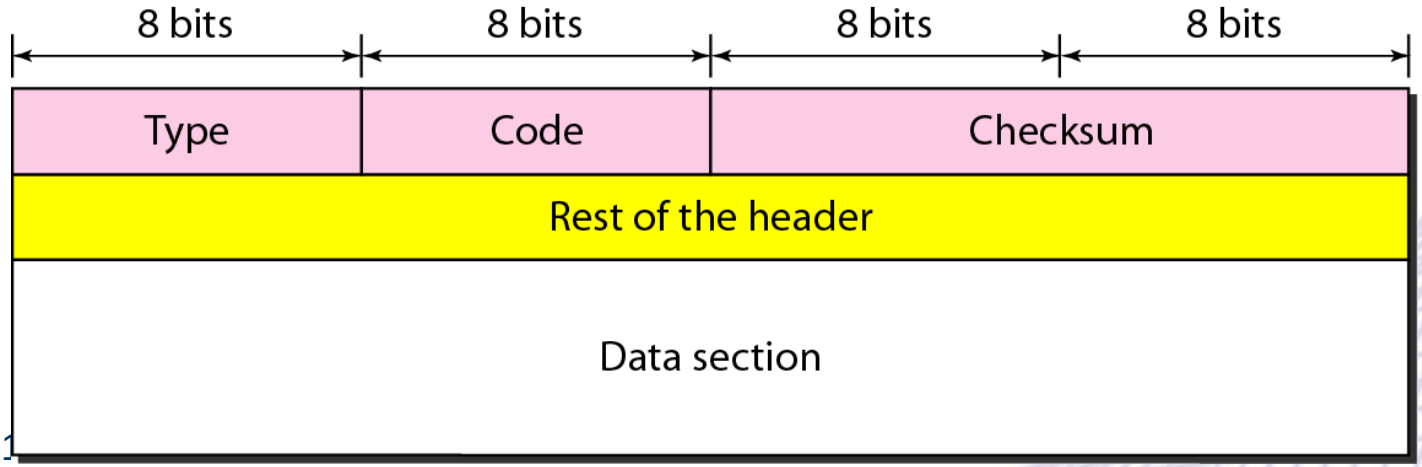
Internet Control Protocol

- Oltre a IP, Internet usa altri protocolli a livello Network
  - ICMP
  - ARP
  - RARP
  - BOOTP
  - DHCP



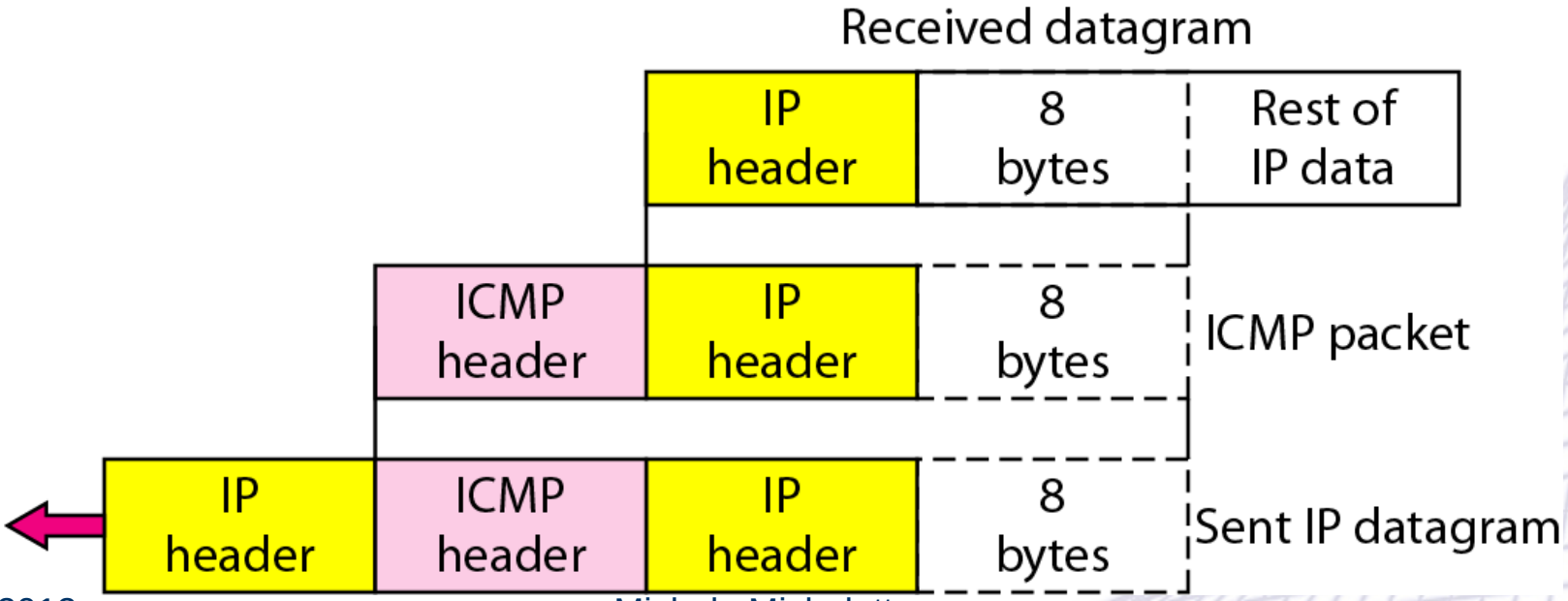
ICMP

- Internet Control Message Protocol
  - I router controllano il funzionamento di Internet
  - Quando succede qualcosa di inaspettato, vengono scambiati pacchetti di tipo ICMP
  - Sono usati anche per test



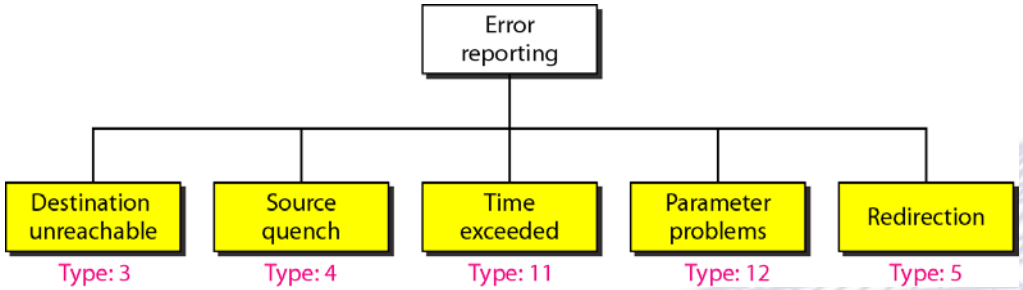
ICMP

- IP non gestisce gli errori.
- Gli errori sono gestiti appunto da ICMP
- I pacchetti ICMP viaggiano incapsulati in pacchetti IP. All'interno del pacchetto IP c'è la prima parte del datagram IP che ha dato errore



Messaggi ICMP

Message type	Description
Destination unreachable	Packet could not be delivered
Time exceeded	Time to live field hit 0
Parameter problem	Invalid header field
Source quench	Choke packet
Redirect	Teach a router about geography
Echo	Ask a machine if it is alive
Echo reply	Yes, I am alive
Timestamp request	Same as Echo request, but with timestamp
Timestamp reply	Same as Echo reply, but with timestamp



Vediamo i più importanti

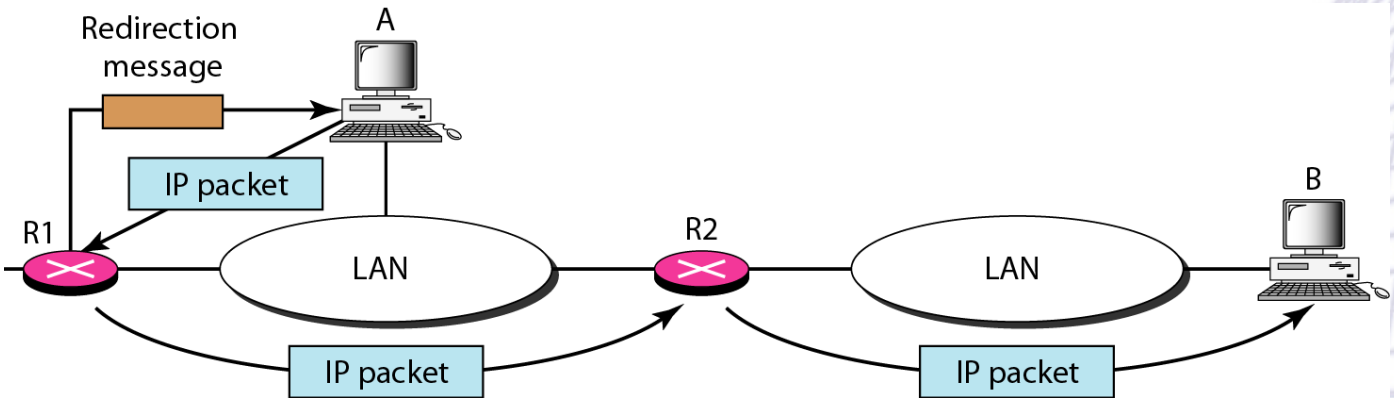
- DESTINATION UNREACHABLE
  - Usato quando la subnet o il router non riesce a trovare la destinazione (es non c'è un'applicazione che vuole ricevere il pacchetto) oppure quando il pacchetto con il campo DF (dont' fragment) non può essere inoltrato perché c'è di mezzo una rete "small packet"
- TIME EXCEEDED
  - Quando un pacchetto viene droppato perché il suo contatore è decrementato fino a zero. Sintomo di problemi di looping oppure TTL settato troppo basso.
  - Anche se il destinatario riceve solo alcuni frammenti di un datagram IP

Parameter e Source Quench

- PARAMETER PROBLEM
  - Indica un valore illegale in un campo del header. Possibile bug nell'host mittente o in un router di transito
- SOURCE QUENCH
  - Una volta era usato per frenare gli host che mandavano troppi pacchetti. Viene usato raramente perché questi pacchetti, in caso di congestione, peggiorano la situazione. La congestione viene gestita solitamente a livello di trasporto.

Redirezione

- REDIRECT
  - Usato quando un router nota che un pacchetto sembra essere stato inoltrato in modo incorretto (R1 si accorge che A potrebbe spedire direttamente a R2 per andare verso B)
- ECHO e ECHO REPLY
  - Usati per vedere se una destinazione è viva e raggiungibile. Ricevuto un ECHO la destinazione rispinde con un ECHO REPLY
- TIMESTAMP e TIMESTAMP REPLY
  - Sono simili ma vengono registrati anche tempo di partenza e arrivo dei pacchetti. Questo facilita misure di performance



Richiesta e risposta

- ECHO e ECHO REPLY
  - Usati per vedere se una destinazione è viva e raggiungibile. Ricevuto un ECHO la destinazione risponde con un ECHO REPLY
- TIMESTAMP e TIMESTAMP REPLY
  - Sono simili ma vengono registrati anche tempo di partenza e arrivo dei pacchetti. Questo facilita misure di performance
- MASK e MASK REPLY
  - Se un host conosce il proprio indirizzo ma non la maschera ad un router della rete o in broadcast

Programmi Diagnostici

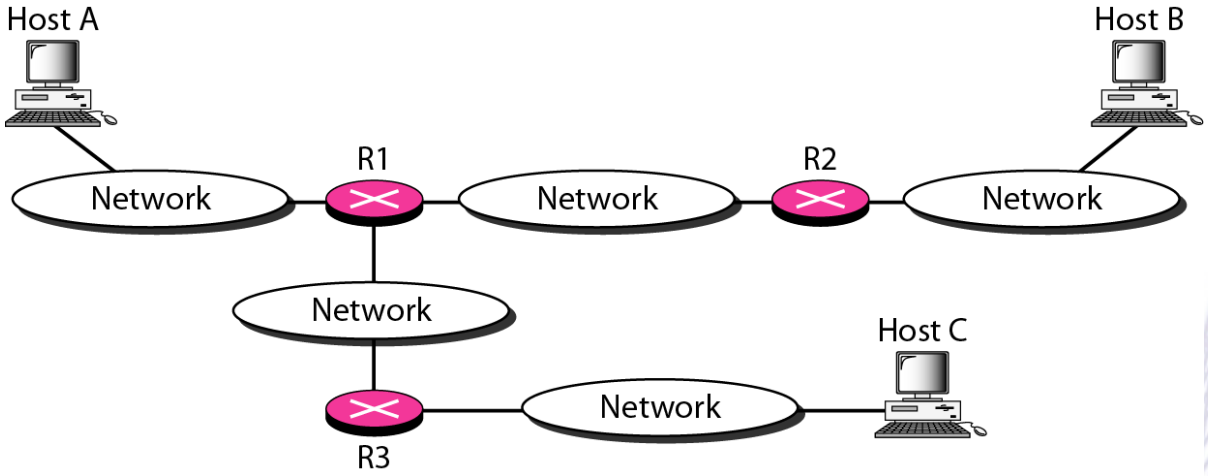
- **PING**
- 56B di dati
- 56+ 20(Header IP) +8(header ICMP)=**84**
- Riceve ogni volta 64Byte di risposta (56+8)
- Alla fine stampa statistiche min/average/max

```
$ ping fhda.edu
PING fhda.edu (153.18.8.1)   56 (84)  bytes of data.
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=0    ttl=62    time=1.91 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=1    ttl=62    time=2.04 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=2    ttl=62    time=1.90 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=3    ttl=62    time=1.97 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=4    ttl=62    time=1.93 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=5    ttl=62    time=2.00 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=6    ttl=62    time=1.94 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=7    ttl=62    time=1.94 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=8    ttl=62    time=1.97 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=9    ttl=62    time=1.89 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=10   ttl=62    time=1.98 ms

--- fhda.edu ping statistics ---
11 packets transmitted, 11 received, 0% packet loss, time 10103ms
      rtt min/avg/max = 1.899/1.955/2.041 ms
```

Programmi Diagnostici

- **Traceroute (tracert)**
- 38B di dati
- 10 di dati + 20(Header IP) +8(header ICMP)=38
- Manda ad ogni router 3 pacchetti ICMP poi INcrementa TTL




```
$ traceroute fhda.edu
traceroute to fhda.edu    (153.18.8.1), 30 hops max, 38 byte packets
 1  Dcore.fhda.edu        (153.18.31.254)    0.995 ms    0.899 ms    0.878 ms
 2  Dbackup.fhda.edu      (153.18.251.4)     1.039 ms    1.064 ms    1.083 ms
 3  tiptoe.fhda.edu       (153.18.8.1)       1.797 ms    1.642 ms    1.757 ms
```

traceroute

- Esempio di traceroute

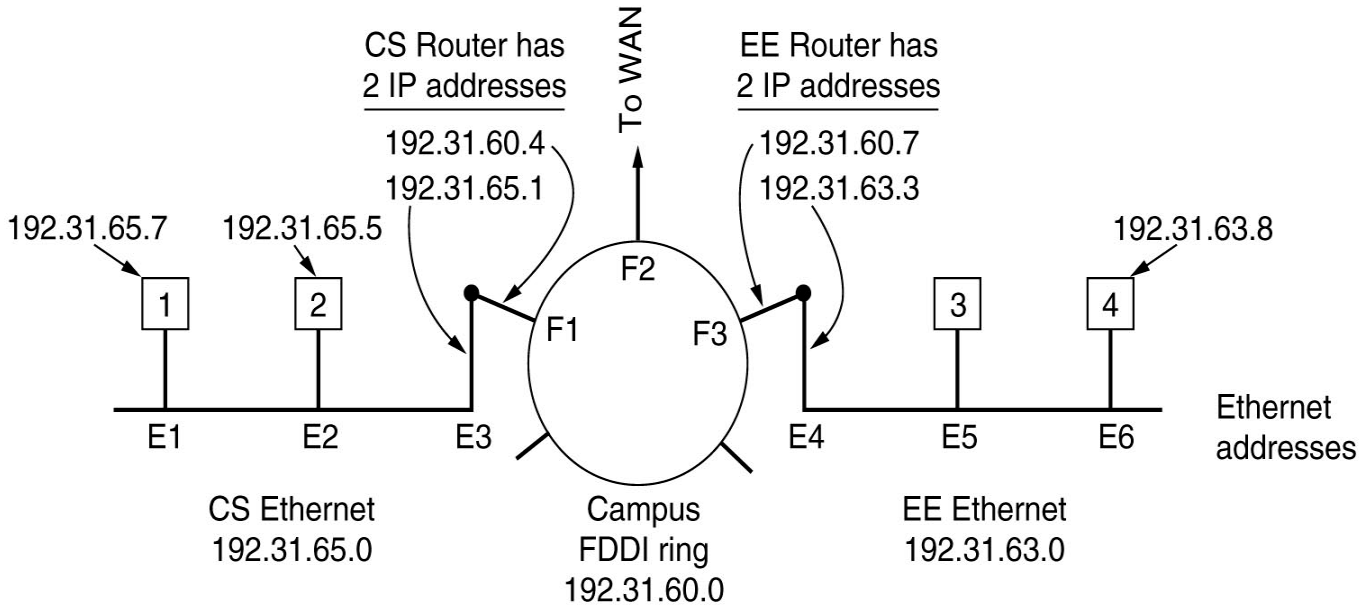
```
$ traceroute xerox.com
traceroute to xerox.com (13.1.64.93), 30 hops max, 38 byte packets
 1  Dcore.fhda.edu          (153.18.31.254)      0.622 ms   0.891 ms   0.875 ms
 2  Ddmz.fhda.edu           (153.18.251.40)      2.132 ms   2.266 ms   2.094 ms
 3  Cinic.fhda.edu          (153.18.253.126)     2.110 ms   2.145 ms   1.763 ms
 4  cenic.net               (137.164.32.140)     3.069 ms   2.875 ms   2.930 ms
 5  cenic.net               (137.164.22.31)      4.205 ms   4.870 ms   4.197 ms
      ....                    ....                ...        ....       ...
14  snfc21.pbi.net          (151.164.191.49)     7.656 ms   7.129 ms   6.866 ms
15  sbcglobal.net           (151.164.243.58)     7.844 ms   7.545 ms   7.353 ms
16  pacbell.net             (209.232.138.114)    9.857 ms   9.535 ms   9.603 ms
17  209.233.48.223          (209.233.48.223)    10.634 ms  10.771 ms  10.592 ms
18  alpha.Xerox.COM         (13.1.64.93)        11.172 ms  11.048 ms  10.922 ms
```

ARP

- Address Resolution Protocol
  - Molte macchine sono attaccate su reti Ethernet che mandano e ricevono pacchetti con indirizzi Ethernet a 48 bit. Non sanno nulla di indirizzi IP a 32 bit
  - Come mappare gli indirizzi IP su indirizzi data-link, per esempio Ethernet?
  - **Risoluzione statica?**
  - Non posso tenere in un file di configurazione tutte le coppie IP address MAC-Address della mia LAN.
    - Sarebbe difficile da mantenere quando passo le poche decine di nodi
    - Un nodo potrebbe cambiare scheda di rete e quindi indirizzo
    - Un laptop invece potrebbe cambiare di rete spontandosi da un sito all'altro, nuovi computer si sostuiscono a vecchi che sono eliminati

Esempio

- E1…Ex sono gli indirizzi MAC Ethernet
- F1…Fx sono gli indirizzi MAC FDDI
- 192.31… sono gli indirizzi IP
- Host 1 cerca host 2



Broadcast!

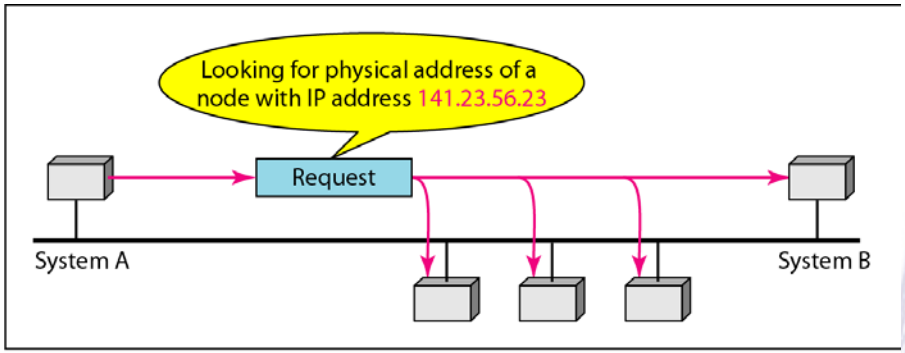
- Invece di usare una tabella fissa, ogni macchina impara la configurazione
  - La macchina 1 fa un broadcast di tipo Ethernet dicendo “Chi possiede l’indirizzo 192.31.65.5 ?”
  - Il broadcast raggiunge tutte le macchine sulla rete 192.31.65.0, ognuna controlla il proprio indirizzo IP ma solo la macchina 2 risponde con il suo indirizzo Ethernet E2
  - In questo modo E1 ha imparato che la macchina che cerca ha l’indirizzo E2
  - Questo protocollo si chiama ARP ed è definito in RFC 826

Semplice!

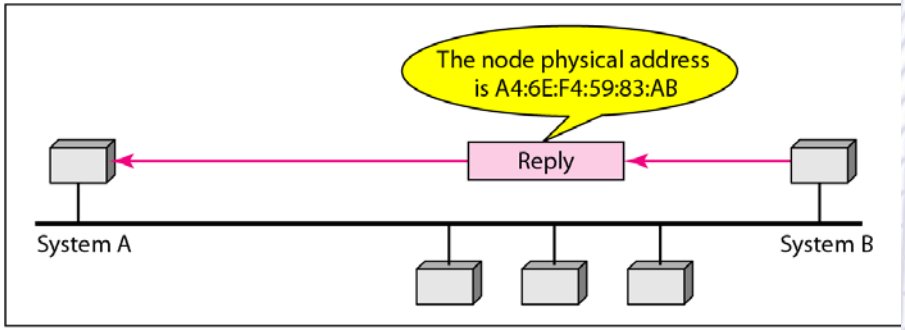
- Il vantaggio di ARP rispetto a file di configurazione sta nella semplicità
- L'amministratore deve solo assegnare un indirizzo IP e una netmask
- ARP fa il resto
- Infatti l'host 1 che ora sa che l'host 2 è nella sua rete, prende il pacchetto che viene da IP e lo mette in un frame Ethernet con indirizzo E2 e lo spedisce sul filo
- L'interfaccia di rete dell'host 2 vede il frame, estrae il contenuto IP e lo processa

Esempio

- System A manda una richiesta in Broadcast
- Tutti la vedono ma solo System B risponde in unicast



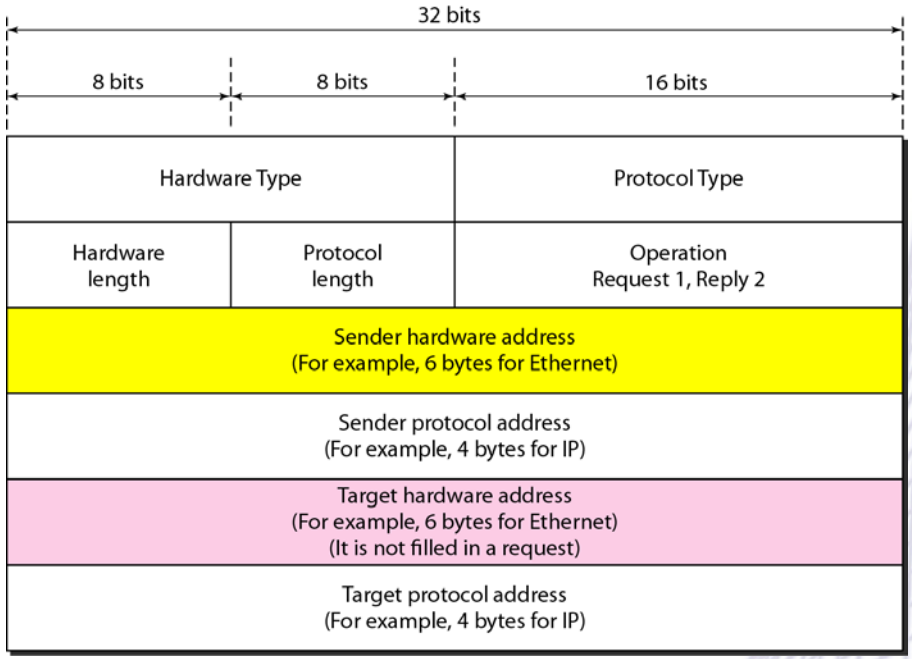
a. ARP request is broadcast

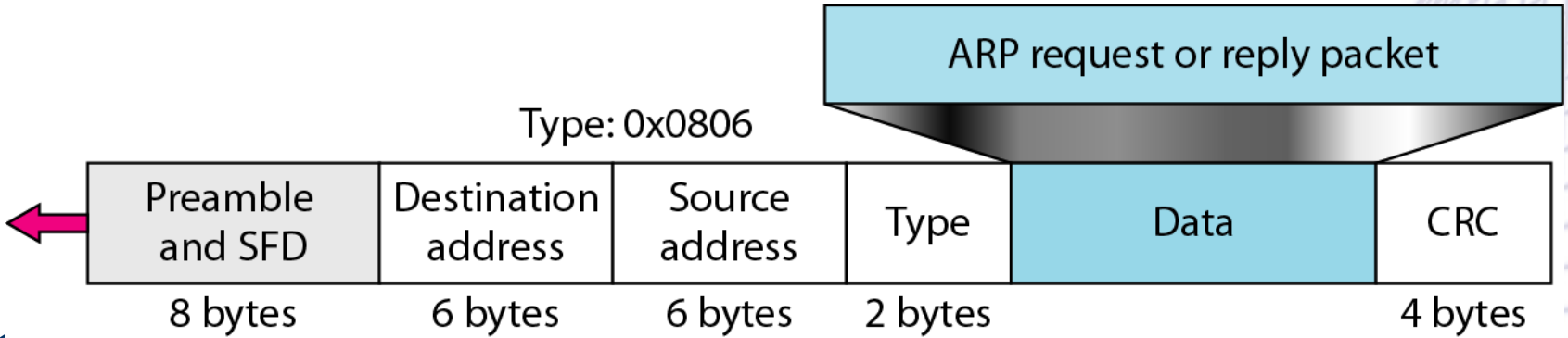


b. ARP reply is unicast

ARP Request

- Hardware type per Ethernet vale 1
- Protocollo: quali indirizzi di L3 vogliamo gestire, per IPv4 vale 0x8000
- Lunghezza indirizzo fisico (per Ethernet 48 bit) questa campo vale 6
- Lunghezza indirizzo protocollo (per IPv4 vale 4)
- Operation (1 req, 2 reply)





ARP cache

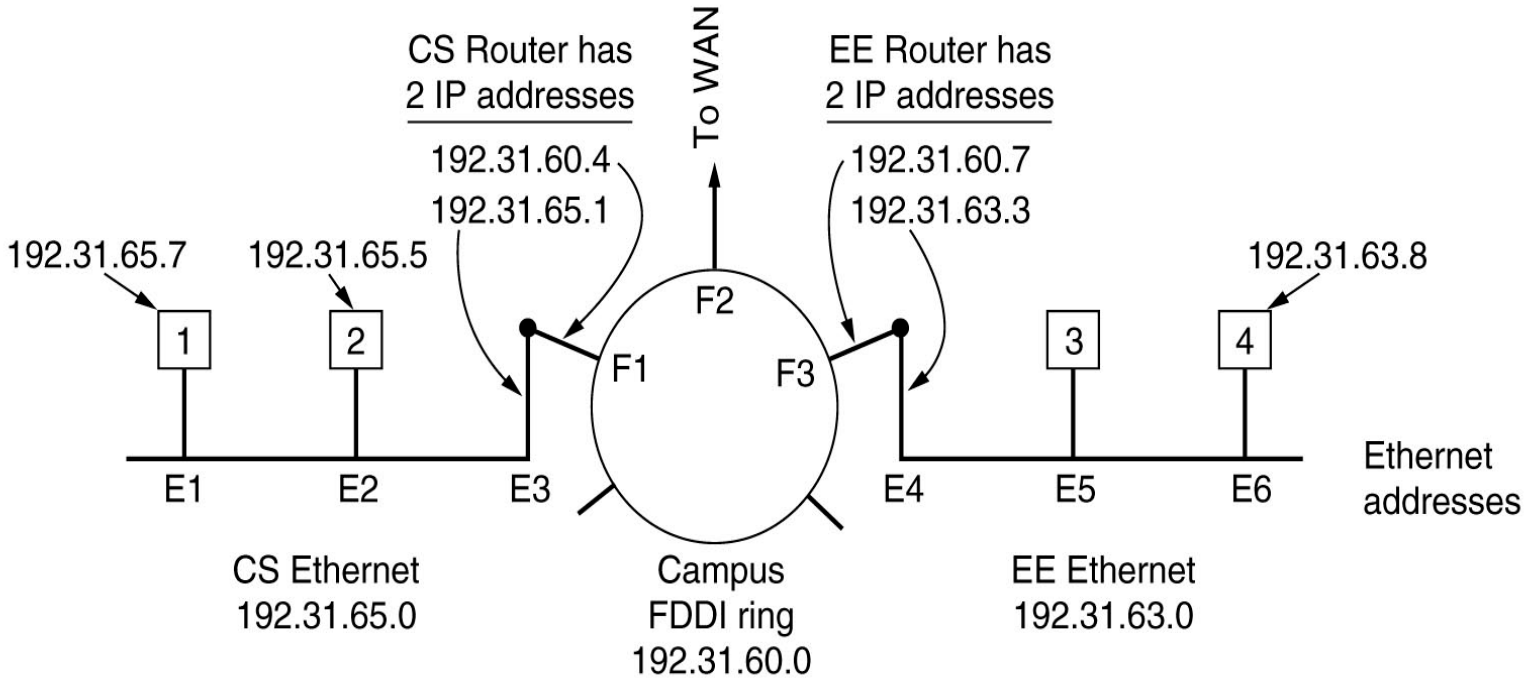
- Quando una macchina ha imparato qualcosa via ARP, se lo mette via in una cache.
  - Nel caso probabile che debba mandare di nuovo pacchetti a quella macchina non deve rifare il broadcast
  - Poi se la tiene buona per 20 – 30 minuti. Non oltre perché sono informazioni che possono cambiare nel tempo.
- L'host 2 non deve fare un broadcast per parlare con l'host 1
  - Deve solo ricordarsi che l'host 1 era quello che gli aveva mandato il broadcast e quindi è noto.
  - In realtà ogni macchina che ha ricevuto il broadcast dovrebbe mettere in cache le informazioni riguardo host 1

ARP cache

- Quando una macchina fa il boot può fare un broadcast sotto forma di richiesta del suo proprio indirizzo IP
  - Non ci dovrebbe essere risposta ma ogni macchina sulla Ethernet ora la conosce e l'ha messa nella sua ARP Cache
  - Se invece (inaspettatamente) una macchina risponde, vuol dire che ci sono due macchine con lo stesso indirizzo IP. L'ultima dovrebbe informare l'amministratore di rete del problema e non continuare il boot (o l'accesso alla rete)

ARP via router

- E1…Ex sono gli indirizzi MAC Ethernet
- F1…Fx sono gli indirizzi MAC FDDI
- 192.31… sono gli indirizzi IP
- Ora host 1 cerca host 4

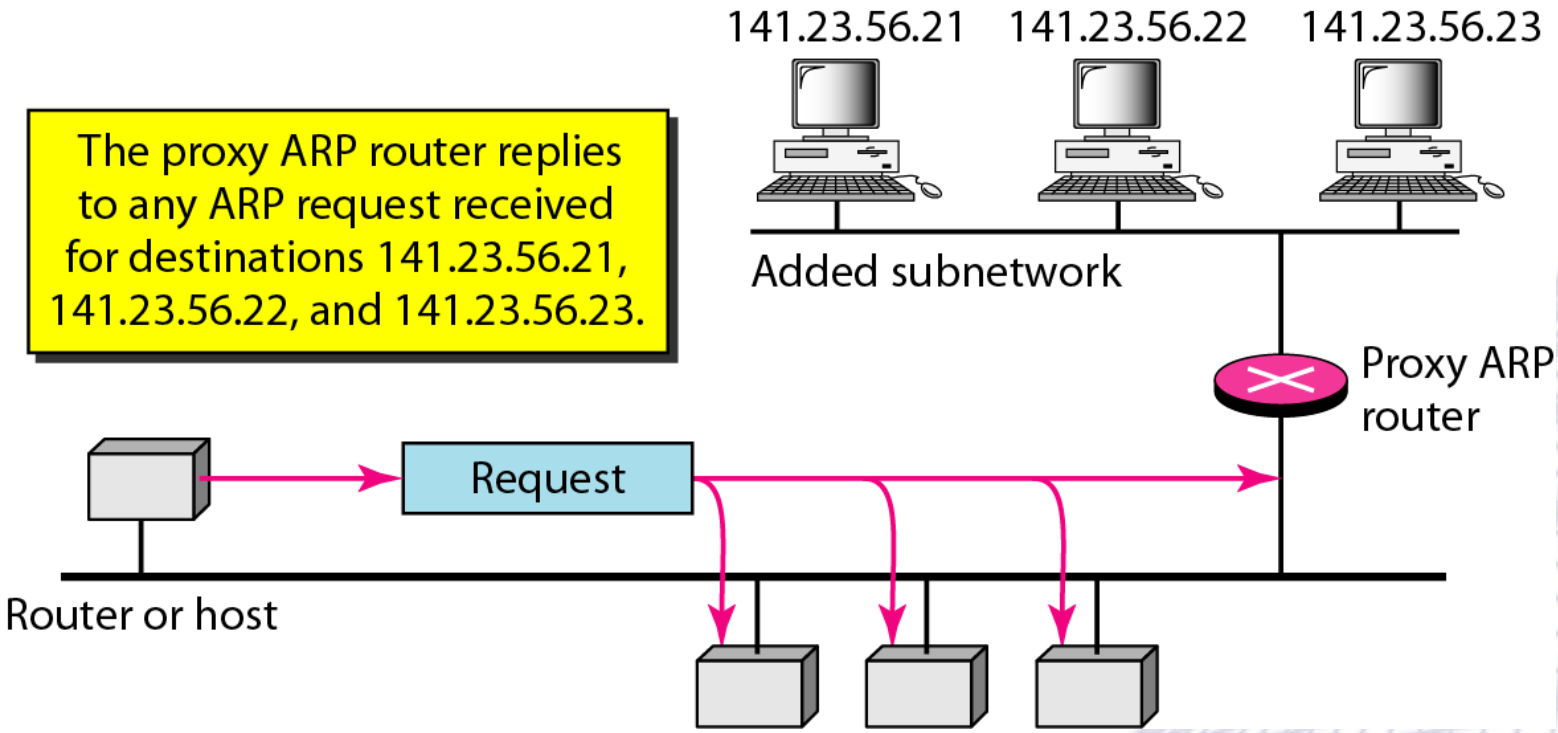


ARP via router

- Se host 1 usa il broadcast, host 4 (192.31.68.3) non lo sente perché il broadcast (Ethernet) non attraversa i router
- Due soluzioni
  - Proxy ARP
    - Il router CS potrebbe essere configurato per rispondere alle richieste ARP della rete 192.31.63.0 (e anche altre reti locali)
    - In questo caso host 1 mette nella sua cache una entry 192.31.68.3, E3) e manda tutto il traffico per host 4 al suo router locale
  - Host1 vede che la destinazione è su una rete remota e semplicemente manda tutto il traffico ad un indirizzo Ethernet di default che gestisce tutto il traffico remoto, in questo caso di nuovo E3
    - Il router CS ora DEVE sapere quali rete sta servendo

Proxy ARP

- Il proxy ARP manda i pacchetti destinati a quei 3 numeri verso quella sottorete aggiunta



Seconda soluzione

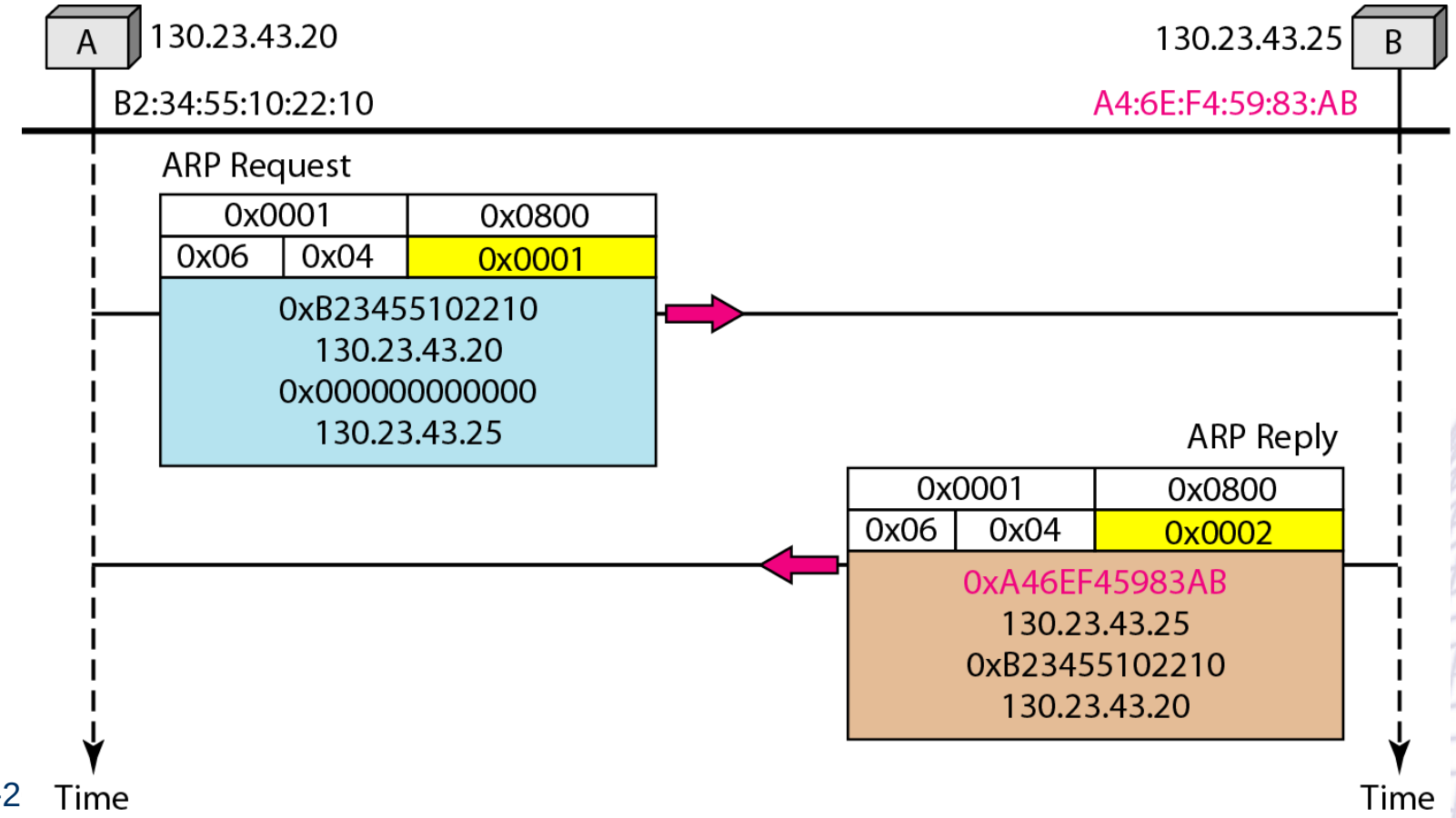
- L'altra soluzione è che host 1 vede che la destinazione è su una rete remota e semplicemente manda tutto il traffico ad un indirizzo Ethernet di default che gestisce tutto il traffico remoto
  - In questo caso di nuovo E3
  - Il router CS ora NON DEVE sapere quali reti sta servendo

Funzionamento

- In entrambi i casi host 1 mette il payload IP in un frame indirizzato a E3
- Il router vede il payload IP e cerca nella sua routing table dove mandarlo. Vede che per la rete 192.31.63.0 bisogna passare per il router 192.31.60.7
- Se non conosce l'indirizzo MAC FDDI (F3) del router EE lo cerca via ARP, poi comunque glielo manda in un frame FDDI
- Il router EE estrae il pacchetto, lo manda al sw IP, che vede la destinazione 192.31.63.8 e quindi lo mette in un frame Ethernet indirizzato ad E6 (se non conosce E6 chiaramente fa un ennesimo ARP)

Esempio ARP

- Flusso dello scambio di pacchetti



Come trovare un IP dal MAC?

- ARP permette di trovare l'indirizzo Ethernet che corrisponde ad un indirizzo IP
- Come faccio a fare il contrario?
  - Mi serve per esempio quando una stazione diskless fa il boot. Deve scaricarsi un file immagine di boot da un file server remoto. Ma come fa a sapere il proprio indirizzo IP?
  - Voglio assegnare dinamicamente gli indirizzi IP

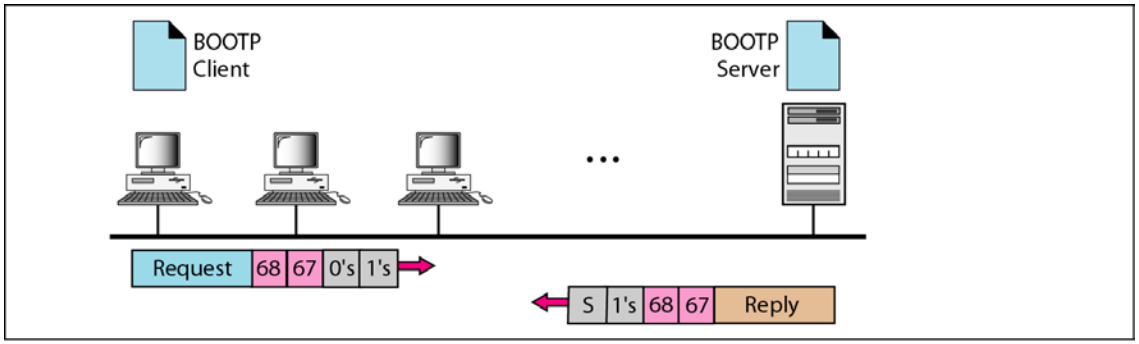
RARP

- Reverse Address Resolution Protocol: RFC903
  - Una stazione fa il broadcast del suo indirizzo dicendo: “il mio indirizzo Ethernet a 48bit è 14:04:05:18:01:05. Qualcuno conosce il mio numero IP?”
  - Il server RARP vede la richiesta, cerca l’indirizzo Ethernet nei suoi file di configurazione e risponde con l’indirizzo IP
  - Usare RARP è meglio che mettere l’indirizzo IP nel file immagine di boot perché posso usare la stessa immagine su tutte le macchine
  - Lo svantaggio è che usa un broadcast locale (con tutti i 48 bit a uno) per raggiungere il server RARP ma i broadcast non passano i router quindi serve un server RARP in ogni rete (o sottorete)

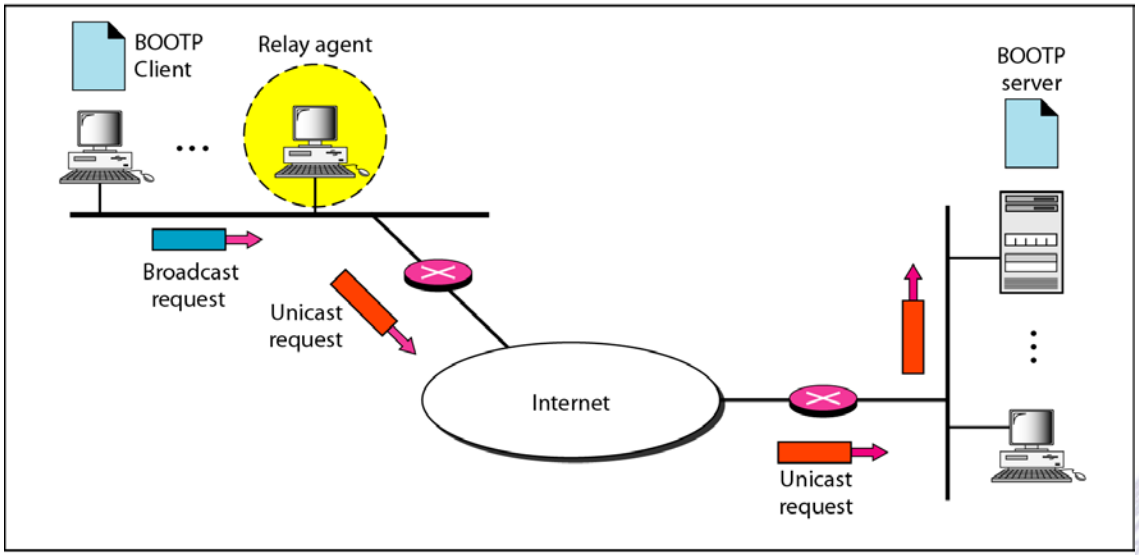
BOOTP

- BOOTP (RFC 951, RFC 1048, RFC 1084)
  - Al contrario di RARP, il protocollo BOOTP è a livello applicativo e usa pacchetti UDP che sono forwardati dai router
  - Inoltre risponde fornendo altre informazioni: per esempio l'indirizzo IP del server che contiene l'immagine di boot, l'indirizzo IP del default gw, e la subnet mask da usare
  - Lo svantaggio di BOOTP è che richiede una configurazione manuale delle tabelle che mappano numeri IP a numeri Ethernet
  - Quando aggiungo una nuova macchina, non posso fare il boot finché l'amministatore non ha assegnato un numero IP e inserito i due numeri (IP e MAC) nei file di configurazione di BOOTP
  - NB. Non avendo un numero IP come faccio ad usare UDP e IP? Uso l'indirizzo IP riservato 0.0.0.0
  - Se non c'è un nodo BOOTP nella mia rete ci deve essere un relay agent che conosce l'indirizzo IP del server BOOTP e gli forwarda la richiesta

BOOTP



a. Client and server on the same network



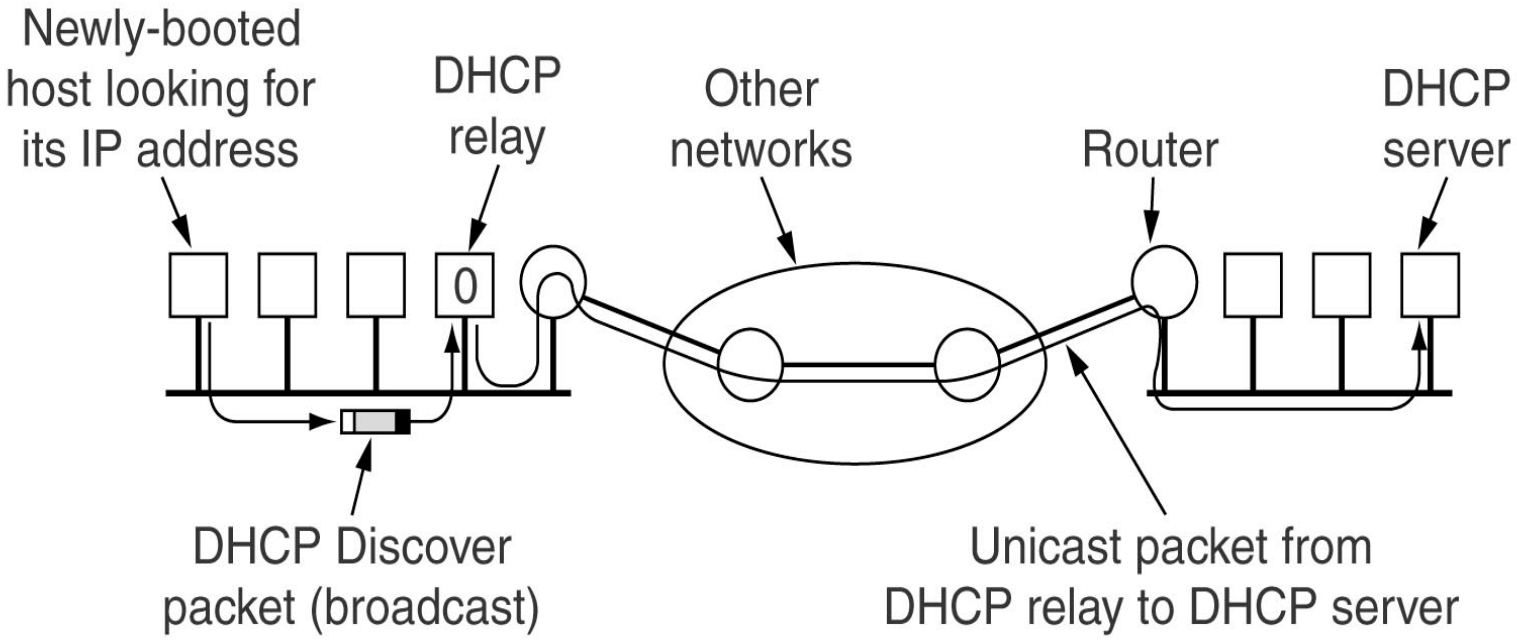
b. Client and server on different networks

DHCP

- Dynamic Host Configuration Protocol (RFC 2131 e 2132)
  - BOOTP non permette una gestione dinamica degli indirizzi. DHCP permette sia assegnazione dinamica che statica e ormai ha rimpiazzato RARP e BOOTP
  - Si basa sull'idea che un server speciale assegni numeri IP agli host che ne chiedono
  - Il server DHCP deve essere raggiungibile via broadcast, oppure serve un DHCP relay agent
  - Il server DHCP si configura in modo da dare un indirizzo a tutti coloro che richiedono oppure solo ad alcuni Mac-Address pre-registrati

DHCP relay agent

- Il client manda in broadcast un pacchetto DHCP Discovery
- Il DHCP relay intercetta tutte queste richieste di broadcast e le manda via unicast al server DHCP eventualmente su una rete remota. L'agente deve solo conoscere l'indirizzo IP del server DHCP

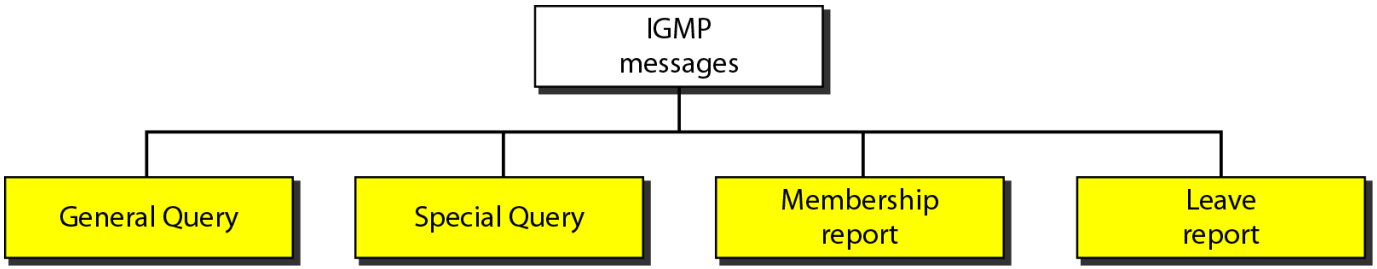


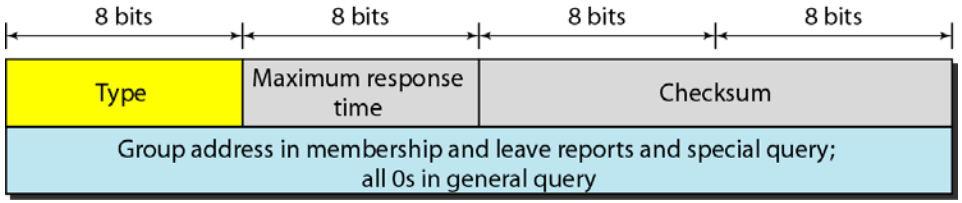
Lease time

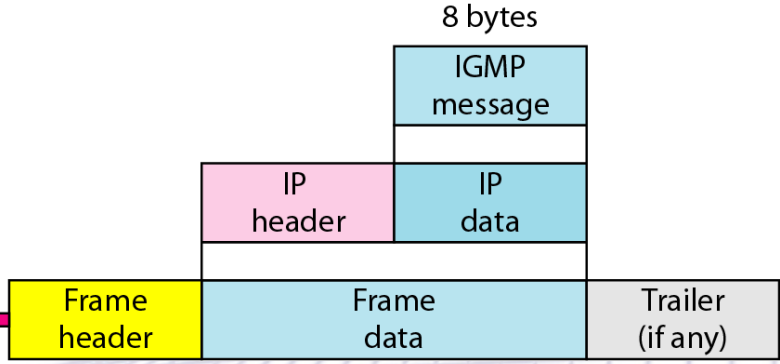
- Per quanto tempo è valido un indirizzo dinamico?
  - Deve avere un modo per riallocare gli indirizzi. Se un host lascia la rete e non restituisce l'indirizzo, questo sarà perso
  - Per evitare questo gli indirizzi IP sono concessi per un tempo limitato "Lease Time"
  - Prima della scadenza del Leasing l'host deve richiedere al server un rinnovo. Se questo viene negato l'host non deve usare ulteriormente l'indirizzo IP ricevuto in precedenza.

IGMP

- Permette ai router di gestire gruppi di nodi per spedire messaggi in multicast (a gruppi di destinatari)
- Usa 3 tipi di messaggi
  - di richiesta, di associazione, di dissociazione
  - Per i messaggi di richiesta viene specificato un tempo massimo per la risposta in decimi di secondo (max 100)
  - Group address sono gli indirizzi di classe D



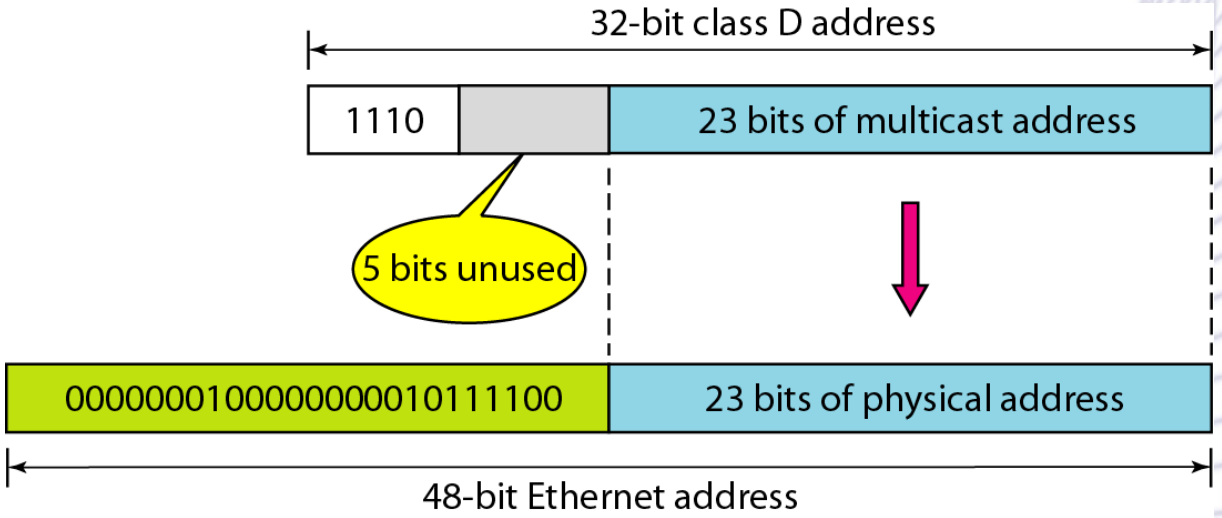




IGMP

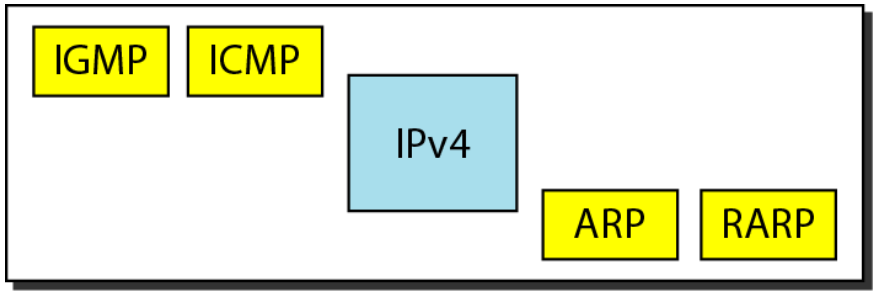
- Se viene incapsulato in L3 va messo in un datagram IP con valore di protocollo=2, TTL=1, l'indirizzo di destinazione secondo la tabella in alto
- A L2 come faccio a sapere a che nodo spedirlo?
- non posso fare una richiesta ARP per un indirizzo multicast.
- Ethernet supporta indirizzi fisici di tipo Multicast. In tal caso la traduzione avviene come in figura

Type	*IP Destination Address*
Query	224.0.0.1 All systems on this subnet
Membership report	The multicast address of the group
Leave report	224.0.0.2 All routers on this subnet

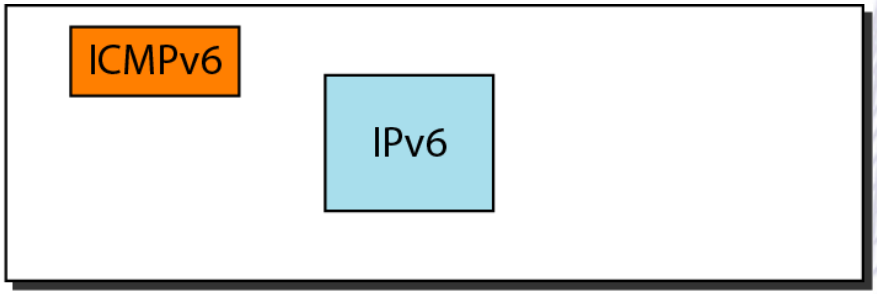


ICMPv6

- Anche IPv6 ha un versione di ICMP
  - ARP e IGMP sono stati incorportati in ICMP, mentre RARP è stato cancellato per obsolescenza



Network layer in version 4



Network layer in version 6

IPv6: Messaggi di notifica

- Manca source quench, si preferisce usare le priorità e le flow label di IPv6
- È stato aggiunto un messaggio di pacchetto troppo grande in quanto la frammentazione in IPv6 deve essere fatta dal mittente.

Type of Message	*Version 4*	*Version 6*
Destination unreachable	Yes	Yes
Source quench	Yes	No
Packet too big	No	Yes
Time exceeded	Yes	Yes
Parameter problem	Yes	Yes
Redirection	Yes	Yes

IPv6: Messaggi di request

- Timestamp e Address-mask eliminati. Non erano mai usati in IPv4
- I messaggi di IGMP e di ARP sono stati inglobati in ICMPv6

Type of Message	*Version 4*	*Version 6*
Echo request and reply	Yes	Yes
Timestamp request and reply	Yes	No
Address-mask request and reply	Yes	No
Router solicitation and advertisement	Yes	Yes
Neighbor solicitation and advertisement	ARP	Yes
Group membership	IGMP	Yes