Politecnico di Torino

Tecnologie e Servizi di Rete

Computer Engineering

Marco Lampis

3 dicembre 2022

Indice

0 Informazioni

I seguenti appunti sono stati presi nell'anno accademico 2022-2023 durante il corso di Tecnologie e Servizi di Rete.

Il materiale non è ufficiale e non è revisionato da alcun docente, motivo per cui non mi assumo responsabilità per eventuali errori o imprecisioni.

Per qualsiasi suggerimento o correzione non esitate a contattarmi.

E' possibile riutilizzare il materiale con le seguenti limitazioni:

- Utilizzo non commerciale
- Citazione dell'autore
- Riferimento all'opera originale

E' per tanto possibile:

- Modificar parzialmente o interamente il contenuto

Questi appunti sono disponbili su GitHub al seguente link:

```
1 https://github.com/Guray00/polito_lectures/tree/main/Tecnologie%20e%20
    Servizi%20di%20Rete
```

Figura 1: Repository GitHub

1 IPv4 Summary

In questo capitolo viene fatto un ripasso generico su quanto visto nei corsi precedenti, con particolare riferimento a Reti Informatiche (o equivalenti).

In ogni sottorete tutti i dispositivi che ne fanno parte avranno lo stesso indirizzo ip.

1.1 Indirizzi speciali

- tutti i bit a 1: indirizzo di broadcast, non può essere assegnato
- `127.x.x.x`: indirizzo di loopback, è una classe di indirizzi e servono a identificare l'host stesso e per tale motivo vengono solitamente utilizzate a scopo di debug.

Spesso oggi giorno non è consentito l'invio di messaggi in broadcast per motivi di sicurezza.

1.2 Indirizzamento IP con classi

Le rappresentazioni possono essere classes (a classe) o classness (senza l'utilizzo di classi). In particolare esistono di tre tipologie:

- **A**: prevede i primi 8 bit per l'indirizzo di rete, i rimanenti sono per identificare i dispositivi. Il totale degli indirizzi è 2^7 per la rete e 2^24 per i dispositivi. Si possono avere 128 networks.
- **B**: 2 bit per la classe, 14 bit per la rete e 16 bit per i dispositivi. Si possono avere 16384 networks.
- **C**: 3 bit per la classe, 21 bit per la rete e 8 bit per gli host.
- **D**: 4 bit per la classe, 28 bit per la rete e 4 bit per gli host. Questi indirizzi sono riservati per i multicast.

Basta guardare il primo bit per capire se era una classe A, B, C o D.

Nota:I bit di riconoscimento servono per sapere quali bit individuano la rete e quali gli host.

1.3 Indirizzamento IP senza classi (CIDR)

Il sistema ***C****lassless* ***I****nter****D****omain* ***R****outing* permette di indirizzare la porzione più precisa di indirizzi tra rete e e dispositivi. La porzione di rete è dunque di lunghezza arbitraria. Il formato con cui può essere rappresentato un indirizzo è il seguente: `networkID + prefix length` oppure `netmask`.

Il prefix length, specificato con `/x`, è il numero di bit di network.

La netmask è identificata da una serie di bit posti a 1 che determinano quali bit identificano la rete, attraverso un and bit a bit.

Esempio:

```
200.23.16.0/23                # prefix length
200.23.16.0 255.255.255.254.0 # netmask
```

L'indirizzo viene espresso attraverso gruppi di 8 bit, rappresentanti in modo decimale puntato (4 gruppi in quanto 32 bit totali). Ogni raggruppamento avrà un valore da 0 a 255.

Non tutti i valori sono permessi possibili, il più piccolo è 252. Questo è dovuto al fatto che abbiamo l'indirizzo dell'intera sottorete e l'indirizzo del inter broadcast che non possono essere utilizzati nell'assegnazione.

Un modo per sapere se un indirizzo è scritto in modo corretto è prendere il prefix length `/x` e controllare che ci l'ultimo numero puntato sia multiplo di 2^(32-x).

Esempi:

```
130.192.1.4/30  =>  4%2^(32-30) = 4%4  = 0 si!
130.192.1.16/30 => 16%2^(32-30) = 16%4 = 0 si!
130.192.1.16/29 => 16%2^(32-29) = 16%8 = 0 si!

130.192.1.1/30 => 1%2^(32-30) = 1%4  != 0 no!
130.192.1.1/29 => 1%2^(32-29) = 1%8  != 0 no!
130.192.1.1/28 => 1%2^(32-28) = 1%16 != 0 no!
```

Per il ragionamento di sopra appare evidente che un indirizzo che termina con `.1` non sarà mai un indirizzo corretto, in quanto ritornerà sempre un resto.

1.4 IP routing

Il routing degli host avviene attraverso la routing table, caratterizzata da due colonne che identificano:

- **destinazione** (indirizzi ip)

- **interfaccia** (eth0…)

Quando viene inviato un pacchetto, si cerca un match all'interno della tabella per identificare dove inviare un pacchetto IP. Se è presente più di un match, viene considerato quello con il prefisso più lungo.

nota: i router sono identificati solitamente con un cerchio con dentro una x.

Di seguito è mostrato un esempio di routing:

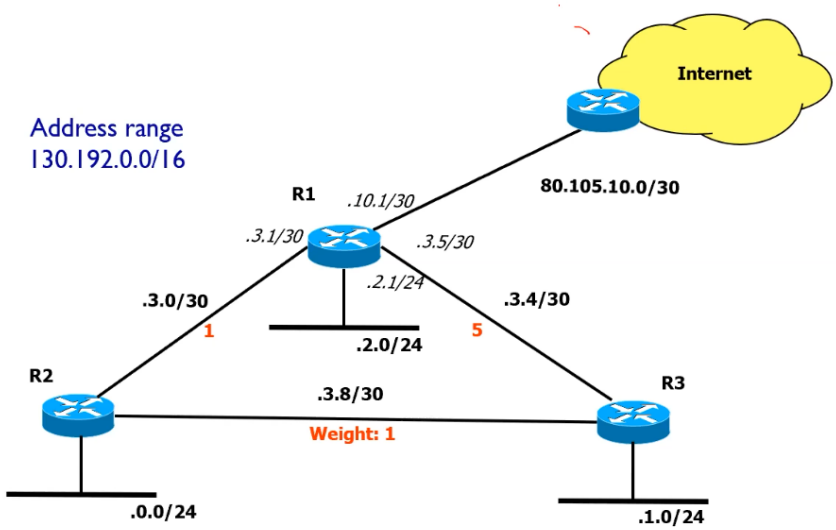


Figura 1.1: routing

Sono presenti in totale 7 sottoreti, di cui 3 reti locali e 4 reti punto punto. Tutta la sottorete ha come indirizzo quello raffigurato in alto a sinistra. Gli indirizzi di ciascuna di queste sono come segue:

Scriviamo la routing table del router identificando le reti direttamente connesse e raggiungibili. Prendiamo come riferimento **R1**:

Destination	Next	Type
130.192.3.0/30	130.192.3.1	direct
130.192.3.4/30	130.192.3.5	direct
130.192.2.0/24	130.192.2.1	direct
80.105.10.0/30	80.105.10.1	direct
80.105.10.0/30	80.105.10.1	direct

Destination	Next	Type
130.192.0.0/24	130.192.3.2	static
130.192.3.8/30	130.192.3.2	static
130.192.1.6/24	130.192.3.2	static

1.5 IP addressing methodology

La metodologia da adoperare è la seguente:

1. Localizzare le reti IP, *in questo caso 3*.
2. Individuare il numero di indirizzi richiesti, *in questo caso nel router in alto a destra è sufficiente `/30` perché ne sono richiesti 4 (2^2), `/26` a sinistra (2^6) e `/25` in basso a destra (2^7)*.
3. Quanti indirizzi posso allocare.
4. Il range di validità degli indirizzi,*in questo caso `/26`, `/25` e `/30` dunque mi basterebbe o tutti e 3, o due `/25` o infine un solo `/24`*
5. netmask / prefix length
6. Address range
7. Host addresses

Nota: in basso a sinistra sono richiesti 43 indirizzi per 40 dispositivi. Ciò è dovuto al fatto che oltre ai 40 richiesti serve l'indirizzo di rete, l'indirizzo di broadcast e l'indirizzo del router.

Per riuscire a trovare le sottoreti, si prosegue in ordine dal maggiore (decimale minore):

```
# tutta la rete
10.0.0.0/24

# subnet2 (/25), 32-25 = 7 => 2^7 = 128 indirizzi
# range: 0-127
10.0.0.0/25
10.0.0.127 <- ultimo

# subnet3 (/26), 32-26 = 6 => 2^6 = 64 indirizzi
# range: 128-191
10.0.0.128/26
10.0.0.191 <- ultimo

#subnet4 (/30), punto punto
10.0.0.192/30
```

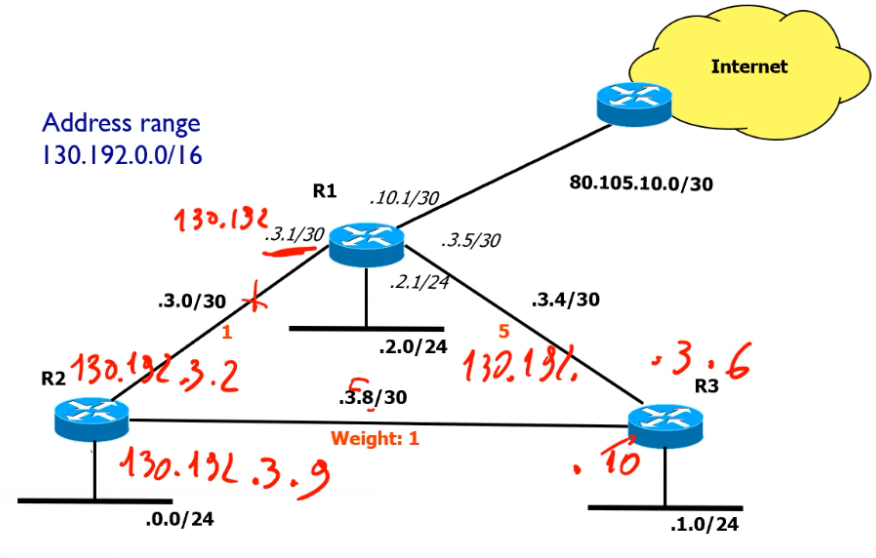


Figura 1.2: routing2

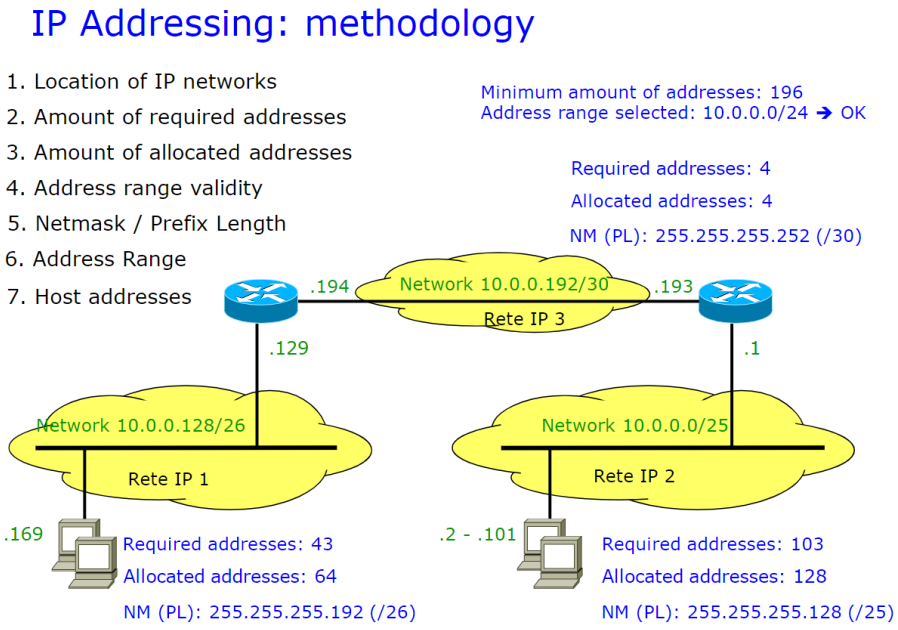


Figura 1.3: Rete di esempio

1.5.1 Esercizio 1

Numero di hosts	NetMask	Prefix Length	Available Addresses
2	`255.255.255.252`	(32-2) -> `/30`	$2^2 - 2 = 2$
27	`255.255.255.224`	(32-5) -> `/27`	$2^5 - 2 = 30$
5	`255.255.255.248`	(32-3) -> `/29`	$2^3 - 2 = 6$
100	`255.255.255.128`	(32-7) -> `/25`	$2^7 - 2 = 126$
10	`255.255.255.240`	(32-4) -> `/28`	$2^4 - 2 = 14$
300	`255.255.254.000`	(32-9) -> `/23`	$2^9 - 2 = 510$
1010	`255.255.252.000`	(32-10) -> `/22`	$2^10 - 2 = 1022$
55	`255.255.255.192`	(32-6) -> `/26`	$2^6 - 2 = 62$
167	`255.255.255.000`	(32-8) -> `/24`	$2^8 - 2 = 254$
1540	`255.255.248.000`	(32-11) -> `/21`	$2^11 - 2 = 2046$

Nota: per calcolare la netmask, si esegue $256 - 2^bit$

1.5.2 Esercizio 2

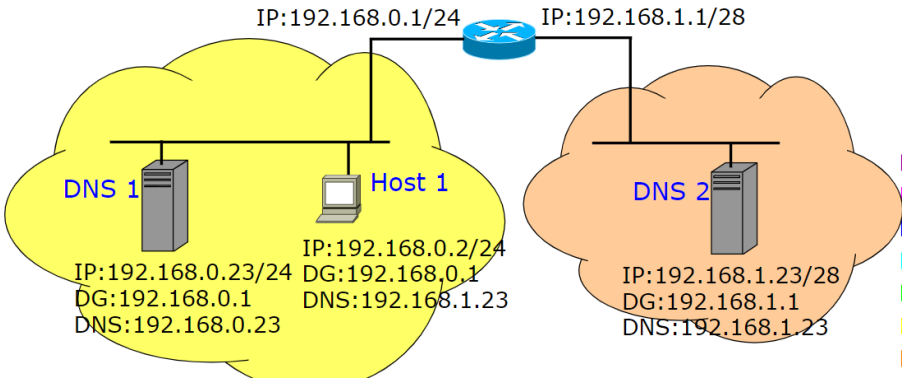
> Verifica se i seguenti indirizzi sono validi o meno.

IP / Prefix Length pair	Valido?
`192.168.5.0/24`	Si, gli ultimi 8bit sono a 0
`192.168.4.23/23`	No
`192.168.2.36/30`	Si, $36 \bmod 2^(32 - 30) = 0$
`192.168.2.36/29`	No, $36 \bmod 2^(32 - 29)! = 0$
`192.168.2.32/28`	Si, $32 \bmod 2^(32 - 28) = 0$
`192.168.2.32/27`	Si, $32 \bmod 2^(32 - 27) = 0$
`192.168.3.0/23`	No, $3 \bmod 2^(1)! = 0$
`192.168.2.0/31`	No, `/31` non ha senso

IP / Prefix Length pair	Valido?
`192.168.2.0/23`	Si, $2 \bmod 2(1)! = 0$
`192.168.16.0/21`	Si, $16 \bmod 2^3 = 0$
`192.168.12.0/21`	No, $12 \bmod 2^3 = 0$

1.5.3 Esercizio 3

> Trova l'errore di configurazione nella rete indicata di seguito e spiega il motivo per cui questa non funziona come dovrebbe.



1.5.4 Esercizio 4

> Definisci un piano di indirizzamento IP per la rete in figura. Considera entrambi i tipi di indirizzamento: "tradizionale" (senza minimizzare) e una soluzione che minimizzi il numero di indirizzi IP utilizzati. Assumi di utilizzare il range 10.0.0.0/16.

Partiamo evidenziando come il router a sinistra, al fine di servire 350 host, ha in realtà bisogno di 353 indirizzi: 350 host + 1 indirizzo di rete + 1 indirizzo di broadcast + 1 indirizzo del router, dunque `/23`. Stesso ragionamento è applicabile al router di destra, che ha bisogno di 123 indirizzi `/25`.

Troviamo così che `10.0.0.0/23` è la rete A (sinistra). Il suo indirizzo di broadcast sarà `10.0.1.255` in quanto adoperiamo 9 bit (quindi gli ultimi 8 bit a 1 e il primo bit del terzo gruppo a 1).

La sottorete C (destra) sarà identificata da `10.0.2.0/25` in quanto l'indirizzo immediatamente successivo. Il suo indirizzo di broadcast sarà `10.0.2.127`.

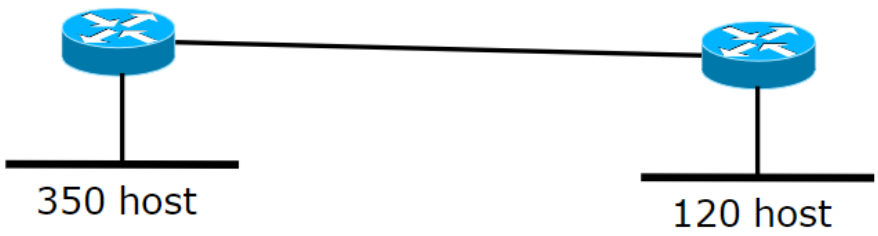


Figura 1.4: Rete

La sottorete B (centrale) sarà identificata da `10.0.2.128/30`, con `/30` proveniente dal fatto che è una sottorete punto punto.

Questa soluzione comporta un grosso spreco, in quanto c'è un `/25` che non viene utilizzato.

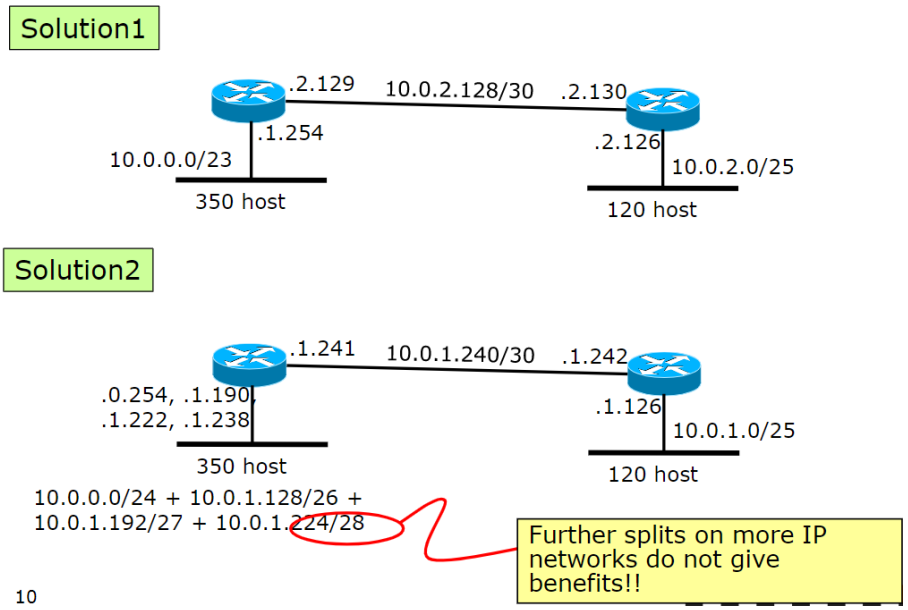


Figura 1.5: Soluzioni

1.5.5 Esercizio 5

> Definisci un albero di routing per tutti i nodi della rete mostrata di seguito.

L'**albero di instradamento** è quello che, a partire da un router della rete, stabilisce i percorsi minimi per raggiungere tutti i nodi. Per calcolare l'albero di instradamento si prende un router come riferimento, ad esempio **A**.

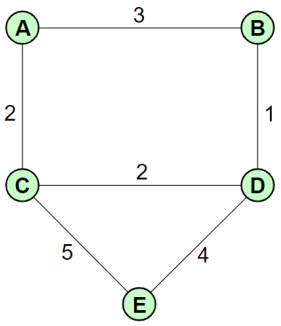


Figura 1.6: Rete esercizio 5

dest	next
B	3 (ramo dx)
C	2 (ramo inf)
D	4 (sia dx che inf)
E	7 (ramo inf)

La stessa procedura dovrà essere poi eseguita per tutti i nodi rimanenti, minimizzando le distanze. A parità di distanza solitamente ci sono motivi differenti per cui si scegli un percorso piuttosto che un altro (es router più nuovi).

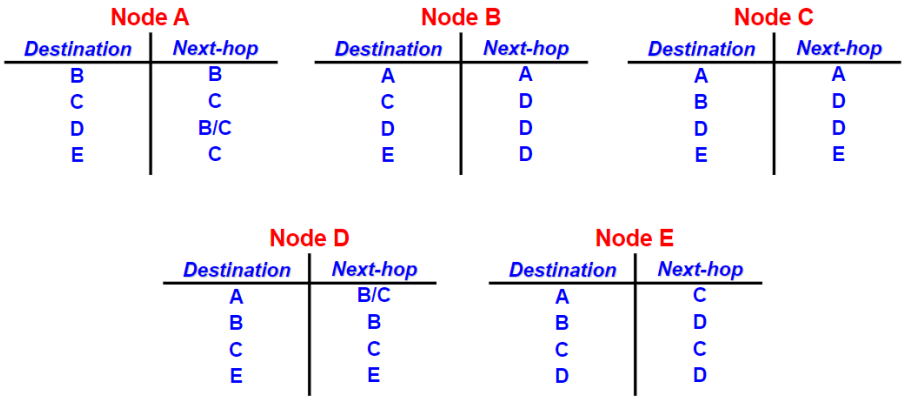


Figura 1.7: Soluzione esercizio 5

1.5.6 Esercizio 6

> Data la rete mostrata di seguito, definire la routing table di R1. La route aggregation deve essere massimizzata. Gli indirizzi ip mostrati in figura sono relativi all'interfaccia del router più vicino.

Cominciamo scrivendo la routing table di **R1**:

dest	next hop	Type
130.192.2.36/30 (A)	130.192.2.37	D

dest	next hop	Type
`130.192.1.126/30` (D)	`130.192.2.38`	S
`130.192.0.0/24` (E)	`130.192.2.38`	S
`130.192.1.128/25` (F)	`130.192.2.38`	S
`130.192.2.32/30` (G)	`130.192.2.38`	S

D ed **F** possono essere accorpati con `130.192.1.0/24`, che a sua volta può essere aggregato con e ottenendo l'indirizzo `130.192.0.0/23` avendo il valore di broadcast pari a `130.192.1.255`, per includere anche **G** è possibile usare `130.192.0.0/22`. Dobbiamo però stare attenti a controllare come questi si rapportano con le entry statiche. In questo caso le include tutte, e non è un problema.

1.5.7 Esercizio 7

> Realizzare un piano di indirizzamento che minimizza il numero di indirizzi necessari.

Troviamo la routing table di **R1**, analizzando ogni nodo a partire dai collegamenti diretti:

- Nella sottorete **A** sono presenti 27 host, per cui sono necessari 27+3 indirizzi e un prefix length di $(32 - 5) = 27$.
- Nella sottorete **B** sono invece necessari 120+3 indirizzi, per cui un prefix length di $(32 - 7) = 25$.
- Le sottorete C e D sono invece una sottoreti punto punto, per cui è necessario un prefix length di 30.
- La sottorete E ha bisogno di 60+3 indirizzi, per cui un prefix length di $(32 - 6) = 26$. Infine la sottorete F ha bisogno di 10+3 indirizzi, per cui un prefix length di $(32 - 4) = 28$.

Troviamo adesso quali sono gli indirizzi delle sottoreti, partendo da quella di dimensione maggiore (B, in quanto `/25`).

- **B**: `130.192.0.0/25`, con indirizzo di broadcast `130.192.0.127` in quanto gli ultimi 7 bit sono a 1.
- **E**: `130.192.0.128/26` con indirizzo di broadcast `130.192.0.191`
- **A**: `130.192.0.192/27`, con indirizzo di broadcast `130.192.0.223`
- **F**: `130.192.0.224/28`, con indirizzo di broadcast `130.192.0.239`
- **C**: `130.192.0.240/30`, con indirizzo di broadcast `130.192.0.243`
- **C**: `130.192.0.244/30`, con indirizzo di broadcast `130.192.0.247`

E' ora possibile calcolare gli indirizzi dei next hop, prendendo come riferimento il router più vicino:

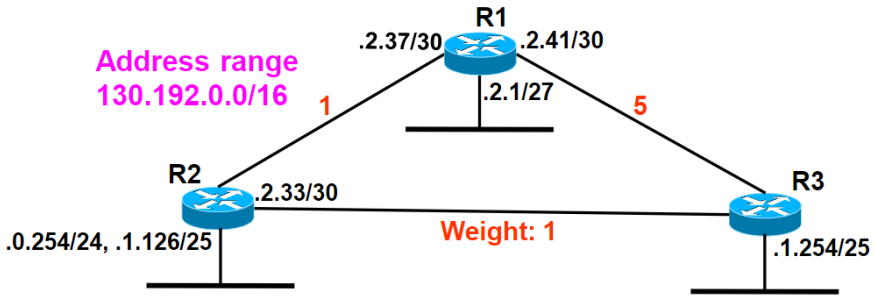


Figura 1.8: Esercizio 6

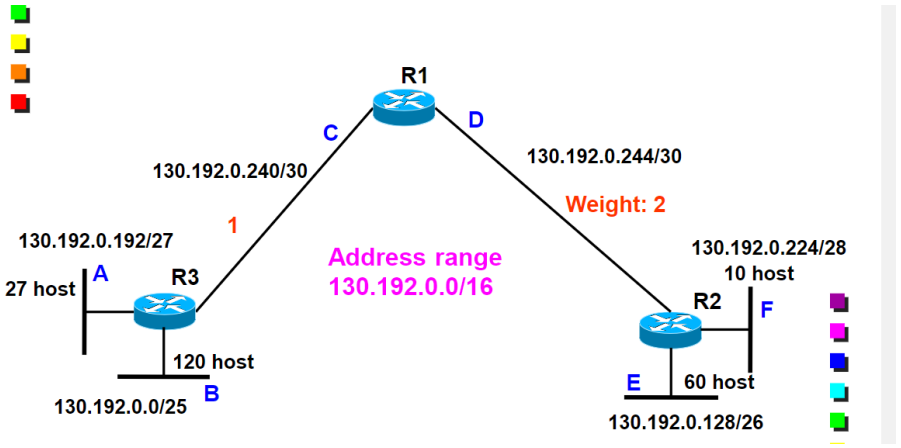


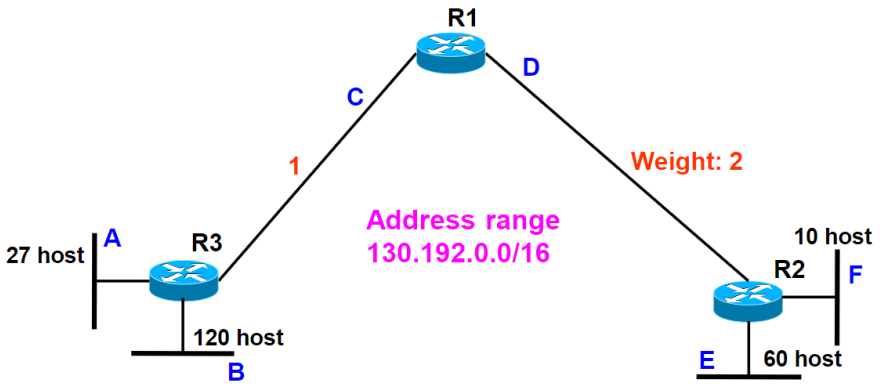
Figura 1.9: Esercizio 7

dest	Gateway	Type
`130.192.0.240/30` (C)	`130.192.0.241`	D
`130.192.0.244/30` (D)	`130.192.0.245`	D
`130.192.0.192/27` (A)	`130.192.0.242`	S
`130.192.0.0/25` (B)	`130.192.0.242`	S
`130.192.0.128/26` (E)	`130.192.0.246`	S
`130.192.0.224/28` (F)	`130.192.0.246`	S

Di queste entry bisogna valutare se è possibile fare qualche aggregazione. E' possibile farlo con **E** ed **F** in quanto: avendo `/26` e `28`, possono essere racchiusi in un `/25` (quindi 2^7) con il medesimo indirizzo di **E** (`130.192.0.128/25` è valido perché 128 % 128 = 0). La soluzione risulta comunque inefficiente perché non abbiamo ottenuto solo una entry.

1.5.8 Esercizio 8

> Realizzare un piano di indirizzamento che minimizza il numero di indirizzi necessari. Utilizzare il risultato della routing table di R1.



1.5.9 Esercizio 9

> Assumendo di avere interamente la cache libera, indicare il numero e il tipo di frames catturati da uno sniffer localizzato nella rete cablata dell'host A.

In una macchina Windows il ping viene eseguito 4 volte.

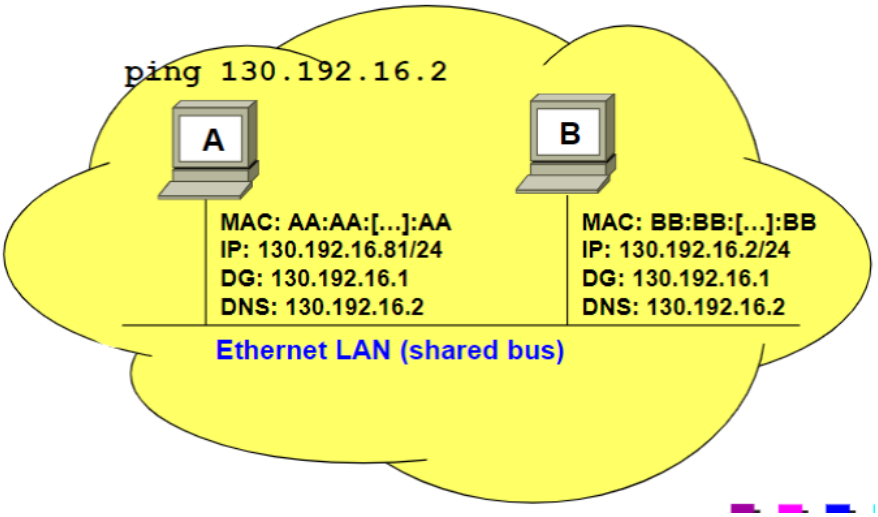


Figura 1.10: Esercizio 10

Bisogna innanzitutto verificare che le due macchine siano effettivamente nella stessa rete, lo si fa vedendo se hanno la stessa sottorete (in questo caso si, entrambi coerenti sulla `130.192.16.0/24`).

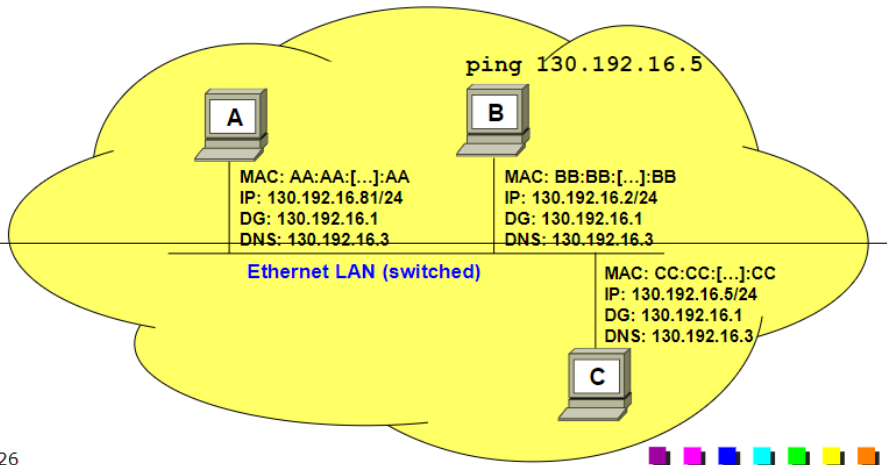
Scriviamo ora la tabella:

ID	MACS	MACD	IPS	IPD	DESCRIZIONE
1	MACA	broadcast	-	-	ARP Request
2	MACB	MACA	-	-	ARP Response
3	MACA	MACB	IPA	IPB	ICMP echo request
4	MACB	MACA	IPB	IPA	ICMP echo response

Il passaggio 3 e 4 sono quelli eseguiti 4 volte.

1.5.10 Esercizio 10

> Assuming that all caches are empty, indicate the number and the type of the frames captured by a sniffer located sulla rete dell'host A.



La sottorete di A ha indirizzo della sottorete pari a `130.192.16.0`, è errato il prefix length in quanto viene indicato `/24` invece di `/23`.

A quando comunica per parlare con il DNS, che è all'esterno della sua sottorete, parla con il suo default gateway.

ID	MACS	MACD	IPS	IPD	DESCRIZIONE
1	MACA	broadcast	-	-	ARP Request
2	MACDG	MACA	-	-	ARP Response
3	MACA	MACDG	IPA	IPDNS	DNS request
4	MACDG	broadcast	-	-	ARP request
5	MACDNS	MACDG	-	-	ARP response
6	MACDG	MACDNS	IPA	IPDNS	DNS request
7	MACDNS	broadcast	-	-	ARP request
8	MACA	MACDNS	-	-	ARP response
9	MACDNS	MACA	IPDNS	IPA	DNS response
10	MACA	MACDG	IPA	IP google	ICMP echo request
11	MACDG	MACA	IP google	IPA	ICMP echo response

Essendo uno shared bus tutti i pacchetti sono condivisi, solo che che chi non è interessato ai pacchetti che riceve li scarta. *Nota: DG viene utilizzato per indicare default gateway; arp è di livello 2.* Il traffico viene ottenuto prima che entri nel nodo A.

Il passaggio 10 e 11 sono quelli eseguiti 4 volte.

1.6 Multicast

Il multicast è un concetto che sta nel mezzo tra una comunicazione unicast (1 a 1) e broadcast (1 a tutti). Una sorgente A manda i pacchetti ad *alcuni* host. Ci sono dunque dei gruppi a cui degli host possono entrare o uscire. E' vantaggioso in quanto l'alternativa sarebbe mandare pacchetti uno ad uno in modo molto più lento. Nel multicast viene inviato un solo pacchetto, che viene poi instradato correttamente dal router ai destinatari utilizzando meno traffico (nel broadcast è sempre un pacchetto, ma viene poi mandato a tutti appesantendo). In IPv4 viene utilizzato poco perché si ha problemi con l'indirizzamento.

E' ampiamente utilizzato in IPv6 ed è chiave per la comunicazioni tra gruppi (videoconferenze, video broadcast ecc).

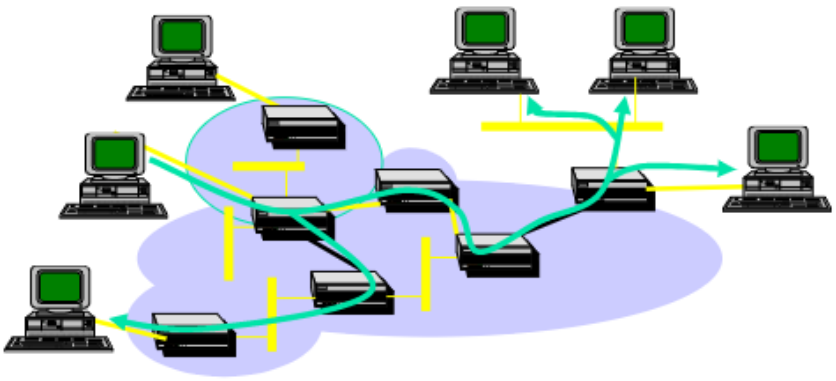


Figura 1.12: Multicast

A ogni gruppo multicast viene associato un indirizzo IPv4. Questo indirizzo è un indirizzo di classe D, che è un indirizzo di broadcast. Fanno parte del range `224.0.0.0` - `239.255.255.255` che sono riservati, ed è per questo necessario acquistarne uno per utilizzarlo.

Il protocollo prevede che il livello 2 scarti i pacchetti che non sono di interesse, ma comunque è possibile associare un indirizzo di livello 2 al livello 3 in modo che possa essere scartato successivamente. L'indirizzo MAC è formato da 48 bit, rappresentato in forma compatta da gruppi di 8 bit ognuno dei quali rappresentato da 2 cifre esadecimali. La parte alta, solitamente riservata al produttore, ha invece la costante `01-00-5E-0` che identifica la mappatura per un totale di 25 bit (l'ultimo gruppo è solo un bit). La mappatura è fatta non comprendendo tutti i casi ma cercando di ridurre il numero di collisioni.

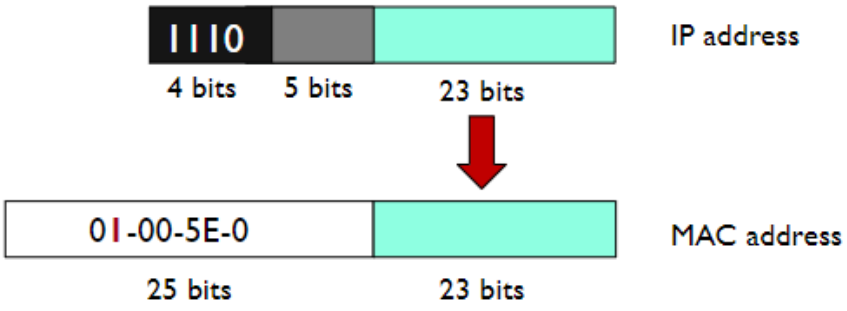


Figura 1.13: Mappatura IP a MAC

2 IPv6

IPv6 nasce per soddisfare le esigenze di un maggior numero di indirizzi, superando i limiti di IPv4. La nuova versione del protocollo risulta sotto molti punti di vista superiore, anche se IPv4 è ancora in uso e non è ancora stato completamente sostituito e nel corso degli anni è stato ampiamente esteso e migliorato.

Altre motivazioni che hanno portato alla nascita di IPv6 sono:

- Più efficiente sulle LAN
- Supporto di Multicast e Anycast
- Sicurezza
- Policy routing
- Plug and Play
- Traffic Differentiation
- Mobility
- Quality of Service support

Per riuscire a definire il protocollo IPv6 ha richiesto molto tempo e siamo attualmente in una fase di migrazione (richiedendo soluzioni temporanea applicate su IPv4).

2.1 Perché IPv4 non basta e soluzioni

Il protocollo IPv4 ha indirizzi di lunghezza 32 bit, con un totale di circa 4 miliardi di indirizzi. Nonostante ciò„ solo parte di questi indirizzi possono essere utilizzati a causa dell'utilizzo di classi, multicast, ecc. Inoltre, molti di questi sono utilizzati in modo gerarchico: il prefisso usato in una rete fisica non può essere usato in una differente. Infine, molti di questi indirizzi IP risultano non utilizzati, causando un grande spreco.

Alcune delle soluzioni utilizzate per risolvere questi problemi sono:

- Introduzione di reti "su misura" mediante l'utilizzo di netmask.
- Indirizzi privati (intranet), ma non abbastanza da risolvere il problema.

- NAT, che però rompe la connessione end to end aumentando il carico dei gateway e la relativa complessità
- ALG (Application Layer Gateway).

2.2 Chi assegna indirizzi IP

Gli indirizzi IP vengono assegnati da parte dell'organizzazione IANA, che assegna a ciascun Regional Internet Registry (RIR) un blocco di /8 indirizzi ip:

- AFRINIC: Africa
- APNIC: East Asia, Australia and Oceania
- ARIN: USA, Canada and some Caribbean islands
- LACNIC: South America, Mexico and some Caribbean islands
- RIPE NCC: Europe, Middle East and Central Asia

Successivamente, le RIR dividono i blocchi in blocchetti di dimensione minore da assegnare alle National Internet Registries (NIR) e alle Local Internet Registries (LIR).

2.3 Address pool status e scalabilità

Ogni singolo indirizzo IPv4 può essere in uno dei seguenti stati:

- part of the IANA unallocated address pool,
- part of the unassigned pool held by an RIR,
- assigned to an end user entity but unadvertised by BGP, or
- assigned and advertised in BGP

Ciò comporta dei problemi anche in termini di scalabilità, dovuti:

- dimensione delle routing table (ogni subnet network deve essere advertised)
- Risorse dei router limitate (troppe informazioni da gestire)
- Limitazioni dei protocolli di routing (spesso i router cambiano)
- Perlopiù riguarda i router backbone

Sono state tentate alcune soluzioni, come:

- aggregazione di router
- CIDR (Classless Inter-Domain Routing)
- Limitazione di assegnamento di prefissi IP "non razionali" e indirizzi IP (es vendita di /8)

Ma nonostante ciò il problema persiste, in particolare la scalabilità dei protocolli di routing risulta attualmente non risolvibile.

2.4 Indirizzi IPv6

E' stato scelto, attraverso un approccio di tipo scientifico e con un focus sull'efficienza, l'utilizzo di indirizzi di lunghezza pari a **128 bit**, con un totale di 2^128 indirizzi.

La notazione non è più puntata, ma bensì si è deciso di dividere in gruppi di **2 byte** (4 cifre esadecimali) separati dal carattere `:`. E' possibile utilizzare due regole per rendere più compatto l'indirizzo:

- è possibile rimuovere cifre pari a `0`. Esempio: da `1080:0000:0000:0000:0007:200:A00C:3423:A089` a `1080:0:0:0:7:200:A00C:3423:A089`.
- e' possibile omettere un gruppo di soli zeri inserendo `1080::7:200:A00C:3423:A089`, ma è lectio **solo una volta**. Questo perché non saprei quanti zeri inserire ciascuna volta.

2.5 Routing

Il routing IPv6 è stato pensato in modo da non modificare la struttura adoperata in IPv4, a eccezione della lunghezza degli indirizzi.

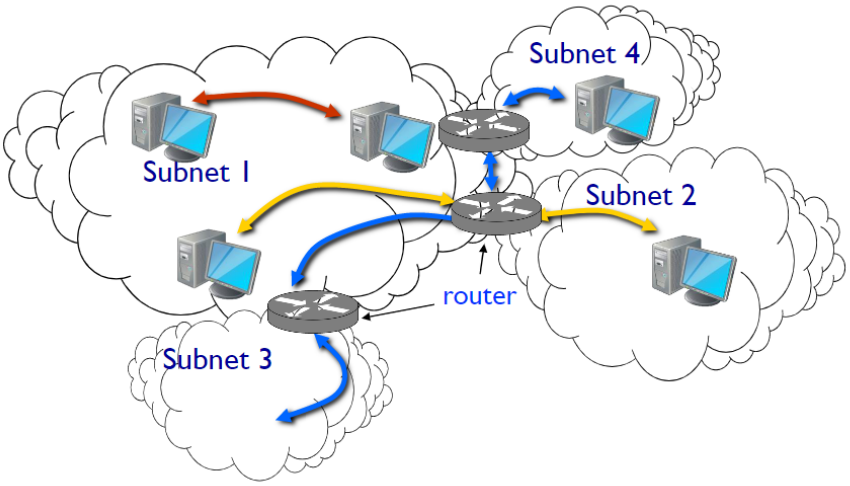


Figura 2.1: Routing

Per dividere la parte del prefisso di rete e la parte dell'interfaccia si è deciso, per il momento, di applicare

una separazione a metà con un prefisso di rete pari ad n=64, ma prevedendo che in futuro potremmo aver bisogno di un prefisso di rete più lungo.

Il concetto di aggregazione rimane il medesimo, è infatti possibile utilizzare il prefix length come già visto, ad esempio: FEDC`:0123:8700::100/40`. Non è necessario l'utilizzo di classi.

> **Nota**: non sarà, per quanto detto precedentemente, superiore a 64.

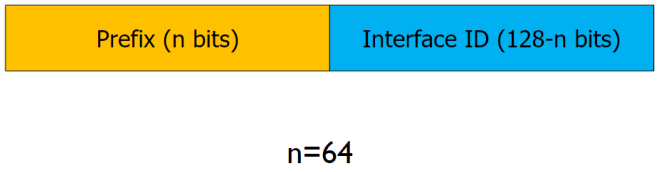


Figura 2.2: Struttura dell'indirizzo

I principi di assegnamento sono i medesimi dell' IPv4, con alcune differenze in termini di terminologia:

- **Link**: physical network
- **Subnetwork**: Link

Dividiamo le comunicazioni in:

- **On-link**: gli host hanno lo *stesso prefisso*, comunicano direttamente tra loro all'interno della stessa sottorete.
- **Off-link**: gli host hanno un *prefisso diverso*, comunicano attraverso un router.

A loro volta è possibile ulteriormente suddividere gli indirizzi di rete:

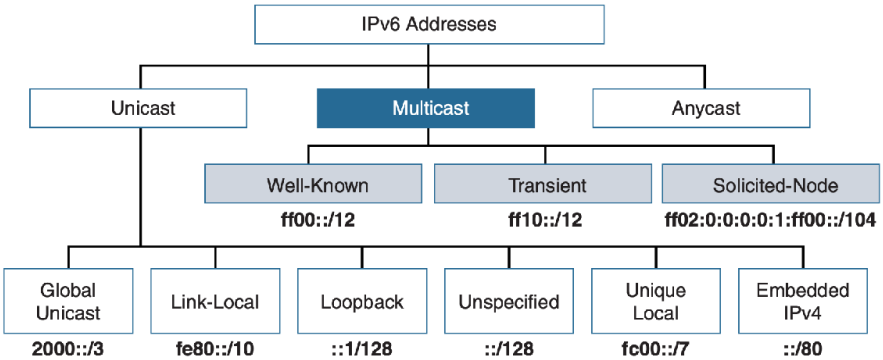


Figura 2.3: Spazio di indirizzamento

2.6 Multicast

L'equivalente dell'indirizzo multicast IPv4 '`224.0.0.0/4` è `FF00::/8`, che si suddivide in questo caso in:

- **Well-know Multicast**: `FF00::/12`, comunicazioni di servizio assegnati a gruppi di dispositivi e sono riservati. Un esempio è l'indirizzo di google.
- **Transient**: `FF10::/12`, indirizzi transitori, assegnati dinamicamente da applicativi multicast (corrispettivo della vecchia modalità multicast in IPv4).
- **Solicited-node Multicast**: `FF02:0:0:0:0:1:FF00::/104`, simile a un indirizzo IP broadcast in ARP.

Una caratteristica importante è notare come in IPv6 scompaia l'utilizzo del broadcast, che in seguito alle evoluzioni ha dimostrato essere un rischio per la sicurezza.

L'indirizzo si scompone in:

- **8 bit** iniziali, identificano che è un indirizzo multicast.
- **4 bit** per il **T flag**, dice se è well known (permanente o non permanente), viene assegnato da IANA.
- **4 bit** per lo scopo, viene lasciato ai dispositivi.
- **112 bit** per il group ID.

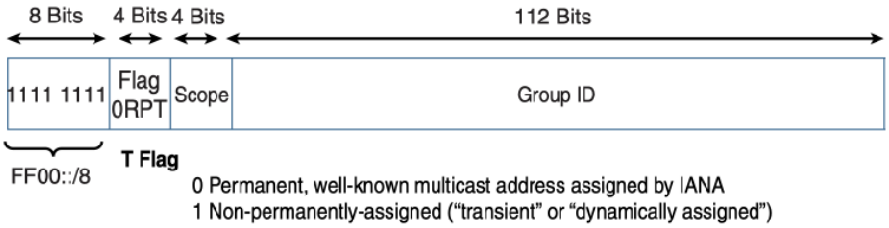


Figura 2.4: Struttura indirizzo multicast

2.7 Unicast

In IPv6 continuano a essere disponibili gli indirizzi unicast, con i seguenti indirizzi:

- `2000::/3` Global Unicast
- `FE80::/10`, Link-Local
- `::1/128`, Loopback (in IPv4 era `0.0.0.0`)
- `::/128`, Unspecified

- `FC00::/7`, Unique Local
- `::80`, Embedded IPv4

2.7.1 Global Unicast Addresses

Sono indirizzi di tipo aggregato, che andiamo a utilizzare in modo equivalente agli indirizzi pubblico IPv4. E' globalmente raggiungibile e indirizzabile ed ha la caratteristica di essere plug and play. Attualmente sono disponibili in un range definito tra `3FFF::` e `2000::`. Questi indirizzi hanno i primi 3 bit posti a `001`.

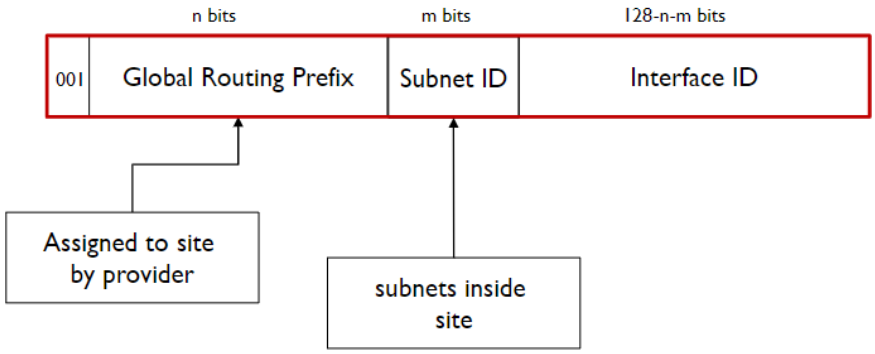


Figura 2.5: Global Unicast Addresses

I prefissi per il Global Routing sono formalmente assegnati da multi-level authorities:

- **3 bit**, tipologia (`001`).
- **13 bit**, TLA ID *(Top Level Authority, grandi ISP)*
- **32 bit**, NLA ID *(Next-level Authority, organizzazioni)*
- **16 bit**, SLA ID
- **64 bit**, Interface ID

2.7.2 Link local/site local Addresses

i link local/site local sono un gruppo di indirizzi che iniziano con `FEBF`, sono assegnati in automatico ai link quando viene acceso un router.

Gli indirizzi Link local vengono assegnati quando più router devono parlare tra di loro oppure devono annunciarsi a un router vicino.

Gli indirizzi site local sono nella rete `FEC0::/10`, sono ormai ritenuti deprecati perché pensati come vecchi indirizzi privati riconfigurabili, possono avere assegnati i router nelle comunicazioni (tipo stella e mesh ecc..). Utilizzano comunicazioni dirette e possono essere assegnati sono a indirizzi di rete.

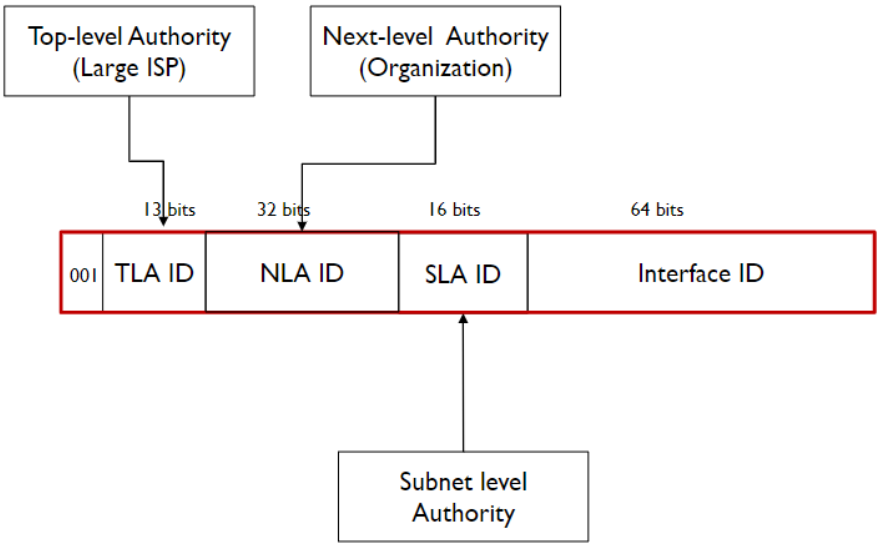


Figura 2.6: Global Routing Prefix

2.7.3 Unique Local Addresses

Gli Unique Local Addresses possono essere utilizzati in modo simile agli indirizzi globali unicast, ma sono per un utilizzo privato e non per l'indirizzamento sull'internet. Sono identificati da `FFC0::/7`, e vengono utilizzati dai dispositivi che non hanno mai necessità di connettersi all'internet e non hanno bisogno di essere raggiungibili dall'esterno. Sono indirizzi privati che possono comunicare su internet grazie ad operazioni di tunneling.

L'ottavo bit è il *Local (L) Flag*, che divide in:

- `FC00::/8`, se L flag è `0`, verrà assegnato in futuro
- `FD00::/8`, se L flag è `1`, l'indirizzo è assegnato localmente

Attualmente gli indirizzi `FD00::/8` sono gli unici indirizzi validi. Sono dunque privati e non utilizzati da altri dispositivi.

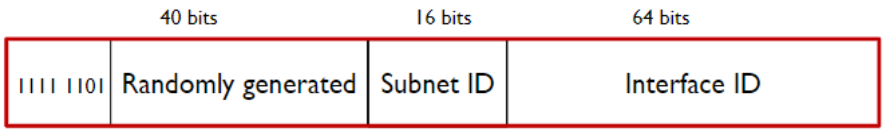


Figura 2.7: Unique Local Addresses

Dopo i primi 8 bit, sono presenti 40 bit generati casualmente in modo da non avere collisioni con altri indirizzi.

2.7.4 IPv4 Embedded Addresses

Gli IPv4 embedded addresses sono utilizzati per rappresentare indirizzi IPv4 all'interno di un indirizzo IPv6. Vengono utilizzati per facilitare la transizione tra i due protocolli. L'indirizzo IPv4 è inserito negli ultimi 32 bit (low order) mentre i primi 80 devono necessariamente essere pari a `0`, a cui seguono 16 bit dal valore di `FFFF` (16 1).

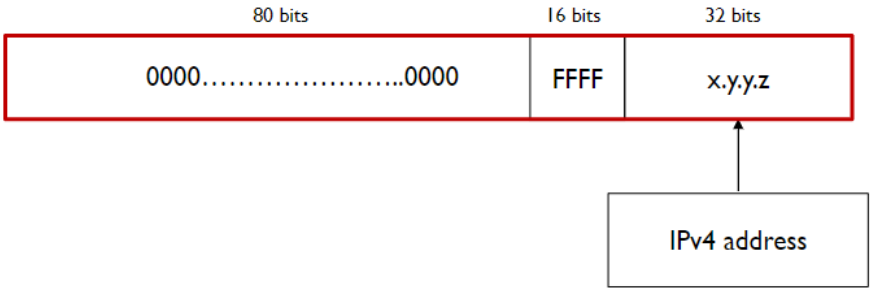


Figura 2.8: Struttura indirizzi IPv4 Embedded

2.8 Anycast Addresses

Gli indirizzi anycast possono essere assegnati a più di una interfaccia (tipicamente su dispositivi differenti), dando dunque la possibilità di avere su dispositivi differenti lo stesso indirizzo anycast. Un pacchetto che viene inviato a un indirizzo anycast viene reindirizzato all'interfaccia più vicina avente quel indirizzo. Questo permette di avere un indirizzo unico per un servizio, ma che può essere raggiunto da più dispositivi. Inizialmente venne realizzato per il DNS, ma è ancora in uno stato sperimentale.

> **Nota**: molto utile, ma non è ancora utilizzato.

2.9 Architettura del protocollo

L'architettura del protocollo IPv6 è molto simile a quella di IPv4, ma presenta alcune differenze:

- **IP**: utilizzato, salvo alcune modifiche
- **ICMP**: viene utilizzato *ICMPv6*
- **ARP**: non più utilizzato, inglobato in *ICMPv6*
- **IGMP**: non più utilizzato, inglobato in *ICMPv6*

Attenzione: non è più possibile utilizzare *ARP* E *IGMP* per risolvere gli indirizzi IPv6.

Sono invece stati aggiornati senza modifiche essenziali:

- DNS (type AAAA record)
- RIP e OSPF
- BGP e IDRP
- TCP e UDP
- Socket interface

2.10 Packet Header Format

L'header è stato modificato in modo sostanziale in seguito all'introduzione del IPv6. Ciò è stato fatto al fine di avere un header il più snello possibile, ottenendo una lunghezza di **40 byte**.

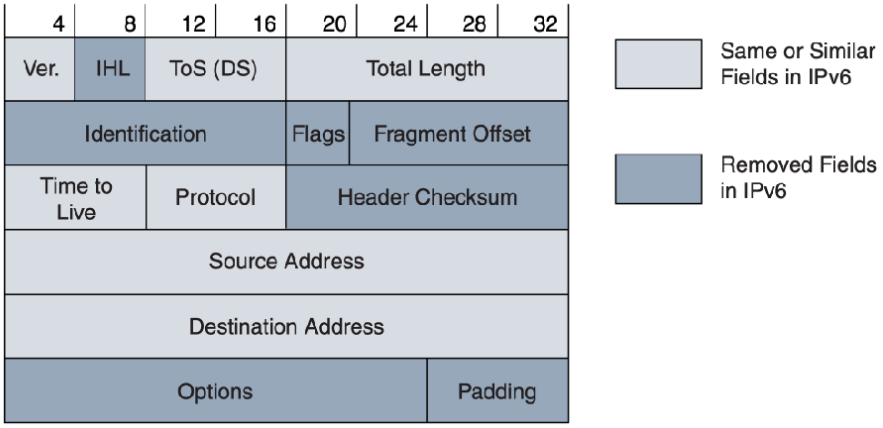


Figura 2.9: Header IPv4

L'header utilizzato in IPv6 è invece il seguente:

Osservando le immagini si può notare come alcune informazioni siano stati rimossi:

- Header Checksum: viene utilizzato per verificare se il dato trasmesso è corrotto, ma non è più necessario in IPv6.
  - Redundant: Layer 2 data link technologies perform own checksum and error control.
  - Upper-layer protocols such as TCP and UDP have their own checksums
- Frammentazione

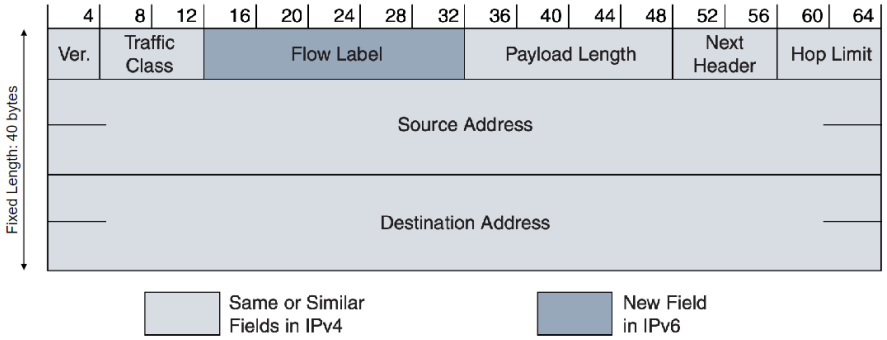


Figura 2.10: Header IPv6

- IPv6 routers do not fragment a packet unless they are the source of the packet
- Packets larger than MTU are dropped and an ICMPv6 Packet Too Big message is returned to source

Nota: Il checksum su UDP diventa opzionale in IPv6.

L'header può essere ulteriormente esteso attraverso il campo next header, che consente di puntare a un altro header contenente ulteriori informazioni creando una catena di header. Funzionano in modo simile al campo "protocol" di IPv4.

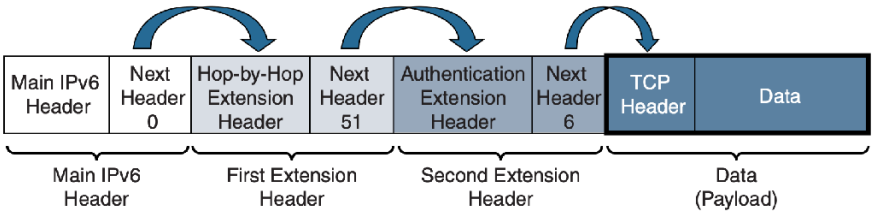


Figura 2.11: Chaining

Inoltre, sono presenti:

- **version**: versione del protocollo
- **traffic class**: permette di indicare la priorità del traffico (quality of service)
- **flow label**: permette di indicare il flusso di dati (nuovo campo), permette di associare un'etichetta a un certo tipo di traffico (label routing). Un esempio è se non mi fido dei miei dipendenti e voglio che tutto il loro traffico passi per un dispositivo di sicurezza che lo analizzi.
- **payload length**: lunghezza del payload
- **hop limit**: numero di router che possono essere attraversati prima che il pacchetto venga scartato. Se il valore è 0, il pacchetto viene scartato. Se il valore è 1, il pacchetto viene inviato al destinatario

senza essere inoltrato. Se il valore è 255, il pacchetto non viene scartato mai.

Nota: Header length non serve più! Viene eseguita la frammentazione attraverso il next header.

Il formato del campo next header è il seguente:

- **next header**: indica il tipo di header successivo
- **length**: lunghezza del header successivo
- **extension header**: header successivo
- **extension data**: dati dell'header successivo

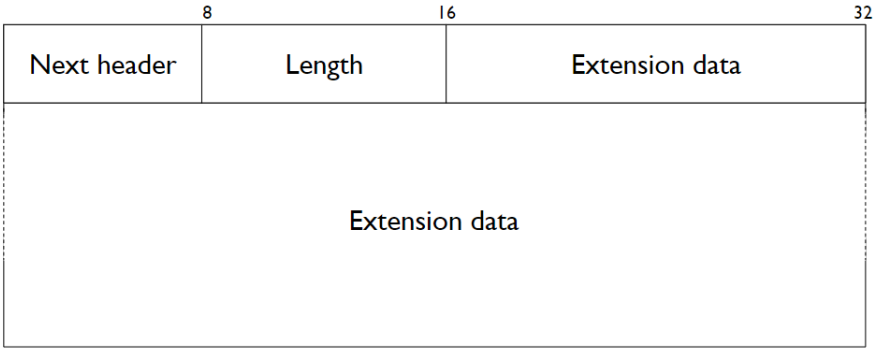


Figura 2.12: Extension Header Format

2.10.1 Hop-by-Hop Extension Header

E' utilizzato per andare a inserire dei campi/vincoli che servono all'hop per capire se il pacchetto deve essere scartato o meno (strumento di analisi). Se è presente, è indicato immediatamente dopo l'header IPv6. Questo header viene utilizzato per inserire dei campi opzionali. Ogni opzione ha un set di:

- **option type**: indica il tipo di opzione
- **option length**: lunghezza dell'opzione
- **option value**: valore dell'opzione

Si ottiene una tripletta **TLV** (type-length-value).

2.10.2 Routing Extension Header

IL routing extension header permette alla sorgente di un pacchetto di specificare il percorso di destinazione, indicando uno o più router intermedi. Viene utilizzato per il supporto alla mobilità in

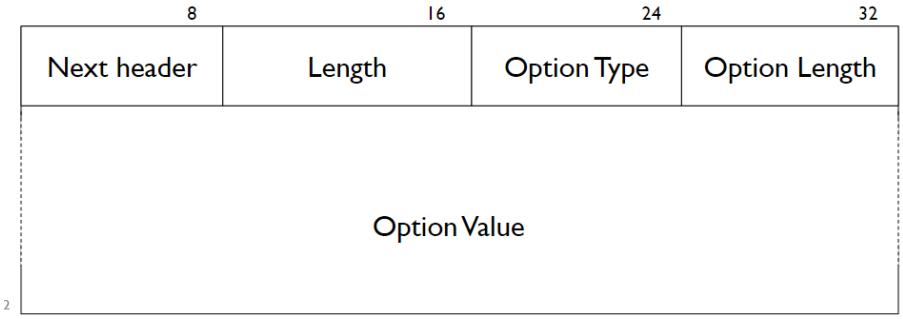


Figura 2.13: Hop-by-Hop Extension Header

IPv6.

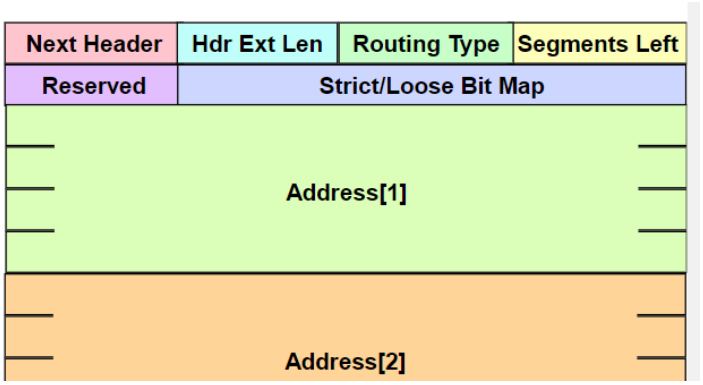


Figura 2.14: Routing Extension Header

2.10.3 Altre estensioni

Sono possibili altri due tipi di estensioni a seconda delle necessità.

2.10.3.1 fragmentation header

Viene utilizzato per la frammentazione dei pacchetti ognuno dei quali ha un proprio header IPv6 e un frammento di extension header. Il ricevente del pacchetto deve riunire i frammenti in un unico pacchetto. A differenza di IPv4, il protocollo IPv6 non frammenta un pacchetto almeno che non sia la sorgente del pacchetto.

2.10.3.2 Authentication and Encapsulation Header

Viene utilizzato per la sicurezza, adoperato da IPsec e fornisce una suite di protocolli per l'invio in sicurezza dei pacchetti in una rete IP. Il Authentication Header (AH) è utilizzato per l'autenticità e la integrità dei pacchetti. Il Encapsulating Security Payload (ESP) è utilizzato per la cifratura, autenticazione e integrità dei pacchetti.

2.11 Interfacciarsi con i livelli più bassi

2.11.1 Incapsulamento

La prima cosa che risulta evidente appena vi si approccia è che lo stack iso/osi prevede un campo in cui viene specificato il contenuto del livello superiore. Questo approccio è detto **dual stack**: creando uno nuovo stack è possibile far funzionare sia i dispositivi in IPv4 chee in IPv6 (lo trattiamo come un nuovo protocollo), senza alterare il funzionamento in IPv4.

I pacchetti IPv6 sono incapsulati nel frame di livello 2, ad esempio per ethernet il tipo è 86DD.

2.11.2 Address mapping

Un indirizzo di un pacchetto IPv6 viene associato a un MAC di destinazione attraverso:

- **IP unicast address**: discovery procedurale (protocol based)
- **IP multicast address**: algorithm mapping

2.11.3 IPv6 Multicast transmission

La trasmissione Multicast si basa sul ethernet multicast, ma a differenza del ethernet broadcast, un ethernet multicast può essere filtrato dalla scheda di rete (NIC).

Gli indirizzi multicast IPv6 vengono mappati su indirizzi MAC, in particolare è riservato l'indirizzo MAC Ethernet 33-33-xx-xx-xx-xx per il trasporto di pacchetti multicast IPv6.

Un esempio può essere il seguente: quando viene inviato un pacchetto all'indirizzo IP multicat FF0C::89:AABB:CCDD, questo viene incapsulato in un MAC frame con indirizzo 33:33:AA:BB:CC:DD.

> **Nota**: abbiamo FF all'inizio dell'indirizzo proprio perchè è multicast.

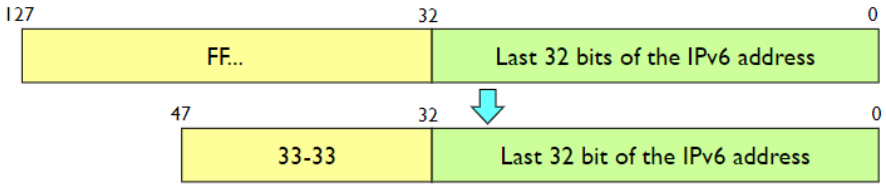


Figura 2.15: Multicast Transmission

2.12 Neighbor Discovery and Address Resolution

ICMPv6 adesso sostituisce completamente il protocollo **ARP**. E' basato su multicast e sfrutta il Solicicited-NOde MUlticast Address. A causa di come il multicast solicited address è realizzato, per lo più solo un nodo viene coinvolto.

2.12.1 Solicited-Node Multicast Address

Gli indirizzi vengono automaticamente creati per ogni indirizzo unicast dell'interfaccia. Tutti gli host si iscrivono e vengono mappati nel seguente modo: `FF:02::1:FF/104 | 24 ip meno significativi` (per lo più un host per gruppo).

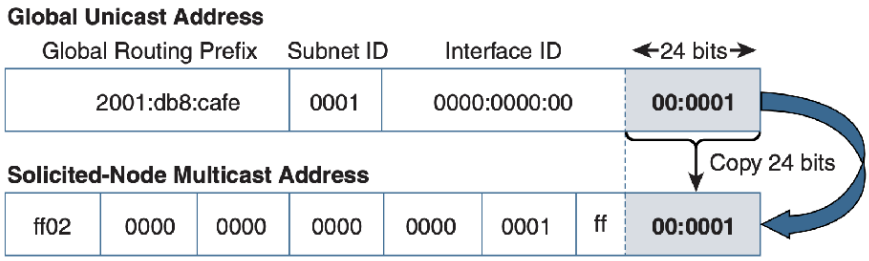


Figura 2.16: Mappatura indirizzo

Risoluzione indirizzo

La risoluzione di un indirizzo avviene attraverso **ICMP Neighbor Solicitation**: Il richiedente invia un frame al Solicited Node Multicast Address dell'indirizzo target IPv6.

:::tip **Come ricordarlo**: Il funzionamento è analogo al seguente: non lo chiedo a tutti, ma soltanto a chi mi potrebbe rispondere. :::

Avviene in seguito la risposta **ICMP Neighbor Advertisement**, attraverso la quale viene inviata la risposta indietro all'indirizzo unicast del richiedente. La mappatura tra IPv6 e MAC address viene memorizzata nella cache dell'host (in modo equivalente alla cache ARP).

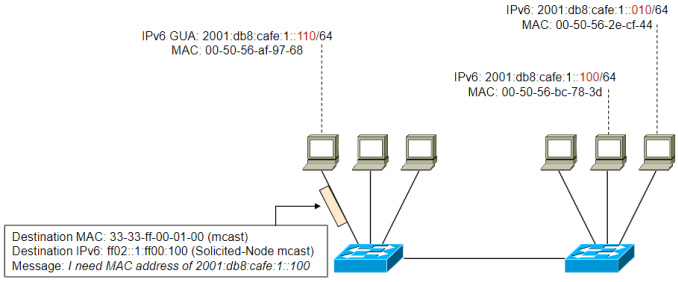


Figura 2.17: Risoluzione dell'indirizzo

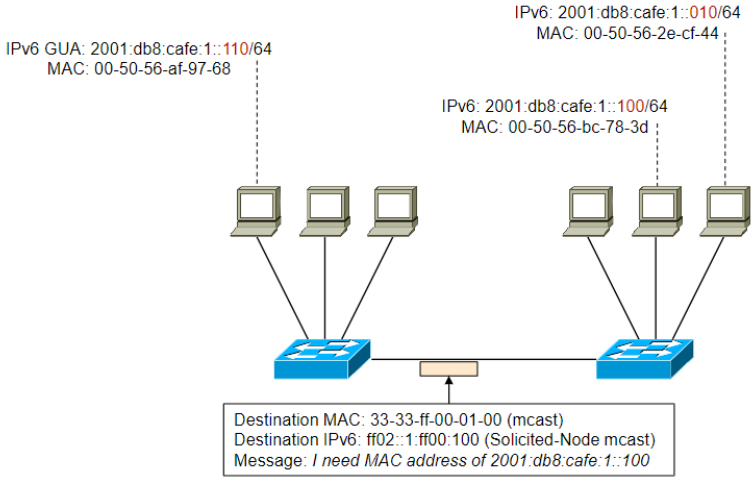


Figura 2.18: Risoluzione dell'indirizzo

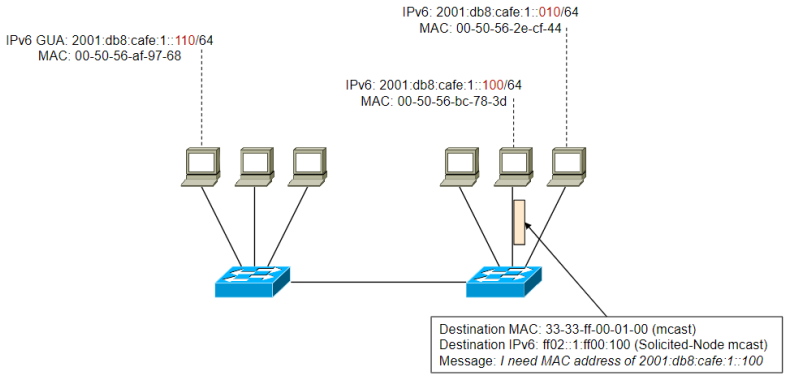


Figura 2.19: Risoluzione dell'indirizzo

Di fatto il numero di MAC aumenta molto, a causa della mancanza degli indirizzi broadcast. Per questo motivo è necessario che il router sia in grado di rispondere alle richieste di risoluzione indirizzo.

2.13 La transizione tra IPv4 e IPv6

La transizione da IPv4 a IPv6 sta venendo in modo **incrementale**, non è stato stabilito un limite entro cui eseguire il passaggio ma bensì sarà stabilito automaticamente quando sarà, nel pratico, il più utilizzato. Questo approccio trasparente e graduale ha consentito che prima di far prendere piede IPv6 nel corso di molto tempo ma in modo **seamless** (ovvero senza cambiamenti). Inoltre, come già accennato, è possibile generare e ricevere pacchetti per entrambi i protocolli senza problemi grazie all'approccio **dual stack**.

Questo risultato viene ottenuto attraverso tre meccanismi:

- Address Mapping
- Tunneling
- Translation mechanisms

Quando è nato IPv6 erano presenti poche reti dual stack, quindi era presente una parte di backbone su ipv4.

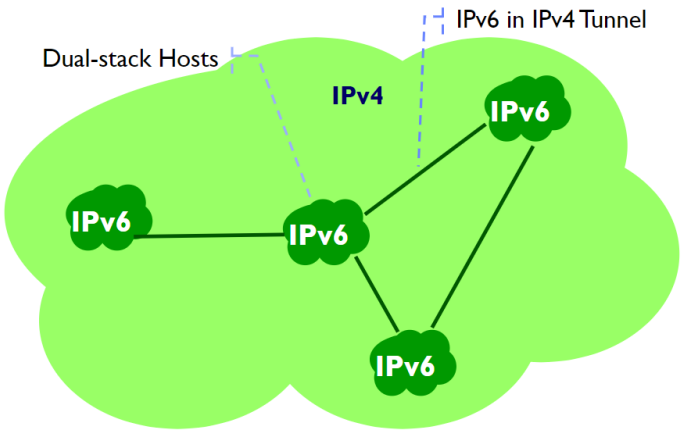


Figura 2.20: Pochi host IPv6

Nel corso del tempo le infrastrutture si sono adeguate al passaggio, aumentando il numero di host con comunicazioni onlink.

L'obbiettivo è quello di riuscire a creare una rete maggioritaria su IPv4 con solo poche connessioni IPv4. In realtà abbiamo già le infrastrutture per eseguire il passaggio completo.

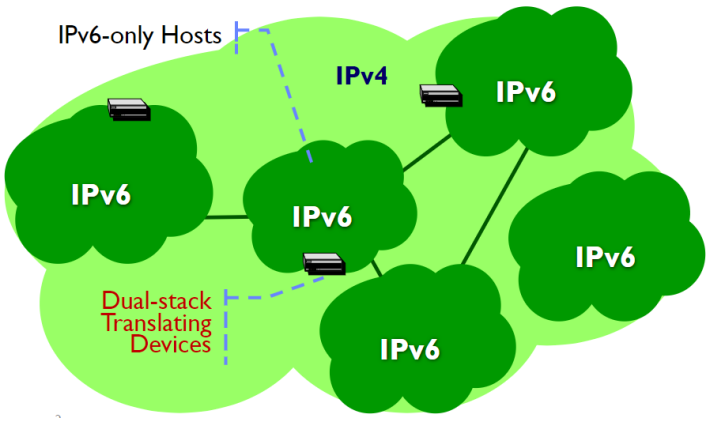


Figura 2.21: Molti host IPv6

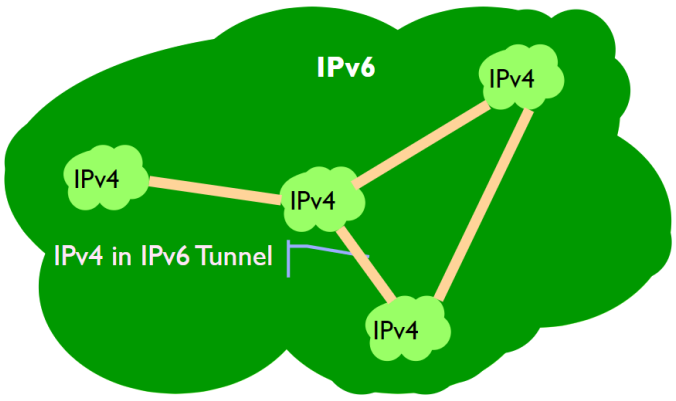


Figura 2.22: Maggioranza IPv6

2.14 ICMPv6

ICMPv6 permette di eseguire operazioni di:

- diagnostica
- neighbor discovery
- Multicast group management
- issue notification

Inoltre, include alcune funzioni che in IPv4 erano delegate ad **ARP** (Address Resolution Protocol) e **IGMP** (Internet Group Membership Protocol).

2.14.1 Formato del messaggio

Il messaggio è incapsulato nei pacchetti IPv6 con `next header = 58`, che mi permette di identificare il nuovo header di tipo **ICPMv6**, che avrà al più **576 byte**.

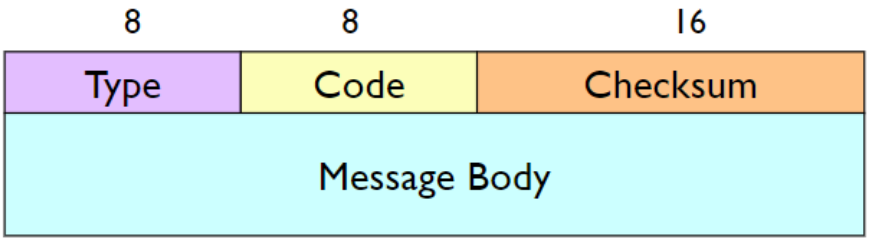


Figura 2.23: Formato del messaggio

Code	Spiegazione	tipo
1	Destination Unreachable	Errore
2	Packet too big	Errore
3	Time exceeded	Errore
4	Parameter Problem	Errore
128	Echo Request	Informativo
129	Echo Reply	Informativo
130	Multicast Listener Query	Informativo
131	Multicast Listener Report	Informativo

Code	Spiegazione	tipo
132	Multicast Listener Done	Informativo
133	Router Solicitation	Informativo
134	Router Advertisement	Informativo
135	Neighbor Solicitation	Informativo
136	Neighbor Advertisement	Informativo
137	Redirect	Informativo

2.14.2 Neighbor Solicitation

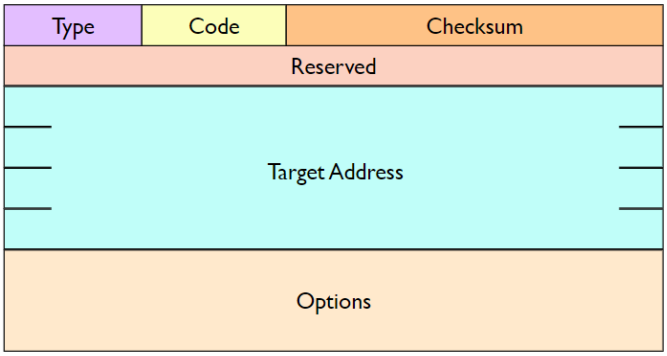


Figura 2.24: Neighbor Solicitation

2.14.3 Neighbor Advertisement

Sono presenti dei flag aggiuntivi:

- R **router flag**, se `true` arriva da un router.
- S **solicited flag**, se arriva da un nodo che ha fatto una richiesta di risoluzione.
- O **override flag**, se la host cache deve essere aggiornata o meno.

Nota: non è presente un campo MAC, in quanto può essere si da per scontato sia presente nelle opzioni. Viene invece specificato l'ip, anche se ridondante, in quanto potrebbe essere sia un nodo che un router.

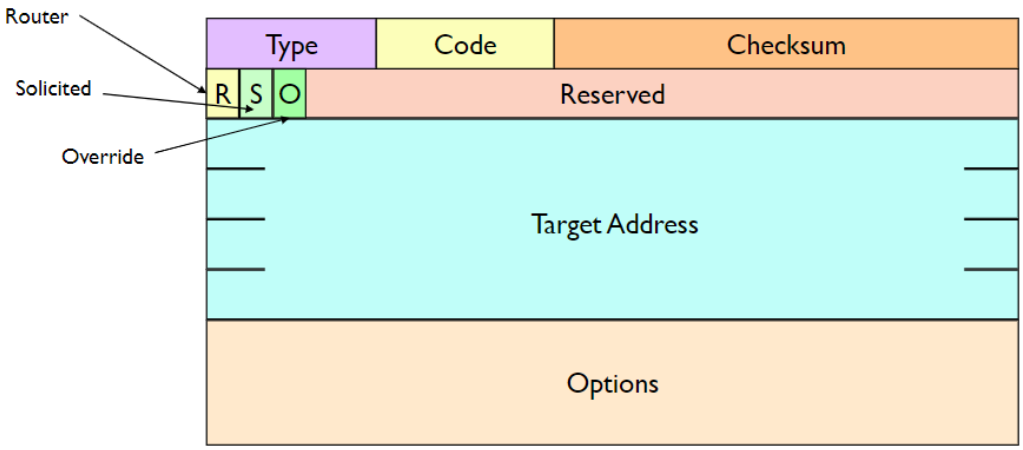


Figura 2.25: Neighbor Advertisment