

Abstract

Questo è un documento in cui si collezionano gli appunti sul protocollo IP tratto dal corso *Reti di Calcolatori e Principi di Cybersecurity*, che fa parte del programma della seconda parte del corso (non è tutta la seconda parte!)

Si sconsiglia quindi di usare questo materiale in alternativa alle lezioni o alle slide fornite dal docente A. Bartoli, in quanto i miei appunti potrebbero contenere refusi, errori di battitura, oppure potrebbero essere difficili da comprendere senza aver avuto prima un'idea dei contenuti, che sono ottenibili seguendo le lezioni.

Detto ciò, non mi assumo nessuna responsabilità del vostro rendimento in questo corso se decidete di basarvi su questi appunti.

Dino Meng

"Protocollo IP"

SEZIONE A. NETWORK

Fondamenti di Network

X

Fondamenti sul network. Definizione di Network: punto di vista "pratico" e punto di vista implementativo. Esempio di network tech: Ethernet, Bluetooth.

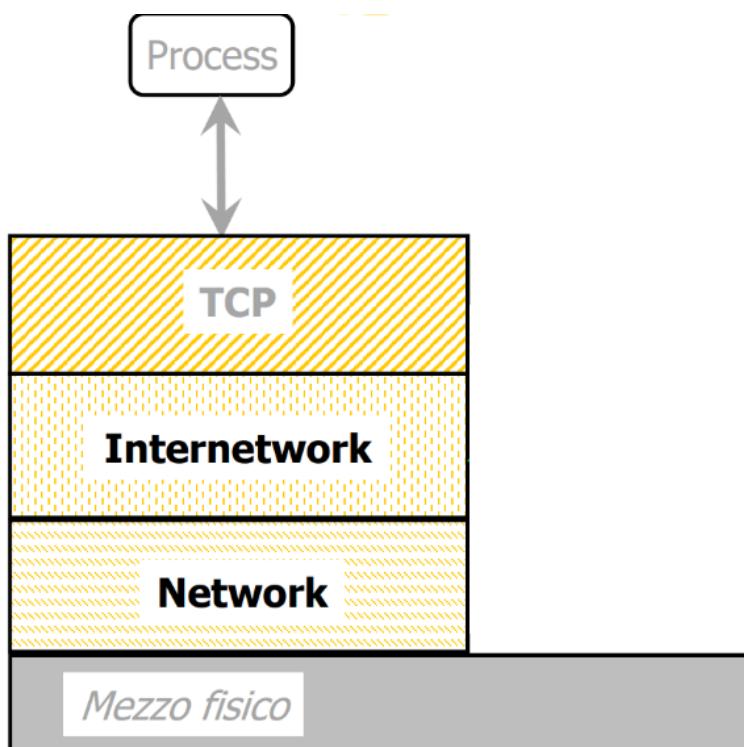
X

0. Voci correlate

- Comunicazione tra Processi
- Proprietà dei Servizi di Comunicazione

1. Fondamenti sul Network

Ricordiamo che per quanto riguarda la comunicazione di *processi*, a livello implementativo (approx.) essa avviene su *più livelli*. Fin'ora (ovvero fino a quanto avevamo trattato DNS, Web e Mail) abbiamo visto il livello più alto, ovvero livello *TCP/UDP* usato dai processi.



Adesso scendiamo di livello. Partiamo dal *network*, ovvero il "*primo*" (o l'ultimo, dipende dall'ordine) livello di comunicazione tra processi.

Network. Per parlare del *network*, distinguiamo due aspetti:

1. L'aspetto "*funzionale*", ovvero vogliamo descrivere il network dal punto di vista di "*internetwork*" (o superiore)
2. L'aspetto *implementativo*, ovvero "*come è fatto*".

"Come si usa": Nel network, abbiamo che ogni *nodo* (calcolatore) può comunicare con un *qualsiasi altro nodo* che abbia la stessa *network tech* (hardware + software).

Inoltre, ogni *collegamento* nodo possiede un *identificatore univoco*, l'indirizzo *network*. Essa identifica l'*interfaccia di rete* nel *network*.

Infine, ogni *network tech* possiede le stesse proprietà sui servizi di comunicazione: quindi il concetto di connectionless/connection oriented, reliability/unreliability e byte-oriented/message-oriented. Inoltre, vengono specificati altri aspetti tra cui *formato degli indirizzi network*, *massimo numero di nodi collegabili tra di loro*, *massima distanza fisica di collegamento* e *la massima dimensione dei messaggi da consegnare* (MTU) (se message oriented).

Gli aspetti di *max nodes* e *max distance* dipendono dalla tecnologia usata, il resto è stipulato mediante le convenzioni (lato sw).

Le network tech possono essere suddivise in due "*tipologie*", a seconda delle loro proprietà.

- *WAN* (Wide Area Network): Estensione geografica
- *LAN* (Local Area Network): Estensione limitata, ma latenza bassa

"Come è fatto": Al livello implementativo, dipende dalla network tech usata. Generalmente, i processi si scambiano "messaggi" mediante i *frame*, che sono composti da *payload* e *control information* (formato specificato da ogni network tech).

X

2. Esempi di Network Tech

ESEMPIO. *Ethernet* è una network tech con le seguenti caratteristiche:

- Connectionless
- Unreliable
- Message-oriented
- Ogni indirizzo ethernet ("MAC") è un numero naturale composto da 6 byte
- All'incirca si collegano 100 nodi massimo, con massima distanza di 100 metri
- MTU è di 1500 byte
- L'indirizzo FF:FF:FF:FF è riservato ai messaggi "*broadcast*", ovvero quelli che vengono mandati a tutti nel network (approfondiremo opportunamente)

Implementazione di ethernet. Reminder sulle proprietà di ethernet, formato dell'indirizzo ethernet (MAC). Formato di ethernet frame. Concetto di switch e access point. Funzionamento dei switch, switch table. Estensione delle switch: "fondare" network ethernet. Access point: funzionamento. Osservazioni.

0. Voci correlate

- Fondamenti su Network

1. Indirizzi Mac, Ethernet Frame e Proprietà Ethernet

Vediamo adesso l'implementazione di *ethernet*, ovvero cosa succede quando un nodo manda un messaggio all'altro.

Vediamo i primi aspetti implementativi dell'ethernet.

INDIRIZZI ETHERNET.

Partiamo dagli *indirizzi ethernet*, conosciuti anche come *indirizzi MAC*. Sono formati da *6 byte* e rappresentano univocamente una scheda ethernet ognuna, ad eccezione dell'indirizzo con tutti 1. Per quanto riguarda il formato, la rappresentiamo in *notazione esadecimale* (che è facilmente convertibile a partire da multipli di byte), e ogni "*due cifre*" vengono divise dal simbolo dei due punti .

L'unico indirizzo che non può essere usato è l'indirizzo broadcast FF:FF:FF:FF:FF:FF, ovvero l'indirizzo con tutti uno. Questo viene usato per instradare pacchetti che vengono *inviati a tutti* nel network (vedremo dopo nei dettagli perché e come).

Esempio. 11010001 00111000 11110111 11010001 00111000 11110111 diventa D1:38:F7:D1:38:F7 (basta suddividere ogni byte in due parti da 4 bit, e convertire ciascuna parte in esadecimale)

ETHERNET FRAME.

Ogni *messaggio* è il *payload di un frame Ethernet*. Ogni payload frame viene trasmessa in una maniera *seriale*, ovvero "*viaggiano un bit dopo l'altro, in sequenza*". Ci sono due mezzi di trasmissione di frame:

- Cavo (*wired*)
- Segnale radio (*wi-fi*)

Ognuno di questi tipi permettono una "velocità" diversa di trasmissione. Osserviamo che quindi in una network coesistono nodi con "velocità" diverse.

Per quanto riguarda il frame stesso, ricordiamo che è composto da *payload* e *control information*.

Preamble (8)	DST Address (6)	SRC Address (6)	Type (2)	Payload (<1500)	CRC (4)
-----------------	--------------------	--------------------	-------------	--------------------	------------

Gli aspetti da ricordare sono *DST address*, *SRC address*, *type* e *payload*. Gli altri sono dettagli da trascurare.

PROPRIETA' ETHERNET.

Come detto prima, ricordiamo le seguenti proprietà dell'ethernet e spieghiamo come si "adattino" in questo contesto.

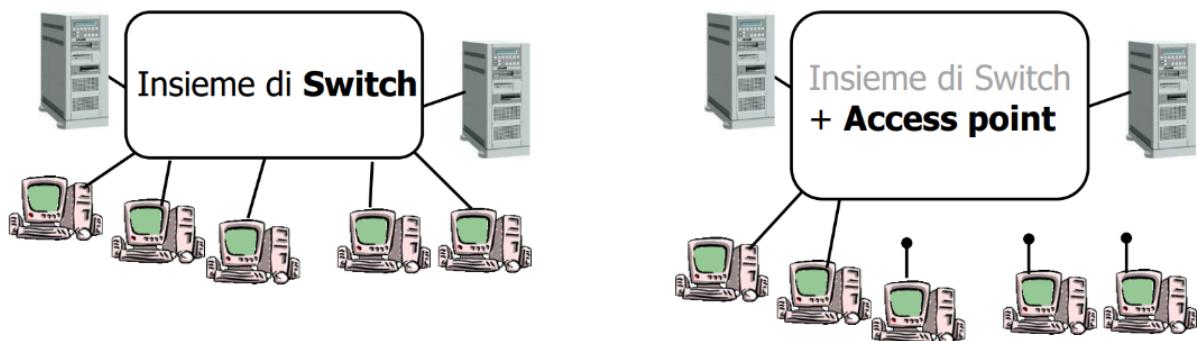
- *Message-oriented*: Ogni "messaggio" è un frame
- *Connectionless*: Non devo avere un "contesto" per mandare frame, e i frame sono indipendenti tra loro
- *Unreliable*: I nodi possono decidere di "scartare" i frame senza avvisare il mittente

X

2. Switch e Access Point

Q. Più nodi hanno velocità diverse, come si adattano? A cosa si collegano?

Essenziale i concetti di *switch* e *access point*, che andranno a definire un insieme di dispositivi collegati tra loro.



Gli *switch* servono per connettere dispositivi connessi via cavo, *access point* invece per dispositivi connessi via Wi-Fi.

Osserviamo che da un punto di vista funzionale dell'utente, non è necessario sapere *come* funziona l'architettura switch/AP per inviare messaggi. Al nodo basta solo l'indirizzo destinatario per inviare un messaggio.

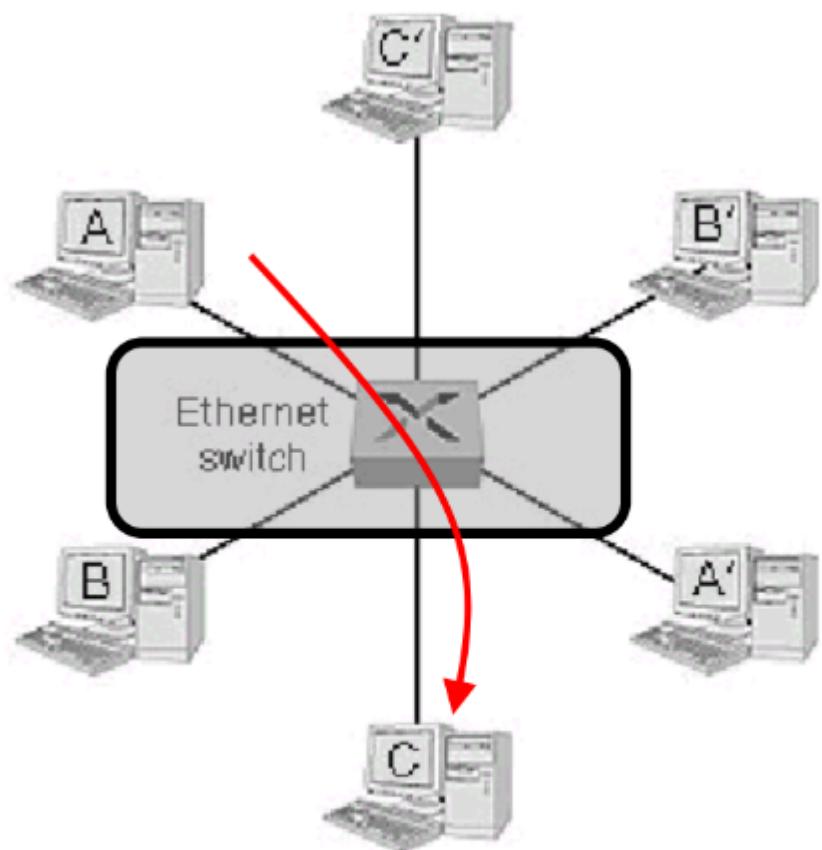
2.1. Switch

Uno *switch* è uno dispositivo composto da *porte*, a cui si possono collegare sia *nodi* che *altri switch*.



Uno switch, sostanzialmente, effettua le seguenti operazioni:

1. Ricevono un *frame* in bit (da una porta "*vedono*" i bit)
2. Analizzano l'*indirizzo destinatario*
3. Instradano il frame alla porta dell'*indirizzo destinatario* (molto velocemente!)



! Gli switch non hanno indirizzo ethernet, siccome sono completamente trasparenti dall'utente.

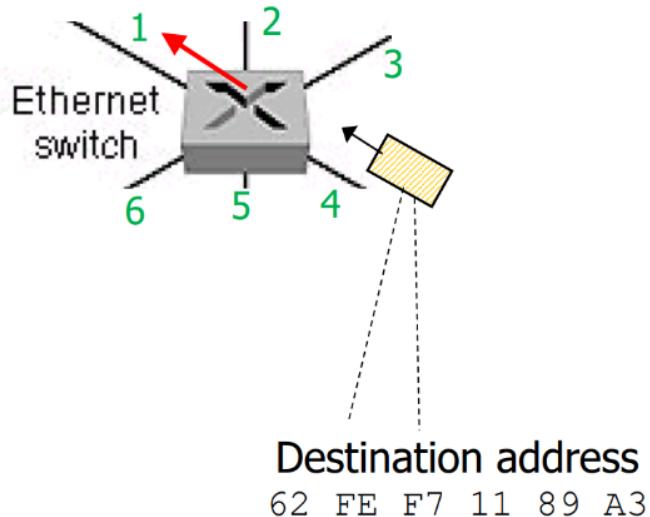
Inoltre, gli switch possono anche instradare più frame *simultaneamente* (come? dipende dalla loro qualità, diciamo...) e devono *gestire velocità di collegamento diverse* (quindi necessità di implementare funzionalità di FRAME QUEUEING).

SWITCH TABLE.

Q. Come fa il switch a conoscere la porta da instradare?

Ogni switch configura in sè automaticamente una *switch table*, ovvero una tabella di ricerca in cui ogni riga è una corrispondenza *port* ↔ *MAC address*.

La switch table viene inizializzata come una tabella vuota, e ogni volta che un nodo invia un *messaggio* lo switch ispezionerà il source address dei frame e salva (port, ETH-src) sulla tabella.



Address	port
01-12-23-34-45-56	2
62-FE-F7-11-89-A3	1
7C-BA-B2-B4-91-10	3
....

Approfondiamo lo *step 3.* del funzionamento dei switch appena visto (^54fd08).

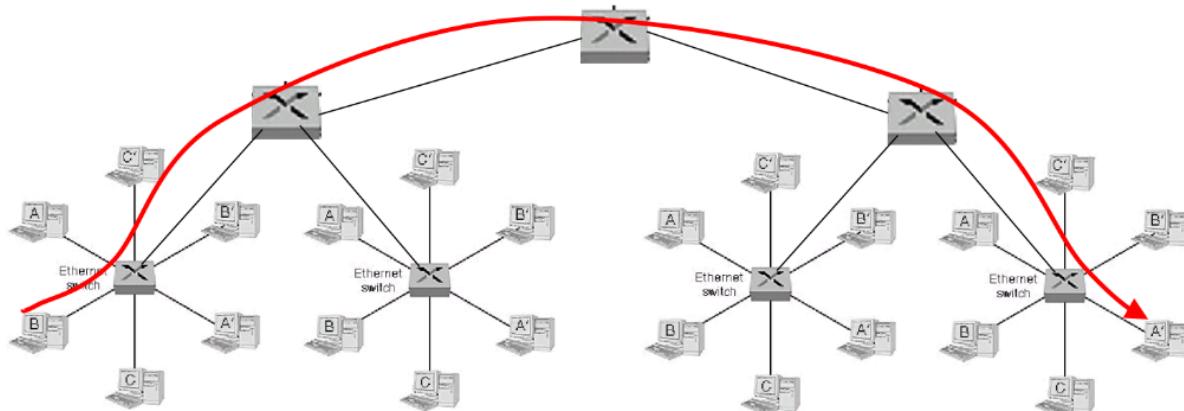
- Se lo switch non conosce la *port* di *ETH-dst*, può adoperare due strategie:
 - *Discard*: Buttare via il frame, va bene in quanto Ethernet è unreliable (e quindi fa parte del gioco)
 - *Overflow*: Inviare il frame a tutti (*ottimizzazione*)

Entrambe le strategie vanno bene. Per riempire la switch table, intuitivamente si ha che ogni volta che un nodo si collega alla network esso invia un *frame* per "farsi scoprire".

Q. Se un switch deve inoltrare un frame ad un indirizzo MAC che non è "*suo*" ma di un altro switch, come fa?

Basta estendere il concetto di switch table, ogni riga può contenere anche *più indirizzi ethernet*.

In questo modo, ogni volta che lo switch deve instradare un pacchetto ad un altro "*switch di competenza*" lontano, basta che instradi il messaggio a quel switch.

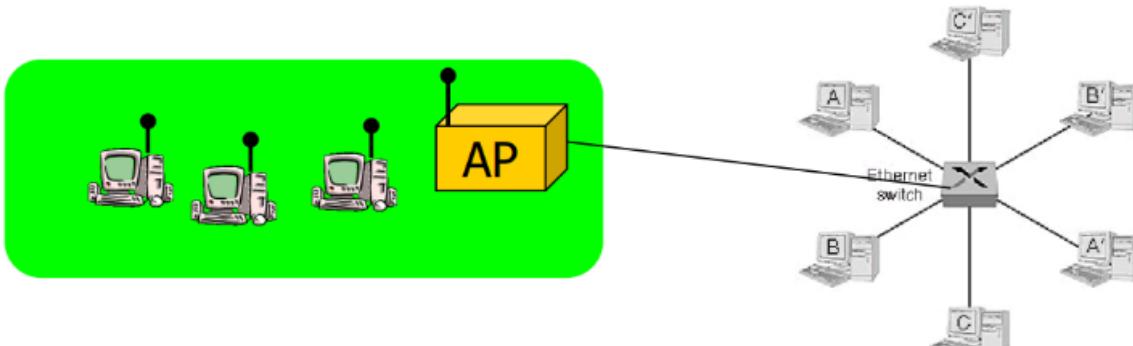


Qui notiamo una peculiarità degli switch: *è tutto automatico!*

2.2. Access Point

Gli *access point* sono dei dispositivi *a due lati*, dove uno *si collega ad uno switch* e l'altro *comunica con gli altri dispositivi "vicini"*. In ogni *cella* c'è un *access point*.

Gli *access point* sono equivalenti ad uno *switch* che non supporta comunicazioni simultanee



Curiosità. Nella realtà i pacchetti trasportati su *wifi* sono *"leggermente"* diversi da quelli trasportati via cavo. Non approfondiremo.

X

3. Ulteriori Osservazioni

Osserviamo che nel gergo tecnico, *network* può anche indicare *alcune parti di network* (che è anche *"sbagliato"*).

Nel nostro caso definiamo *network* nella seguente maniera precisa:

- A, B appartengono alla stessa *network* sse A può ricevere frame inviati da B

Osservazione. Nella realtà, quando compriamo uno *switch/AP* compriamo anche il *calcolatore in più* che serve per la configurazione del network. Per noi uno *switch/AP* è solo lo *switch/AP*

SEZIONE B. INTERNETWORK

Fondamenti di Internetwork

X

Aspetti motivanti dell'internetwork, terminologia Internet e Internetwork, proprietà e funzionalità. Implementazione dell'internetwork, pacchetti IP. Azioni del sender, receiver e del router. Domande da rispondere sull'internetwork (programma del capitolo).

X

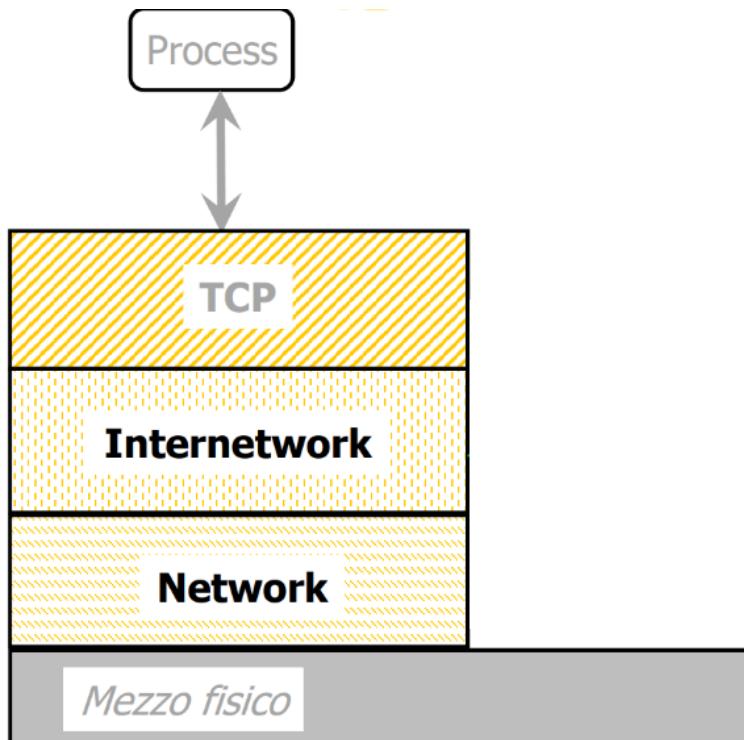
0. Voci correlate

- Fondamenti su Network

1. Introduzione all'Internetwork

PROBLEMA. Abbiamo moltissimi nodi con *tecnologie network diverse* o *lontane tra loro*, come possiamo integrarli in un'unica struttura di comunicazione? E se sono milioni di nodi? (nessuna network tech è in grado di integrarli in un'*unica* network...)

Il concetto di *internetwork* andrà a risolvere il problema appena posto, e andrà a rappresentare lo strato di astrazione "*sopra*" i network:



DEFINIZIONE. In particolare, definiamo l'*internetwork* come una *struttura logica* ("virtuale") dove ogni *nodo ha un identificatore unico* e può comunicare con *un qualsiasi altro nodo*.

OSSERVAZIONE. (*terminologia*)

Nel gergo tecnico, il termine "*internet*" può indicare due cose differenti; o intendiamo l'*internetwork*, o intendiamo *una particolare network*, che è quella usata tutti i giorni. Nel secondo caso, di solito si scrive *Internet* con la I maiuscola.

Ci sono molti "*modi*" per implementare l'internetwork, ma il nostro unico caso di interesse è l'*IP protocol*.

Proprietà. L'internetwork *IP* è un servizio di comunicazione che possiede le seguenti proprietà:

- Connectionless
- Unreliable
- Message-Oriented
- Indirizzi IP formati da 4 byte, con una certa "*struttura*" (vedremo dopo molto approfonditamente)
- MTU di 64 KB
- I nodi possono essere "*moltissimi*" e "*lontanissimi tra loro*" e possono usare network technologies diverse

X

2. Implementazione dell'Internetwork

Vediamo l'*internetwork* da un punto di vista implementativo. L'idea di base è quello di trovare un modo di collegare nodi di network diverse.

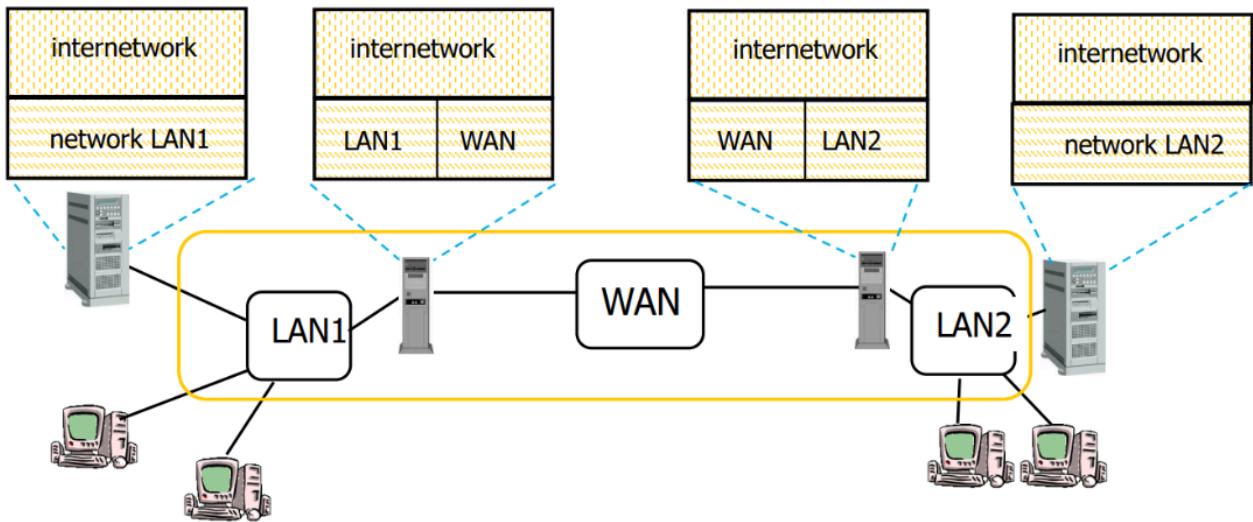
Indirizzi Nodi. Partiamo dall'*indirizzo dei nodi*. Ogni nodo in realtà possiede due indirizzi:

1. Indirizzo della *network* a cui appartiene, si dice *indirizzo fisico*
2. Indirizzo della *internetwork*, si dice *indirizzo logico*

Gli uni non hanno nulla a che fare con gli altri. Di solito, l'*indirizzo fisico* sarà solitamente l'indirizzo *ethernet*, invece l'indirizzo logico sarà l'*indirizzo IP*.

Router. Vediamo addesso come possiamo "*incollare le network*". Ci sono dei *nodi in più* per "*intradare*" e "*convertire*" pacchetti di network diverse, e si dicono *router*. Ogni router ha quindi più schede ed indirizzi network (!).

E' importante osservare che i router hanno *più schede network*, quindi più *indirizzi logici* e *fisici*!



Osservazione. Notiamo che quindi in ogni end point dobbiamo avere *software* che implementa l'internetwork.

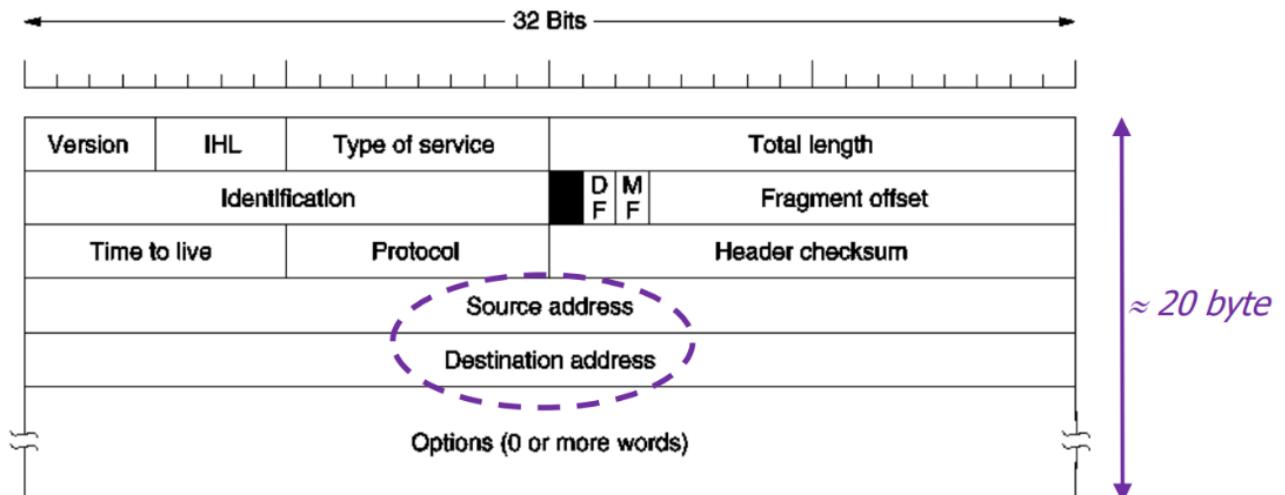
Osservazione. Solo lo *strato TCP/UDP* usa il protocollo IP, quindi ci troviamo in un livello basso di astrazione. Infatti, teniamo nota che le applicazioni *non* usano direttamente il protocollo IP.

Pacchetti. I "frame" (pacchetti) IP sono composti da due parti: *payload* e *header*.

Sinteticamente, possiamo descrivere ogni pacchetto come "*send m to IP-dst*" o "*received m from IP-src*". Gli IP header che ci interessano sono solo due, IP-dst e IP-src (gli altri sono header di controllo).

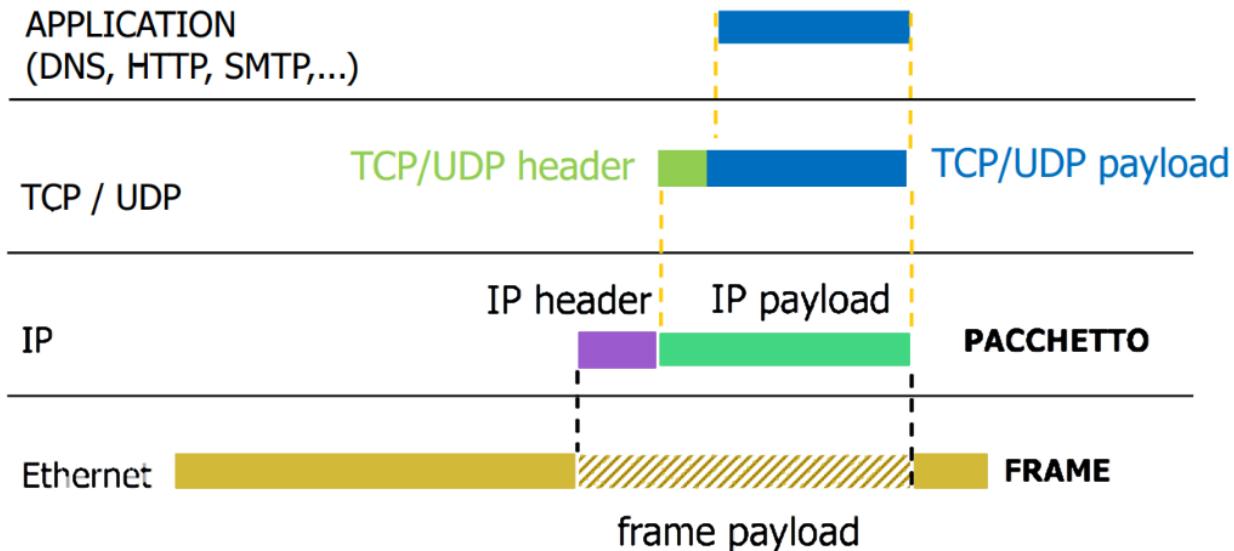
La parte "byte" è interessante, e indica come va interpretato il payload. In particolare è formato da due byte, e ogni numero indica un protocollo specifico. Nei protocolli che vedremo:

- 800 è IP
- 806 è ARP



Encapsulation. Per fare mente locale, vediamo lo scenario dell'*incapsulazione* dei pacchetti, ovvero quando dal lato applicativo vogliamo inviare un messaggio, per ogni strato che *"mandiamo giù"*, il messaggio diventa il payload di un altro frame, con ulteriori informazioni di contesto.

Esempio: HTTP request -> TCP/UDP frame -> IP packet -> Ethernet Frame



Esercizio. Scrivere il pseudocodice di un programma che invia una richiesta HTTP per prelevare un certo documento in URL-x. Riferendosi a questo programma, supponiamo di eseguirlo e tracciare tutto il traffico in un unico file. Cosa vediamo nel file?

Traccia: Contare prima le DNS request per risolvere URL-x (o proxy se facciamo ulteriori ipotesi), poi l'apertura della connessione TCP e infine le richieste/risposte HTTP opportune.

X

3. Azioni Sender, Receiver e Router

Vediamo addesso *come* vengono mandati i pacchetti, dal punto di vista dei nodi. Partiamo innanzitutto dalla *configurazione* dei calcolatori.

Ogni nodo deve avere in configurazione le seguenti informazioni:

1. *IP Address* del nodo mittente
2. ... (vedremo dopo)
3. *IP-Gateway*: Default Gateway (*router collegato alla stessa network del nodo*)
4. *IP-Name Server* (questa non è indispensabile per la sola connettività IP, ma a fine pratici è necessario avere in configurazione)

Adesso siamo pronti per descrivere le azioni in dettaglio

Sender. Il sender effettua le seguenti operazioni, descritte dal seguente pseudocodice:

- IF IP-dst "si trova sulla mia network" (!):
 - Pacchetto è payload di un frame ethernet indirizzato all'indirizzo fisico di IP-dst (!!)
- ELSE:
 - Pacchetto è payload di un frame ethernet indirizzato all'indirizzo fisico di IP-Gateway (!!)

Receiver. Deve decifrare il *pacchetto IP*; deve quindi conoscere il *protocollo IP* (implementato con del software) per "*decifrare*" il payload

Router. Anche qui pseudocodice:

- IF IP-dst != my_IP:
 - "Devo instradare pacchetto", ossia
 - IF IP-dst "si trova su una delle mie network" (!):
 - Pacchetto è il payload di un frame indirizzato all'indirizzo fisico di IP-dst (!!)
 - ELSE:
 - Sia RX il *router per arrivare a IP-dst* (!!)
 - Pacchetto è il payload di un frame indirizzato all'indirizzo fisico di RX (!!)

Notiamo che nei procedimenti descritti, ci sono un sacco di operazioni descritte "*idealmente*", senza aver idea di come vengono effettuate. Sono segnati con i punti esclamativi, e le operazioni da approfondire meglio sono le seguenti:

- *Ottenere le informazioni di configurazione*
- *Avere un test per dire se due indirizzi IP si trovano sulla stessa network o meno* (!)
- *Risalire a indirizzi fisici dagli indirizzi IP* (!!)
- *Trovare un percorso "corretto" che instradi un pacchetto* (!!)

Per risolvere i problemi, vedremo gli seguenti argomenti:

- Configurazione statica/dinamica (DHCP)
- Network number, subnet mask
- Protocollo di risoluzione ARP
- Routing

In realtà, sarebbe da farsi una domanda aggiuntiva: "*Cosa fare se un pacchetto è troppo grande per un frame?*"; la risposta viene data dal concetto di *frammentazione*, che non vedremo.

Osservazione. In ogni nodo di una internetwork, esso comunque conosce *solo* la network a cui si collega. Infatti, per mandare pacchetti non sa niente dell'internetwork! In un certo senso, ha l'illusione di comunicare con tutti mediante *l'ethernet*.

Modello Fisico dell'Internet

X

Modello fisico (approssimato) dell'Internet. Modi di collegare endpoint. Router ISP, collegamenti locali (Internet Exchange Point). Osservazioni pratici: dispositivi con più schede network, throughput nominale. Cenni storici alla topologia dell'Internet. Notazioni per disegnare reti. Osservazione: l'inaffidabilità dell'internet.

X

0. Voci correlate

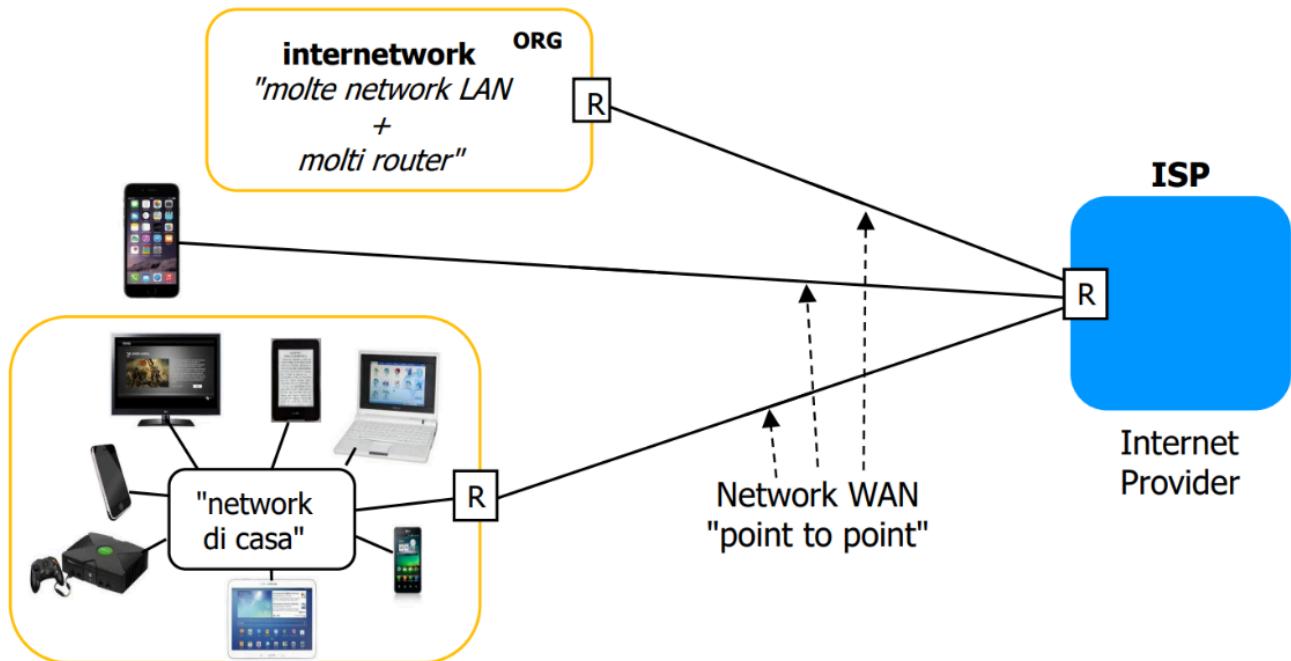
- Fondamenti su Internetwork

1. Modello Approssimato dell'Internet e Cenni Storici

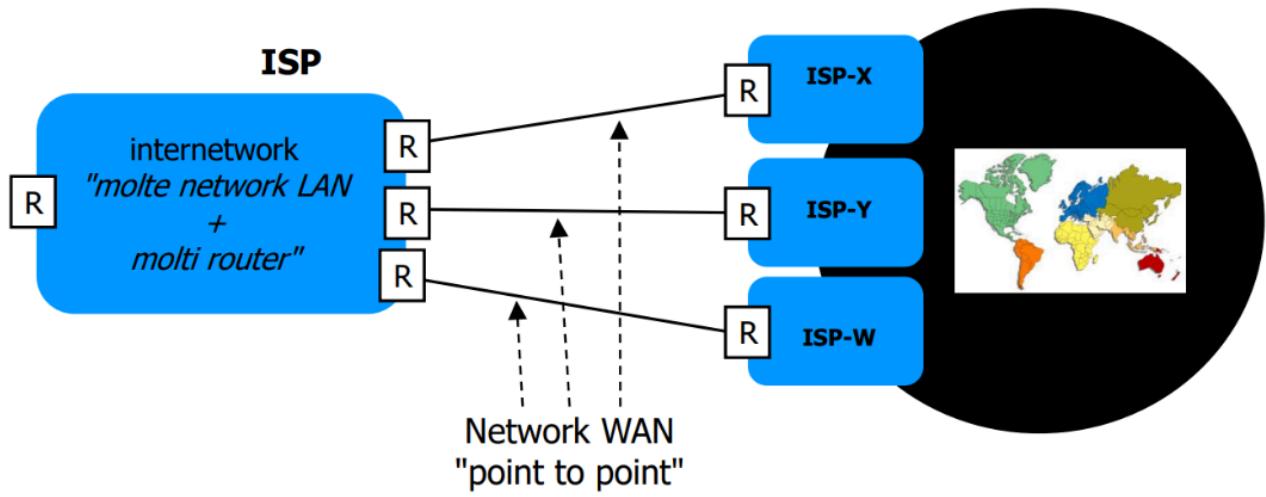
Descriviamo un *modello fisico approssimativo dell'Internet*.

Gli endpoint si collegano ad un "*router ISP*" (vedremo dopo cos'è) in tre modi:

1. In maniera *diretta*, mediante una network WAN "point-to-point"
2. *Network Ethernet* in cui collegiamo più dispositivi ad un *router di frontiera*, che poi si collega al router ISP
3. *Internetwork*, quindi più router e poi *uno di frontiera*



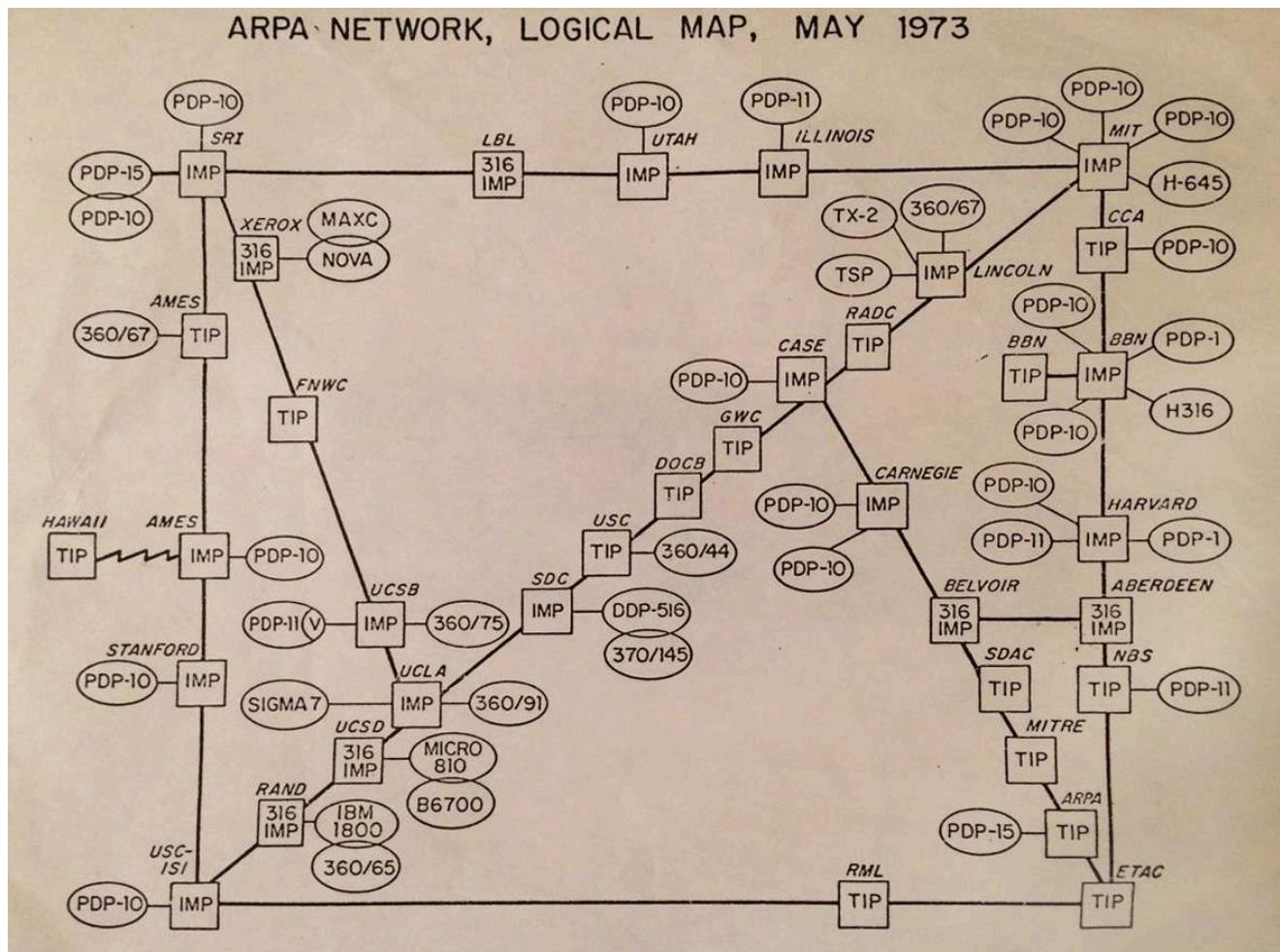
Dopo, nello "*strato successivo*" abbiamo che ogni ISP si collega ad altri ISP mediante *collegamenti diretti in data center dedicati*, chiamati *Internet Exchange Point*.



Osservazione. Molti dispositivi hanno *più interfacce network*. Non possono essere simultaneamente attive. (esempio: cellulare -> WiFi e dati mobile)

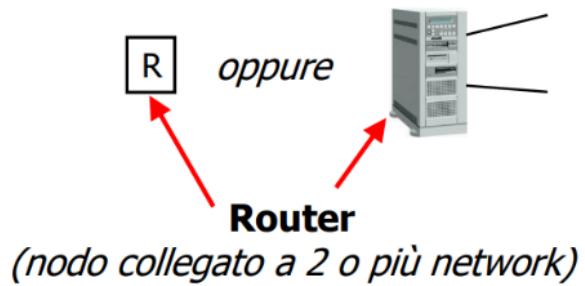
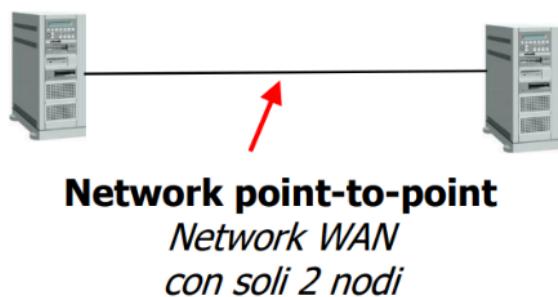
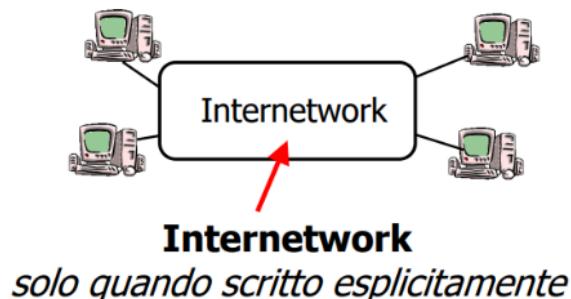
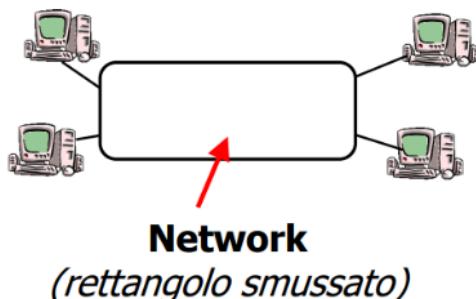
Osservazione. I contratti con gli *ISP* indicano un *throughput nominale* delle network a cui si collegano, che è (circa) la massima velocità di trasferimento dei collegamenti. Solitamente è *assimmetrico*, dove il t.n. download \gg t.n. upload

Osservazione. Storicamente, era possibile osservare la *topologia dell'Internet*, quando era davvero un modo per interconnettere *poche decine di reti* (Università, centri militari)

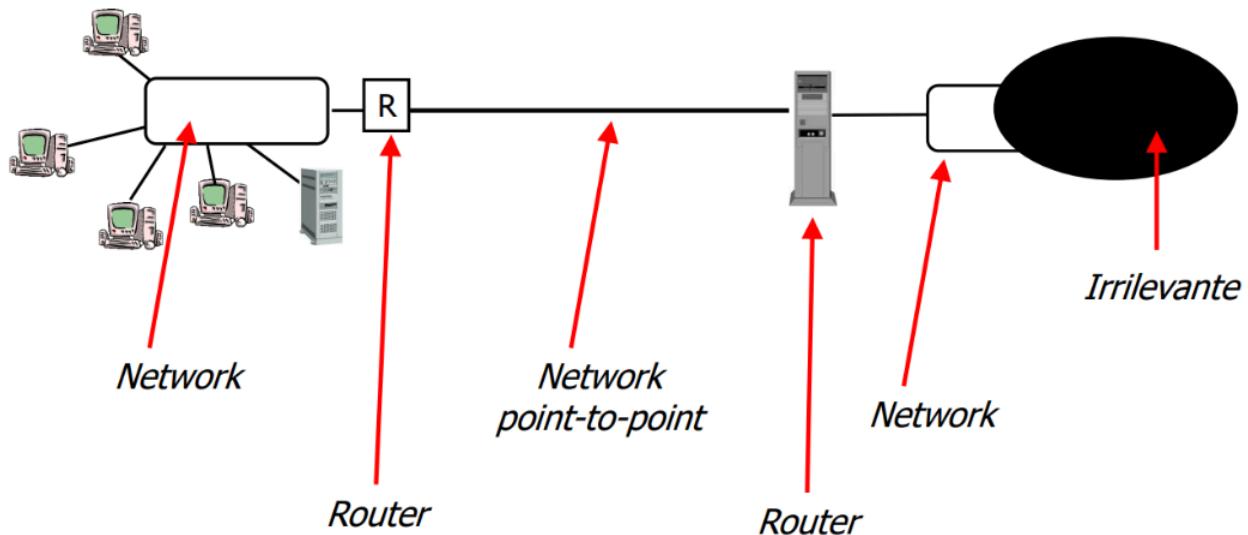


2. Notazioni Internet

Nel corso, useremo le seguenti notazioni per descrivere "*un pezzo di Internet*" (o uno schema ipotetico)



Esempio.



3. Inaffidabilità dell'Internet

Osservazione. L'Internet è *intrinsecamente* inaffidabile, ovvero non presenta dei meccanismi per rilevare e risolvere certi problemi: ne dovranno occuparsi i lati *endpoint*! Questo principio si chiama "*end-to-end*".

Q. Quando l'internet può perdere pacchetti?

1. Una delle network coinvolte è intrinsecamente inaffidabile
2. Un router "*decide*" di scartare dei pacchetti (ci saranno molti motivi per farlo...)

Q. Quando abbiamo dei duplicati?

1. Le network sono intrinsecamente inaffidabili e quindi le network possono duplicare frame

Q. Perché i pacchetti possono arrivare in ordine diverso?

1. Servizio *connectionless*: Ogni pacchetto viene instradato indipendentemente dagli altri pacchetti, quindi possono seguire *cammini di routing diversi* e dunque arrivare alla loro (stessa) destinazione in ordine diverso.

Network Numbers

X

*Definizione e caratterizzazione dei network number. Assunzioni (del corso) sulle network number.
Notazioni per indicare le network numbers.*

X

0. Voci correlate

- Fondamenti su Internetwork

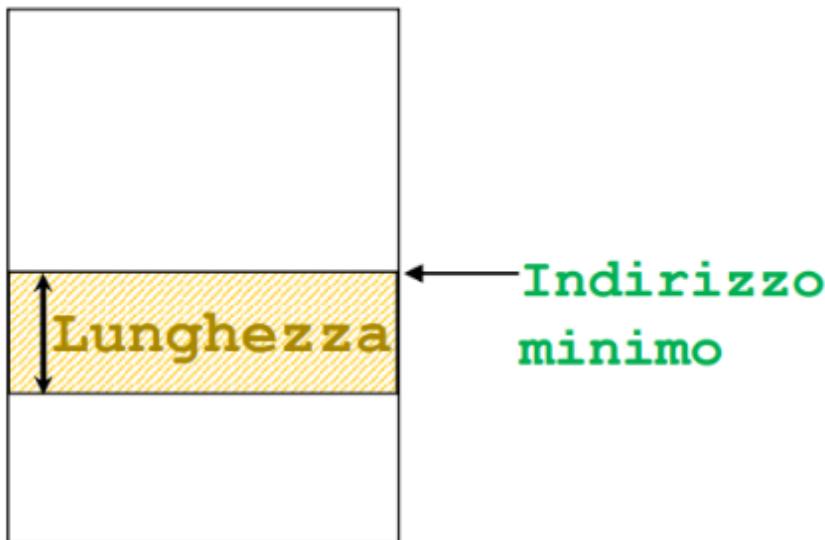
1. Definizione di Network Number

Q. Come implemento il test di appartenenza alla stessa network tra ≥ 2 indirizzi IP?

Una nozione preliminare per capirne la risposta è come segue.

Definizione. Un *Network Number* è un intervallo contiguo di indirizzi IP tali che:

- La lunghezza è un multiplo di 2, in tal caso denotiamo il multiplo con k (quindi la lunghezza è 2^k)
- Il *minimo* del network number ha k bit meno significativi tutti fissati a zero



Qual è la struttura degli indirizzi IP nello stesso network number? Si "assomigliano" nelle prime parti ma poi si differenziano nelle *parti meno significative*. Infatti, dato un network number (ad esempio) di lunghezza 2^4 abbiamo un generico indirizzo IP che va da xxxx.xxxx.xxxx.0000 a xxxx.xxxx.xxxx.1111.

Enunciamo quindi il seguente principio:

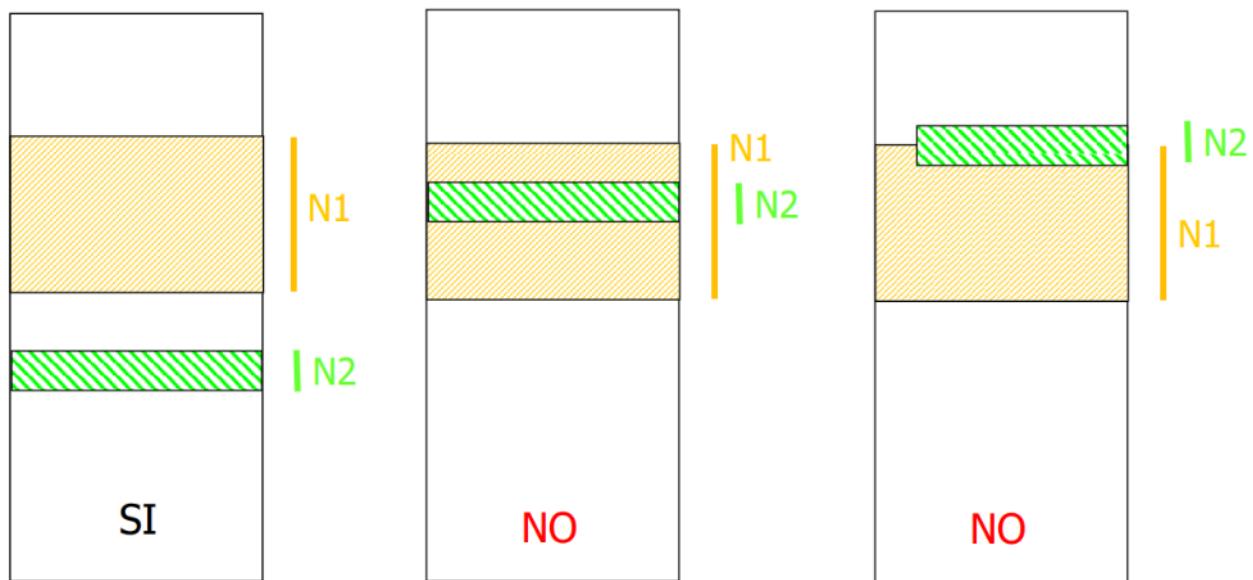
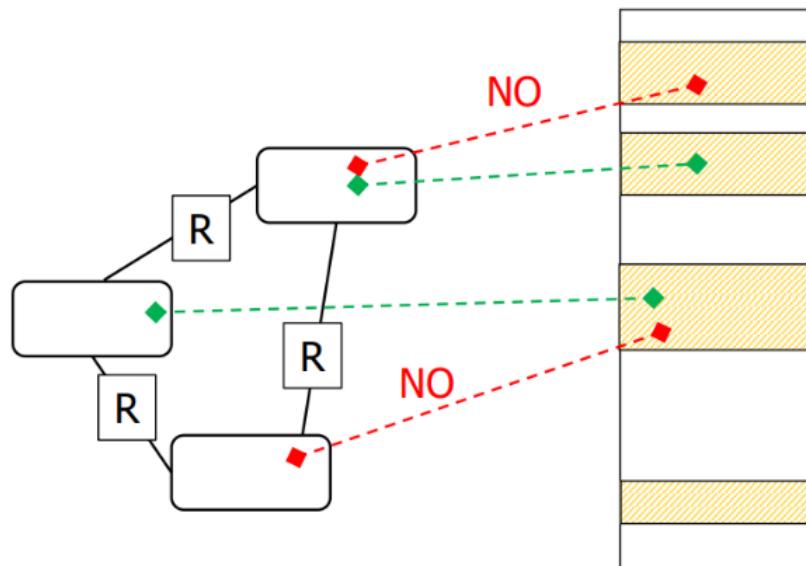
"Ogni nodo di una network si deve avere un idirizzo IP del network number del network appartenente"

X

2. Assunzioni del corso

Inoltre, poniamo un paio di assunzioni sulle *network number*.

1. Ogni *network* ha *un solo network number* ed è univoca. Quindi si ha una corrispondenza uno-ad-uno tra network number e network
2. Non ci sono sovrapposizioni tra network number



La prima assunzione ha le seguenti conseguenze:

- La size della network dipende anche dal network number *"assegnato"*

- Gli indirizzi IP non utilizzati in una network *non* verranno mai utilizzati da nessun'altro (uno "*spreco*")

La realtà delle assunzioni 1), 2) sono diverse, ma non lo vediamo nel corso.

X

3. Notazioni per le Network Numbers

Vediamo delle *notazioni* per indicare le *network numbers*.

Notazione. Una *network number* con lunghezza 2^k e con indirizzo minimo `min-addr` si indica con `min-addr/(32-k)`, ovvero la prima parte indica l'indirizzo *minimo*, la seconda parte indica *il numero dei bit "fissati"*.

Esempio. 131.0.0.0/8 indica tutti i network number compresi tra 131.0.0.0 e 131.1111.1111.1111

Osservazione. Notiamo che se la lunghezza è un *multiplo di 8* allora la network number è visualizzabile a "*occhio*", siccome la suddivisione tra "*parte fissa*" e "*parte libera*" si trova proprio su uno dei "*punti*". Vedremo successivamente che significato hanno la "*parte fissata*" e la "*parte libera*", quando vedremo il partizionamento delle network numbers.

Osservazione. Incrementare la "*lunghezza*" del network number equivale a dimezzare l'intervallo indicato dal network number. Ossia se .../24 indica un network number con 2^8 indirizzi allora .../25 indica un network number con 2^7 indirizzi.

Esercizio. 10.93.22.12/23. Qual è il range degli indirizzi IP?

Allocazione delle Network Number

X

Allocazione delle network numbers: chi e come, operazioni di routing e forensics. Concetto di network number e host number di ogni indirizzo IP. Host number non ammissibili (convenzione), numero di indirizzi IP utilizzabili per ogni network number.

X

0. Voci correlate

- Network Number

1. Chi Alloca le Network Numbers?

Q. Chi decide la suddivisione/distribuzione delle network numbers? Come viene fatta?

L'operazione aritmetica della *suddivisione* delle network numbers si chiama *subnetting* e viene fatto con un procedimento aritmetico simile alla progressione geometrica, ovvero usare il fatto che la seguente serie converge:

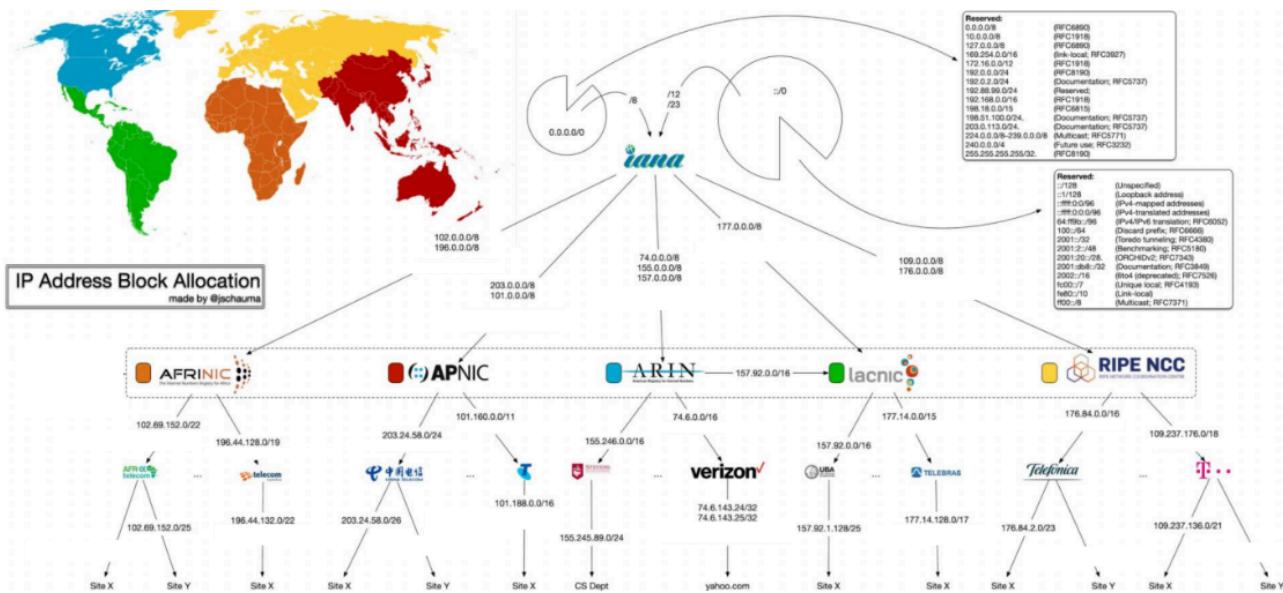
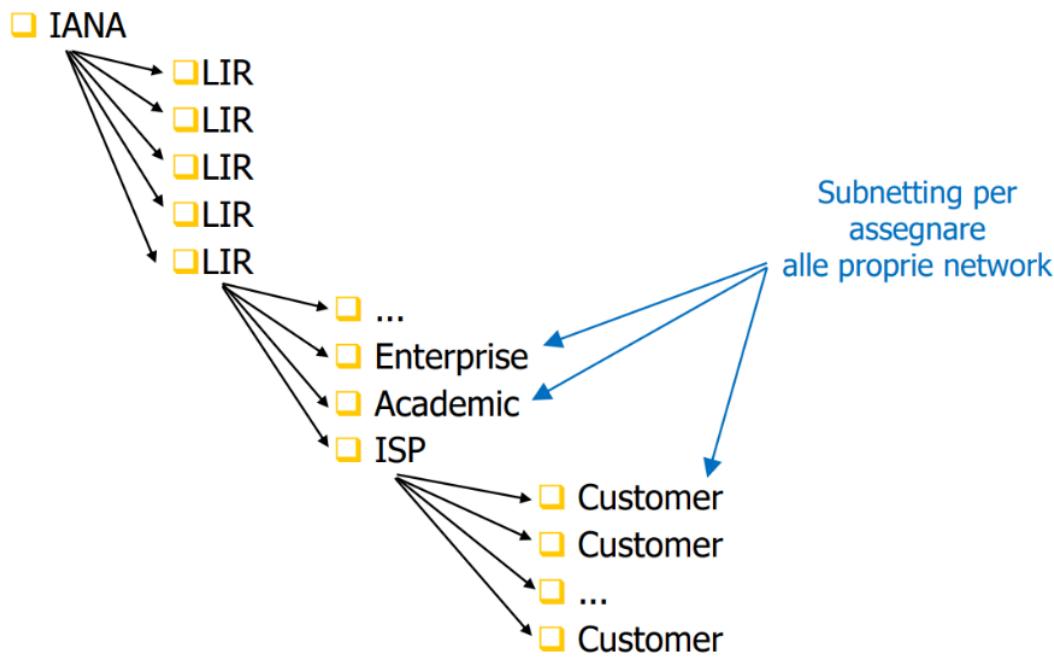
$$1 = 1 + 1/2 + 1/2 = 1 + 1/2 + 1/4 + 1/4 = \dots = \sum_i \frac{1}{2^i}$$

Vedremo dopo *bene* come viene fatto, per ora vediamo *chi* lo fa e quali sono le conseguenze sul significato delle network numbers.

IANA. Un'organizzazione "*possiede*" tutto lo spazio delle network numbers (IANA) e definisce più network numbers e assegna ognuno ad ogni "*continente*", ovvero un'*autorità continentale* che si chiama *LIR* (Local Internet Registry). Le LIR sono 5.

LIR. Dopodiché, ogni LIR-x suddivide il proprio spazio in più network numbers e li assegna ad *ORG-n*, che sono

- O enterprises
- O *istituzioni accademiche*, che effettueranno ulteriore subnetting
- O *Internet Service Providers*, che assegneranno degli indirizzi IP a dei "*customers*" (o un solo indirizzo o un "*piccolo*" network number)



Q. Come associo il network number al proprietario?

Questa è l'operazione di "*forensics*", e dipende dal livello di "*profondità*" della network number in cui si trova.

- Se è della IANA, o uno dei LIR, o una delle istituzioni/ISP/imprese assegnate dalla LIR, è possibile consultare un *servizio di database pubblico* mediante il *protocollo whois*
- Altrimenti i dati non sono pubblicamente accessibili e le suddivisioni non sono pubblicamente descritte, solo le autorità giudiziarie hanno possibilità di effettuare ulteriori indagini.

Osservazione. Le allocazioni dei network number sono gestiti in una maniera *autonoma!*

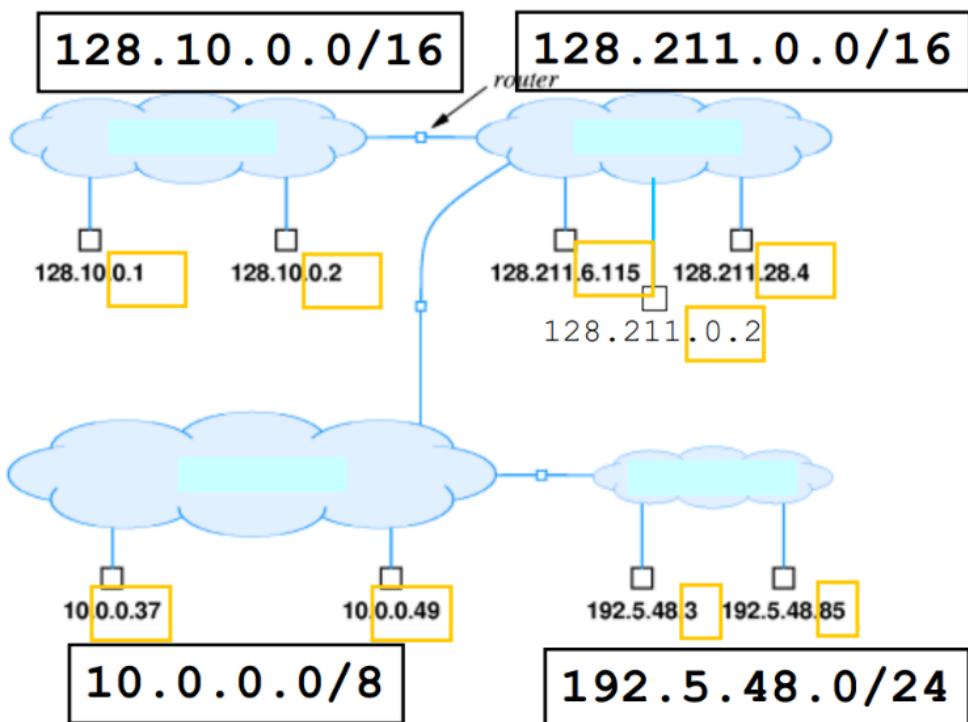
Osservazione. Nonostante ciò, *intradare* i pacchetti è comunque possibile. Infatti, forensics != routing.

2. Network Number e Host Number

Quindi, dato una *network number* $x.x.x.x/k$, abbiamo che possiamo suddividerla in due parti: una "*fissa*", ovvero i primi k bit, e un'altra "*libera*", gli ultimi k bit.

In virtù delle osservazioni fatte, possiamo dire che la prima parte è la *network number* e identifica la network, invece la seconda parte indica il *nodo* e la si chiama *host number*.

La network number è assegnata da un'autorità centrale e identificano nodi con la *stessa network*, l'*host number* viene invece assegnato da un'autorità "*locale*" e identificano i singoli nodi all'interno di una network.



Alcuni *host number* non sono assegnabili per convenzione. In particolare sono:

- Host number con tutti *zero*, siccome dovrebbe rappresentare la *network number* (?)
- Host number con tutti *uno*, siccome rappresenta il "*broadcast*" nel network

Quindi dato una network number di lunghezza x , in realtà posso assegnare 2^{32-x} indirizzi IP a nodi.

Osservazione. Quindi .../31 non esiste come host number, e .../30 è l'host number più piccolo possibile

Subnet Mask

X

IP to network number, notazione aumentata. Definizione di subnet mask, intuizione delle subnet mask. Test di appartenenza alla stessa network tra due nodi.

X

0. Voci correlate

- Network Number
- Allocazione delle Network Numbers

1. IP to Network Number

Q. Supponiamo di avere un indirizzo IP, come ricaviamo la sua network number?

Con un esempio, osserviamo immediatamente che non possiamo sapere a priori la sua network number di appartenenza. Infatti, ad esempi 131.114.9.252 potrebbe appartenere a più network number:

- 131.0.0.0/8
- 131.114.0.0/16
- Eccetera

Quindi dobbiamo "*incrementare*" la nostra notazione e trovare un modo per indicare la network number di appartenenza. Supponendo di avere la coppia (*ip-addr, netnum/k*), indichiamo *l'indirizzo IP* con *ip-addr/k*.

Possiamo comunque *differirlo* da un network number, siccome non è il *numero minimo* (che non può essere un host number!)

In realtà ci sono altre notazioni usate nel gergo tecnico che sono più ambigue, ma nel corso non le useremo.

X

2. Subnet Mask

Vogliamo realizzare finalmente il *test* per indicare se due indirizzi IP appartengono alla stessa network o meno.

Specifichiamo il *network number* al suo *network di appartenenza* indirettamente mediante una *subnet mask*. In particolare, una *subnet mask* è una sequenza particolare a 32 bit che "*inizia con tutti uno*" e a un certo punto "*conclude con tutti 0*".

Esempio. 255.255.0.0 è una subnet mask

Notiamo che quindi, dato un *indirizzo IP* e una *subnet mask*, possiamo ricavare:

- *Lunghezza*: codificata direttamente dalla subnet mask, basta contare gli uno
- *Network Number*: effettuare l'operazione bitwise IP and SUBNET_MASK
- *Host Number*: effettuare l'operazione bitwise IP and not(SUBNET_MASK)

Osservazione. Ci sono molti modi capire se una sequenza *non* è una subnet mask. Esempi:

- Avere l'ultimo byte come un numero dispari
- Altri casi evidenti, come 255.128.255.0

X

3. Test Two IPs in Netnum

Finalmente possiamo relizzare il test per determinare se un certo indirizzo appartiene alla mia stessa network o meno.

Supponendo di avere la *mia subnet mask e il mio indirizzo IP* in configurazione, ho che il predicato $P(x)$ "x sta sulla mia network number" è equivalente a

$$P(x) \iff (\text{my-IP AND my-SM}) = (\text{other-IP AND my-SM})$$

La dimostrazione è *omessa*.

Configurazione Statica e Dinamica dei Nodi su Internet

X

Configurazione dei nodi collegati ad una network. Configurazione statica, configurazione dinamica. Schema generale.

X

0. Voci correlate

- Fondamenti su Internetwork

1. Configurazione Statica e Dinamica

Q. Un nodo collegato ad una network deve avere le seguenti informazioni, per mandare un pacchetto ethernet, deve possedere le seguenti informazioni:

- Indirizzo IP sorgente (proprio)
- Indirizzo IP destinatario
- Subnet Mask
- IP Gateway
- IP del Name Server (non indispensabile nel contesto delle network/internetwork)

Come si ottengono queste informazioni? In *configurazione*. Vediamo che ci sono due metodi per acquisire le informazioni menzionate sopra.

Statico. Sono *inserite manualmente* e non cambiano mai, inoltre vengono memorizzate in memoria. I dettagli per questo tipo di configurazione dipende dal sistema operativo usato.

Dinamico. Ottiene le informazioni automaticamente quando "*si accende*" il calcolatore. In particolare, funziona che un *server di configurazione*, raggiungibile *da broadcast*, assegna tutte le informazioni necessarie. Tuttavia le informazioni fornite possono essere diverse ad ogni "*momento*" di collegamento, ossia ogni volta che "*riaccendiamo il calcolatore*" possiamo ottenere informazioni diverse.

Tipicamente, si usa la configurazione *statica* per i router, "*server*" node e i proxy; invece i "*nodi non server*" usano la configurazione dinamica.

Osservazione. Assumiamo sempre che le informazioni fornite siano sempre corrette.

Protocollo DHCP e ARP

X

Protocollo DHCP e ARP. Protocollo DHCP (semplificazione): funzionamento del protocollo, convenzioni. Protocollo ARP: come trovare indirizzi Ethernet (fisici) da indirizzi IP (logici). Idea del protocollo ARP, notazione dei frame ARP. ARP caching. Esercizio di riepilogo.

X

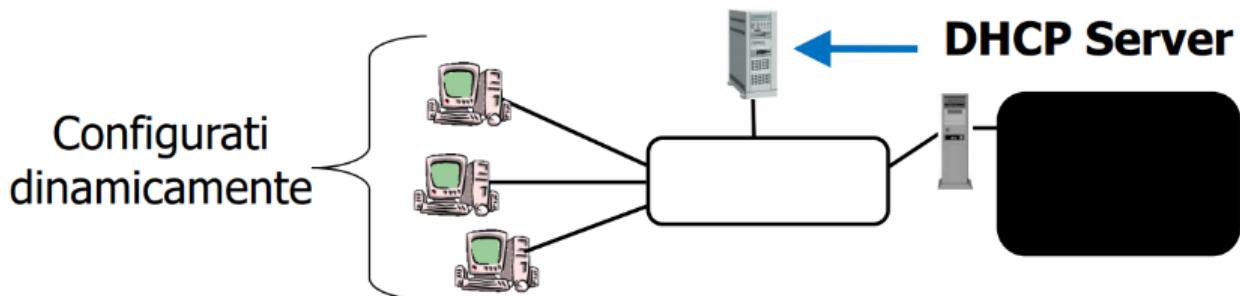
0. Voci correlate

- Fondamenti su Internetwork
- Configurazione Statica e Dinamica dei Nodi su Internet

1. Protocollo DHCP

N.B. Questo argomento è stato presentato in una maniera semplificata, con delle discrepanze col funzionamento reale del protocollo DHCP.

Nella network ho dei nodi da *configurare dinamicamente*. Col protocollo DHCP, questi nodi comunicano col *DHCP server* (ovviamente sulla stessa network) mediante dei *frame broadcast*.



Si segue quindi questo procedimento:

1. Il *client* invia un frame *DHCPDISCOVER* in broadcast ("chi è il server?")
2. Solamente il *server DHCP* ne prende nota e sceglie un *indirizzo IP libero* e crea l'associazione tra indirizzo logico e fisico
3. Il *server DHCP* invia un frame *DHCPOFFER* al client
4. Per "sempre":
 1. Il client invia un frame al DHCP per "*confermare*" che le sue informazioni siano ancora attive
 2. DHCP conferma o revoca l'allocazione

Il contenuto del frame (*payload*) viene descritto dal protocollo DHCP che non vedremo e ci limiteremo a dire che i contenuti siano *DHCPDISCOVER* e *DHCPOFFER + config. info..*

Come sempre, osserviamo che il protocollo si basa sul principio dell'onestà, ossia solo il server DHCP risponde al frame DHCPDISCOVER.

X

2. Protocollo ARP (Address Resolution Protocol)

Q. Come traduco un *indirizzo IP* in un *indirizzo Ethernet*? Ossia, come associo l'indirizzo logico all'indirizzo fisico? Questo serve per inviare *frame* nel network.

Intuitivamente devo inviare frame *al mio network*... ma come?!

IDEA. Il nodo sorgente invia un *frame broadcast* con "payload ARP", che a grossomodo vuol dire "*Chi è IP-x? Dire a ETH-src*", e risponde solo chi ha in effetti l'indirizzo IP.

Notazione. Useremo la seguente notazione per scrivere il payload ARP.

- *Request*: ETH-src, IP-src, ?, IP-dst
- *Response*: ETH-src, IP-src, ETH-dst, IP-dst

Osserviamo che la prima coppia è "*inutile*", in quanto viene ripetuta e non è indispensabile per inviare il frame di richiesta/risposta. I motivi veri ci sono, ma sono complicatissimi e non li vedremo...

Quindi in totale ho i frame composti nei seguenti modi:

ETH-dst	ETH-src	TYPE	PAYLOAD	
FF:...:FF	ETH-A	ARP	ETH-A, IP-A, ?, IP-B	# request
ETH-A	ETH-B	ARP	ETH-A, IP-A, ETH-B, IP-B	# response

Q. Se l'indirizzo IP non sta nel mio network?

Semplicemente mando il pacchetto IP al router gateway di default... ovviamente risolvo il suo indirizzo IP con ARP.

Q. Per risolvere nomi già visti prima, devo ripetere le ARP request?

Nella realtà non è necessario siccome nel sistema operativo è presente il *modulo ARP* che "*la comunicazione ARO*", ossia gestisce anche il caso in cui l'indirizzo IP non sta nel mio network o quando riceve una richiesta ARP, e tiene in memoria una *ARP cache*.

Facciamo degli approfondimenti importanti sul protocollo ARP.

GRATUITIOUS ARP. Osserviamo che nella realtà ci sono *richieste ARP* per "*risolvere il proprio indirizzo*", ossia un frame ARP con payload ETH-A, IP-A, ?, IP-A.

Quando e perché: Ogni volta che un nodo si collega/riconnella ad una network, così:

- Se capisce che c'è un altro nodo avente lo *stesso indirizzo IP*, che è un problema di configurazione.
- Rende obsolete le vecchie entry ARP; infatti, quando si invia una ARP request, tutti i nodi potrebbero prenderne nota e aggiornare la loro ARP cache salvando l'indirizzo IP del richiedente.
- Modifica la switch table

X

3. Riepilogo DHCP e ARP

Per consolidare gli argomenti visti, vediamo uno *scenario comune*.

Esercizio. Supponiamo allocazione dinamica delle informazioni di configurazione. Sia A un nodo "*appena acceso*" e manda un frame al nodo B, che sta in un network diverso. Quali sono i frame inviati da A?

Traccia: Quali informazioni ci servono? IP-dst: OK; IP-src: Non lo conosco; ETH-src: OK; ETH-dst: Non lo conosco

Quindi il procedimento dei frame che invierà A serviranno per:

- Scoprire il suo indirizzo IP, quindi inviare frame DHCPDISCOVER
- Inviare una gratuitous ARP
- Scoprire l'indirizzo IP di IP-dst, che è il gateway siccome conoscendo la subnet mask può determinare che IP-dst non appartenga alla sua network, mandando frame ARP
- Inviare il frame IP

Avendo il seguente elenco, in formato tabella.

dst	src	type	payload
FF:...:FF	ETH-A	DHCP	(DHCPDISCOVER)
ETH-A	ETH-DHCP	DHCP	(DHCPOFFER + INFO)
FF:...:FF	ETH-A	ARP	(ETH-A IP-A ? IP-A)
FF:...:FF	ETH-A	ARP	(ETH-A IP-A ? IP-GW)
ETH-A	ETH-GW	ARP	(ETH-A IP-A ETH-GW IP-GW)
ETH-GW	ETH-A	IP	(... IP-A IP-B ... PAYLOAD-IP)

N.B. Da non imparare a memoria, capire perché abbiamo svolto l'esercizio così

Network Routing

X

Network routing. Metodi di routing: routing statico e routing dinamico, idea. Notazione dei router. Idea del routing statico: routing table, algoritmo per routing statico. Generalizzazione su internetwork: azione di default della routing table. Errori comuni sul routing. Routing delle organizzazioni: come funziona.

X

0. Voci correlate

- Fondamenti su Internetwork

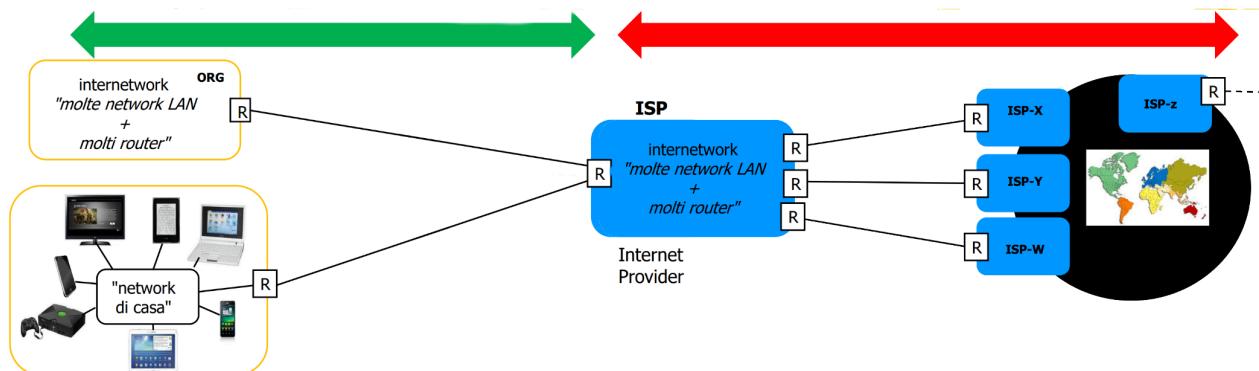
1. Routing Statico e Dinamico

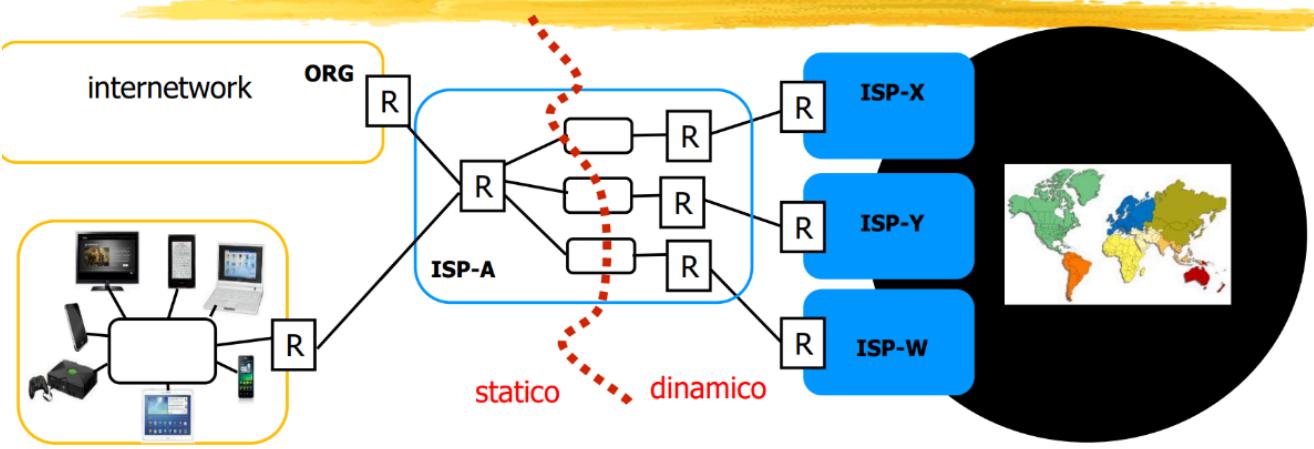
Q. Come fa un router a instradare *"correttamente"* i pacchetti?

Questo è un tema molto ampio, e comprende anche degli argomenti complicati. Per iniziare, ci sono *due* metodi per effettuare il routing:

- *Statico*: il router è configurato da qualcuno e *non cambia mai*; lo si usa quando siamo *"vicini"* agli end point
- *Dinamico*: una tabella di routing si aggiorna automaticamente (*"circa"* una switch table) comunicando con i router vicini (differenza dalle switch table) mediante protocollo BGP; lo si usa quando siamo invece *"lontani"* dagli end point; non vedremo questo tipo di routing nei dettagli.

I vantaggi del routing statico sono la *semplicità e velocità*, tuttavia non reagiscono a guasti o variazioni di traffico. D'altra parte, il routing dinamico è più *complicato e lento* ma è in grado di reagire a guasti, picchi di traffico o variazioni.





2. Routing Statico

Vediamo come un router che effettua il routing statico può scegliere il "*next hop*" (ossia il nodo successore a cui inviare il frame).

Notazione. Osserviamo che ogni router ha un'identificazione per *ogni sua interfaccia* ("lato"), ed essendo a più interfacce (per collegarsi a più network), ha più informazioni di configurazione su network; e ciò comprende gli indirizzi IP. Ogni router associa un identificatore alle proprie interfacce, e li indichiamo in forma simbolica senza ambiguità. Alcuni esempi:



Vediamo l'idea di base per scegliere il "*next hop*" nel routing statico.

Idea. Ogni router ha una *routing table statica*, composta da due colonne:

- *Network number*: Il network number di IP-dst
- *Interfaccia e Azione*: Sull'interfaccia specifica, compiere un'azione specifica. Le azioni possono essere due:
 - "*Consegno direttamente*" se il destinatario appartiene ad una delle network number del router
 - "*Instrado*" altrimenti, e lo invio al router successivo

In un certo senso, è una mappa "*network number* \mapsto *interfaccia, azione*"-

Notazione. "*consegnare*" avvolte lo si scrive come "*direct*", e "*instradare verso IP-x*" lo si scrive solo come "*IP-x*".

Quindi si ha il seguente algoritmo:

ALGORITMO. (*Routing Statico*)

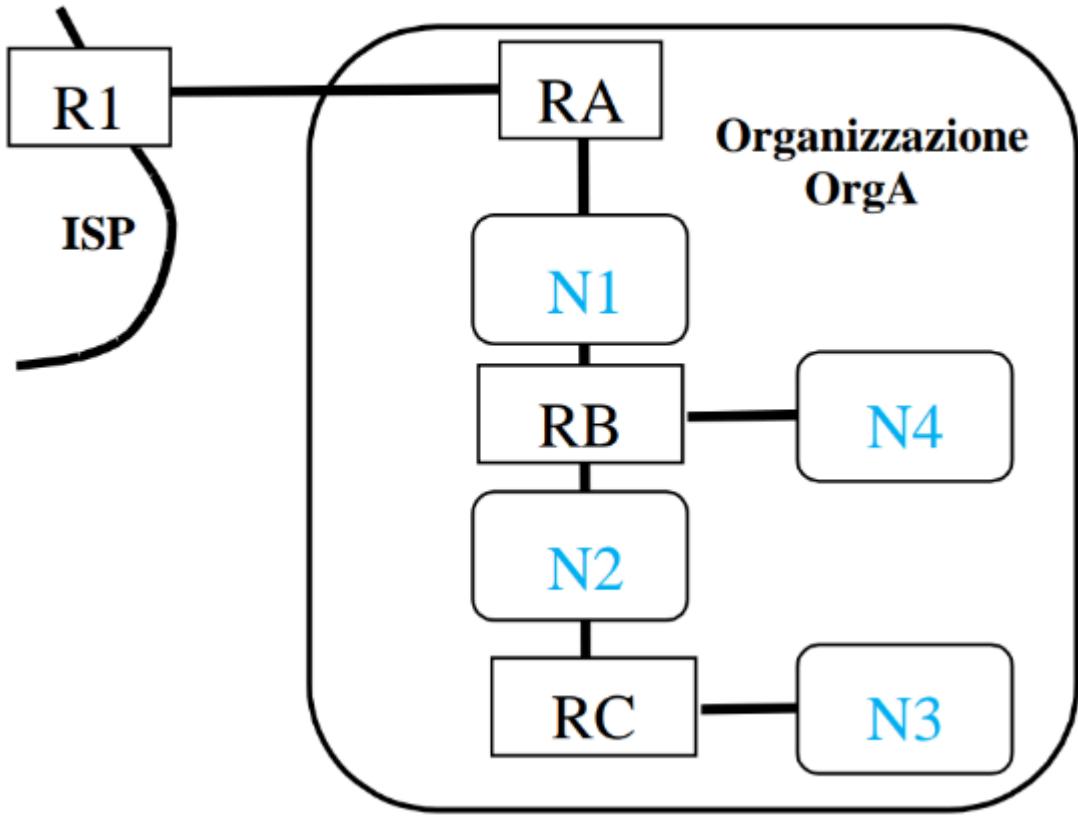
1. Per ogni pacchetto (IP-src, IP-dst, ...) da instradare:
 1. Se esiste una riga del routing table x tale che $\text{IP-dst} \in x.\text{range}$ (per fare il test basta usare le subnet mask):
 1. Eseguire $x.\text{action}$
 2. Altrimenti:
 1. *Scartare* (!)

Notiamo che in un passo successivo il passo "*scartare*" rende inoperabile la comunicazione tra la "*organizzazione interna*" e il "*mondo esterno*" (l'Internet), siccome ogni network number dev'essere descritto almeno in una riga.

L'*azione di default* è in grado di generalizzare il routing statico in un modo tale da rendere possibile comunicazione con l'internet.

IDEA. Invece di scartare pacchetti destinati a network "*esterni*" (ovvero che non appartengono alle network number che conosco), lo mando di default al *router di frontiera*. Quindi nel routing table si ha una riga composta da (default, IP-R') dove R' è il router successivo per poter raggiungere al router di frontiera (per poter "*uscire*" dall'organizzazione).

Esercizio. Creare delle tabelle di routing per la seguente organizzazione:



Esercizio. Supponiamo che il router RC riceve un pacchetto destinato a IP-A-N4. Cosa fa? Scrivere il frame che andrà a inviare.

Traccia: lo instrada verso IP-RB-DOWN, il frame sarà ARP(IP-RB-DOWN), ETH-RC-UP, IP, (pacchetto). Nello svolgimento specificare anche come si effettua la risoluzione ARP.

Consiglio. Partire dal caso simbolico ed eventualmente (se necessario) ricondursi al caso numerico.

Osservazione. Nelle tabelle di routing si ha solo una visione parziale del *network*, conosce solo la "*direzione*" della destinazione. Non conosce MAI il cammino completo (nel caso di "*network lontane*")!

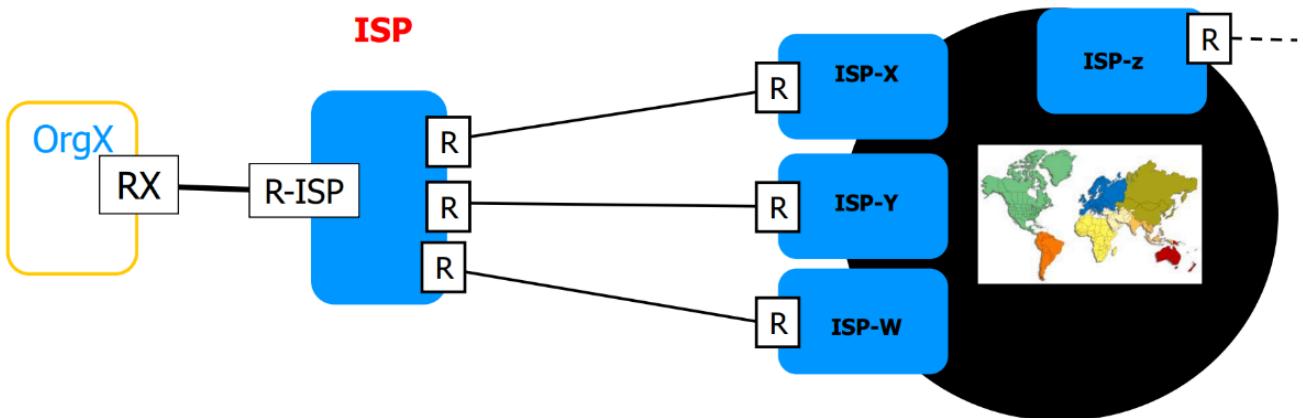
ERRORI COMUNI. (*Ocio!*)

1. Omettere la riga di default; come fa a comunicare con l'Internet?
2. Indicare indirizzi IP non direttamente connessi col router mediante network; come fa ad inviare le richieste ARP? (l'osservazione di prima)
3. Indicare "*direct*" per l'invio del pacchetto al router di frontiera; in questo modo manderei il pacchetto *a tutte le calcolatrici collegate su Internet*.

3. Routing delle Organizzazioni

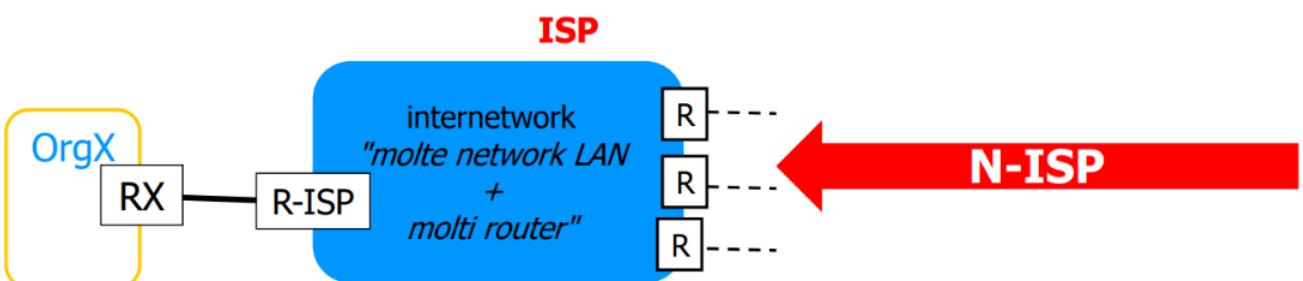
Vedremo il *procedimento operativo* per suddividere le network number. Tuttavia, ci sofferiamo *come* funziona il routing sulle organizzazioni.

Abbiamo il seguente scenario: un'organizzazione ORG desidera collegarsi ad internet, e quindi acquista un network number N-ORGx da un'ISP (che è contenuto in N-ISPs). Dopodiché, l'ISP ha ulteriori router che si "collegano all'Internet"



1. *Network di collegamento tra RX e R-ISP*: è un collegamento "*point to point*", ed è una network a tutti gli effetti. Quindi deve avere una network number. In linea di principio, può essere o a carico di ORG o a carico di ISP; ovvero, o è *interno* a N-ORGx o è aggiuntivo. Nel nostro caso, considereremo sempre questa network a *carico di ISP*.
2. *Resto dell'internet*: Il collegamento di ORGx è irrilevante all'Internet, non ha necessità di informare nessuno!
3. *ISP*: Deve modificare le tabelle di routing dei propri router, in un modo tale da instradare N-ORGX verso IP-RX-Inbound, invece di scartarla.
4. *ORGx*: Può suddividere il proprio network in più network col procedimento di *subnetting*. Si usa il *routing statico*, e le regole "*otherwise*" instradano verso ISP-R-ISP-Inbound

Notiamo che nessuno è informato del subnetting di ORGx, è irrilevante dal punto di vista esterno. Per l'ISP c'è solo un network number (N-ORGx) e per l'Internet c'è un solo network number (N-ISPs).



3.

Subnetting

X

Subnetting in termini di operazione aritmetica. Osservazione sui numeri in base 2: partizionamento in metà inferiore e superiore di un numero a n bit. Problema di subnetting: calcolare la dimensione del blocco di network number da comprare aver avere m network con n_1, \dots, n_m nodi. Procedimento errato e corretto.

X

0. Voci correlate

- Allocazione delle Network Numbers
- Network Routing

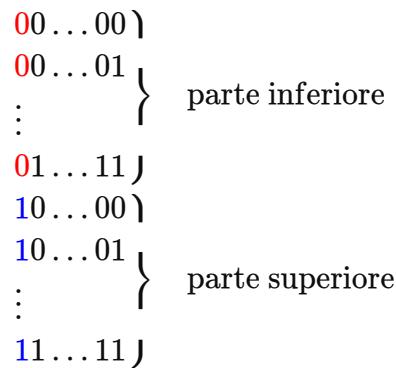
1. Subnetting

Q. Supponiamo di comprare un network number e li vogliamo "*distribuire*" tra tre network; come li suddivido in una "*maniera equa*"?

Il *subnetting* ha come obiettivo rispondere alla domanda di prima, in particolare andrà a creare *quattro suddivisioni del network number*; tre di queste parti andranno alle network, una sarà inutilizzata. Vediamo come si fa facendo la seguente *premessa matematica*.

Osservazione. Data una sequenza di N bit, si ha che n appartiene alla metà superiore di tutti i numeri possibili rappresentabili *se e solo se* il bit più significativo è impostato a 1.

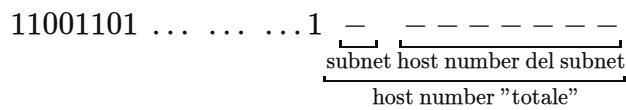
Abbiamo quindi lo seguente schema:



Analogamente possiamo creare un sottopartizionamento della parte inferiore fissando il *secondo bit significativo*, creando quindi altre due partizioni (se lo faccio anche dalla parte superiore), andando così via...

In un certo senso, i primi k -esimi bit *fissati* e *immutabili* definiscono il *subnet*, invece gli ultimi bit liberi formano degli *host number* nel network number partizionato.

Esempio. Supponiamo di avere una network number del tipo xxx/23, ossia gli ultimi 9 bit sono liberi. Per fare il subnetting, fisso ulteriori k bit degli ultimi 9 bit e il resto sono gli host number effettivi. Uno schema è del tipo



Il proprietario del network può quindi suddividere come vuole il suo network number, fissando i primi k bit più significativi a disposizione. Notiamo che è pressapoco lo stesso procedimento che fanno le organizzazioni a livello mondiale.

Esercizio. Dato il network number 200.23.16.0/20, dire:

- Il numero di indirizzi disponibili e usufruibili
- Suddividere la network number in:
 - 4 blocchi della stessa dimensione
 - 8 blocchi della stessa dimensione
 - 2 blocchi di 1/4 e 4 blocchi di 1/8
- Al riferimento di esercizio di prima, calcolare l'indirizzo IP minimo di ogni suddivisione

Traccia: Vediamo solo l'ultima richiesta. Prima dividiamo in 4 blocchi da 1/4 fissando i primi 2 bit, avendo così i 2 blocchi da 1/4. Dopodiche per i restanti blocchi, li dimezziamo di nuovo fissando il terzo bit ottenendo dunque 4 blocchi di 1/8. Rimane al lettore da svolgere effettivamente l'esercizio, creando una tabella di suddivisioni.

Vediamo un altro aspetto pratico del subnetting.

Q. Supponiamo di volere m network, in cui ognuna ha n_1, \dots, n_m nodi. Di quale dimensione devo comprare il network number?

Procedimento errato: Sommare n_1, \dots, n_m e arrotondarlo alla potenza di 2 più vicina (in alto), e comprarsi una network number di quel esempio.

E' un procedimento errato, in quanto non garantisce che la dimensione fornita sia sufficiente. Per esercizio dimostrare che non funziona per $m = 2$ e $n_1 = 520$ e $n_2 = 270$ (intuitivamente, prendo numeri "*vicini*" a potenze di due dal basso).

Procedimento corretto: Arrotondare n_1, \dots, n_m alle potenze di due più vicine e denotarle con $\hat{n}_1, \dots, \hat{n}_m$ e poi sommare le potenze di due e arrotondare la somma alla potenza più vicina di due.

Usando lo stesso controesempio di prima, vedremo che funziona.

Osservazione. Il problema dello *spreco* degli indirizzi IP è purtroppo inevitabile... in particolare è frequente quando abbiamo network point-to-point (4 indirizzi /30).

SEZIONE C. ULTERIORI ASPETTI DELL'INTERNET

Indirizzi IP Pubblici e Privati

X

Fondamenta sui collegamenti residenziali. Nuovo principio: differenza tra indirizzi IP privati e pubblici, i 4 intervalli di indirizzi privati. Definizione intuitiva di indirizzo IP privato, differenza dagli indirizzi IP pubblici. Struttura di assegnazione di indirizzi IP privati e pubblici. Conseguenze del principio. Motivazioni. Modulo NAT per collegamento diretto dai nodi con indirizzo IP privato (cenni). Osservazioni varie e curiosità storiche.

X

0. Voci correlate

- Fondamenti su Internetwork

1. Indirizzi IP Privati

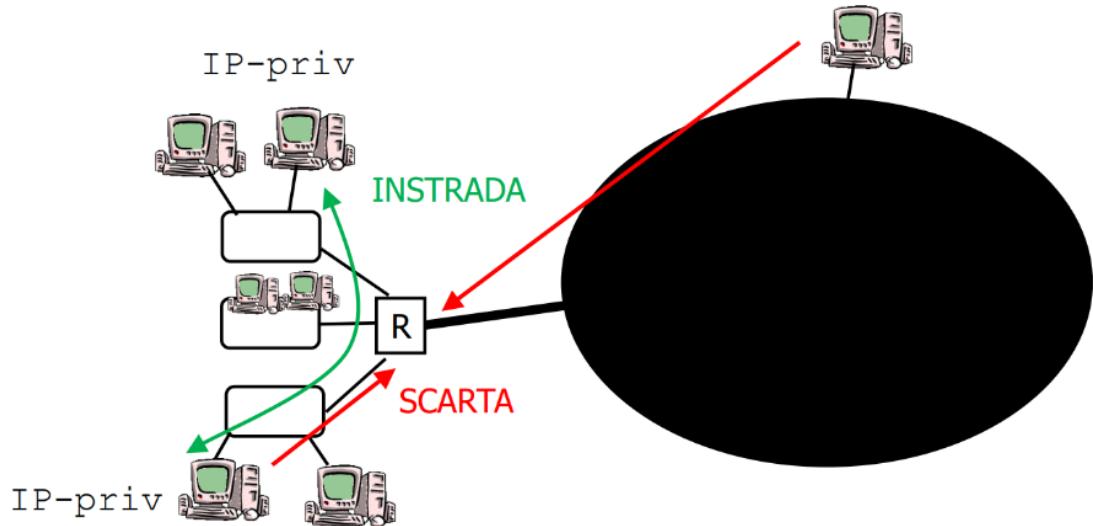
Fin'ora avevamo dato per scontato che tutti gli indirizzi IP fossero "*pubblici*", ossia tutti avessero un indirizzo IP univoco in *tutto Internet*. In un certo senso, davamo per scontato che fossero "*equivalenti*".

Nella realtà, questo non accade e bisogna fare la distinzione tra *indirizzi IP privati* e *indirizzi IP pubblici*.

Nel protocollo IP sono stati designati *quattro intervalli* degli indirizzi IP (network numbers, diciamo) per contenere gli "*indirizzi privati*", e sono:

- 10.0.0.0/8
- 172.16.0.0/12
- *192.168.0.0/16* (!, l'unico da conoscere per cultura)
- 224.0.0.0/4: Indirizzi IP per multicast (*non vedremo*)

Intuitivamente, un indirizzo IP è un indirizzo per cui ogni *router di frontiera deve scartare pacchetti* provenienti da indirizzi IP privati o destinati verso indirizzi IP pubblici. In un certo senso, avere un indirizzo privato vuol dire poter usarlo solo all'interno dell'*organizzazione* (o network, o internetwork).

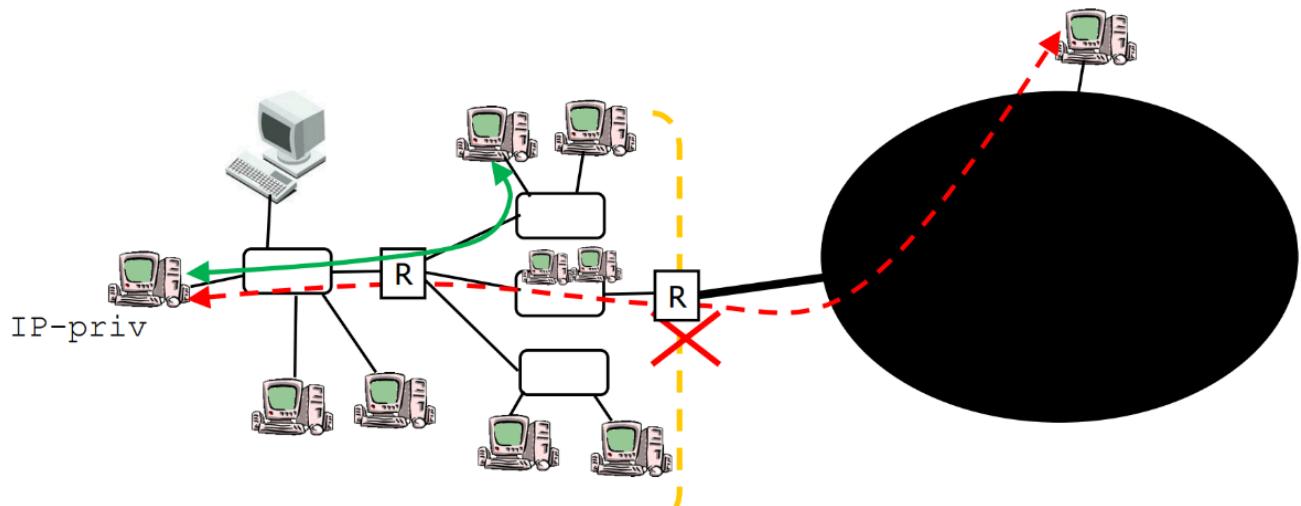


X

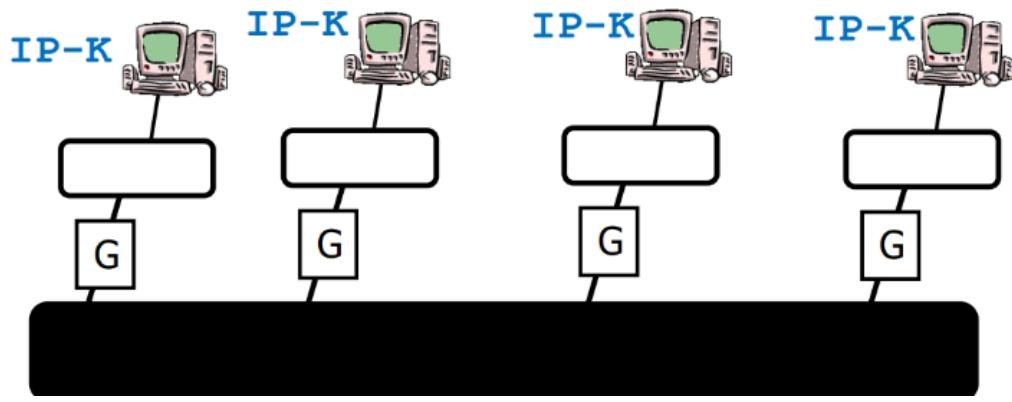
2. Conseguenze degli Indirizzi IP Privati

Il principio degli *indirizzi IP privati* fa crollare molte certezze, portandosi con sè molte conseguenze. Vediamo le conseguenze più fondamentali:

1. *Nodi con indirizzi IP privati non hanno più connettività col "resto del mondo"*: Sarà in grado solo di comunicare con i nodi all'interno dell'organizzazione ("ambiente"), ossia tutti i nodi per cui non passa il router di frontiera in mezzo

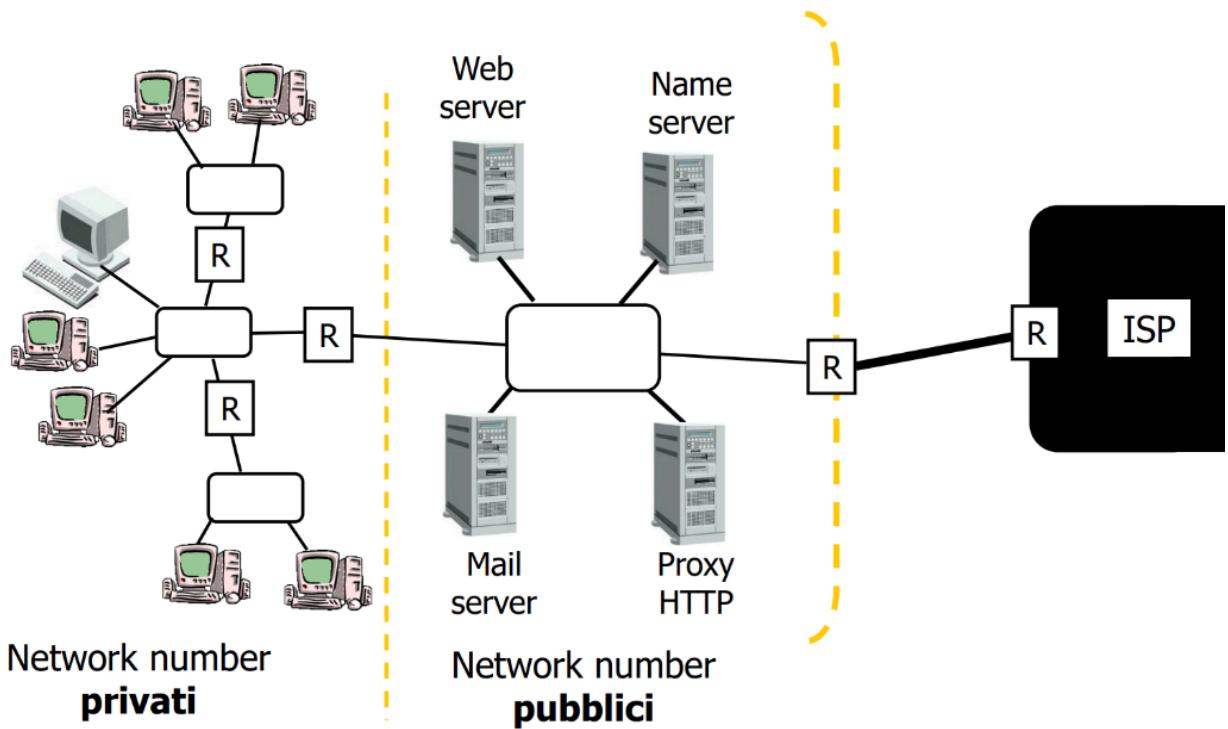


2. *Gli Indirizzi IP privati non vanno acquistati e informati*: Essendo gli indirizzi IP privati usufruibili solo all'interno delle organizzazioni e non *tra* "organizzazioni", l'assegnazione degli indirizzi IP privati *non sono da informare*, quindi neanche da acquistare. Al contrario, l'assegnazione degli indirizzi IP pubblici vanno informati e acquistati da ISP.
3. Ogni *network number* può comprendere o solo *indirizzi IP privati* o *indirizzi IP pubblici*.
4. Molte organizzazioni *diverse* possono usare gli stessi *network number* per indirizzi IP privati. Gli indirizzi IP privati sono quindi univoci *solo a livello della "organizzazione"*.



Riflettendo sulle conseguenze (in particolare 1., 2. e 3.), si evince che in un'organizzazione si effettua tipicamente l'assegnazione di indirizzi IP pubblici/privati, a seconda della *necessità* di comunicare con Internet. Ossia:

- Endpoint "*user*": privati
- Endpoint "*server*" (come mail client, DNS, eccetera...): pubblici



3. Motivazioni degli Indirizzi IP privati

Q. Perché effettuare la distinzione tra indirizzi IP privati dagli indirizzi IP pubblici?

Tanti motivi di natura pratica o tecnica:

- Si rende più *semplice* ed *economico* gestire le *network number* (il vero motivo "originale", da un punto di vista storico)
- Molto meno nodi sono visibili al *pubblico*; quindi abbiamo meno rischi di attacchi (il vero motivo "odierno")

- La stragrande maggioranza delle app funziona senza dover "parlare con Internet". Infatti:
 - Mail: contatto il *mail server* interno all'organizzazione
 - Web: contatto il *proxy*
 - Name Resolution: contatto il *DNS interno*

Quindi se nella mia organizzazione ci sono già il mail server, il proxy, il DNS interno allora ogni applicazione non dovrebbe avere necessità di comunicare con Internet.

X

4. Cenni al NAT

Q. Se un nodo con indirizzo IP privato, per qualche motivo, avesse bisogno di comunicare col resto del mondo?

Esempio. Navigazione web senza proxy, o collegamenti residenziali (vedremo bene dopo)

La *tecnologia NAT* (Network Address Translation) permette la comunicazione con Internet in queste casistiche. La vediamo solo da un *punto di vista "funzionale"* (ossia, *"come si usa?"*) e vediamo i seguenti 3 aspetti:

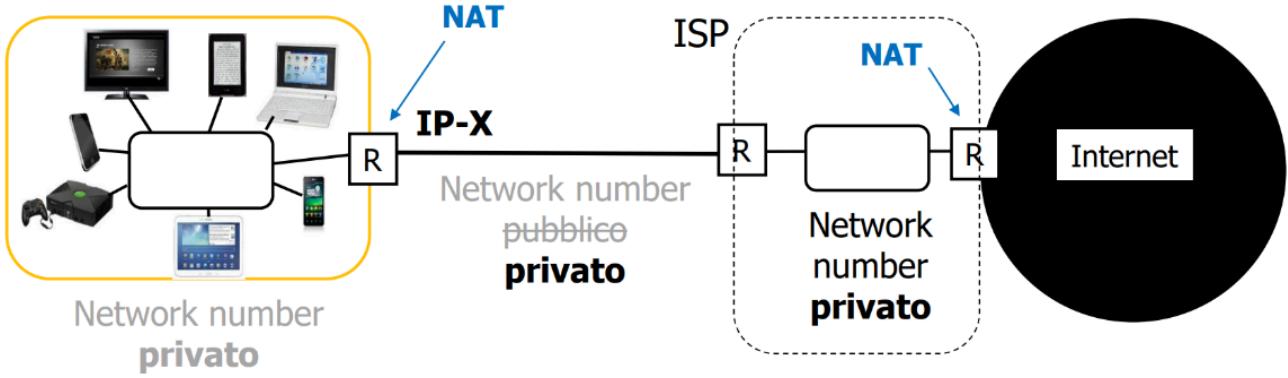
1. Ogni *organizzazione* deve avere un "*dispositivo NAT*" sul router di frontiera
2. Tutti i nodi dell'organizzazione hanno "*lo stesso indirizzo IP*" dall'esterno
3. Le comunicazioni *uscenti* non necessitano configurazione, viene gestito dal router; le comunicazioni *entranti* necessitano invece di configurazione, per poter essere "*raggiungibile*".

Gli aspetti *implementativi* ("come funziona?") del NAT sono oggetto del corso "*Reti di Calcolatori 2*" del prof. Martino Trevisan (il bro 😎).

Osservazione. Nella comunicazioni client-server (client è il nodo con indirizzo IP privato), il NAT non necessita di configurazione. Invece, se abbiamo *applicativi server* allora sarà necessario configurare il NAT

Osservazione. Un nodo con indirizzo IP privato non sa qual è il suo indirizzo IP visibile dall'esterno

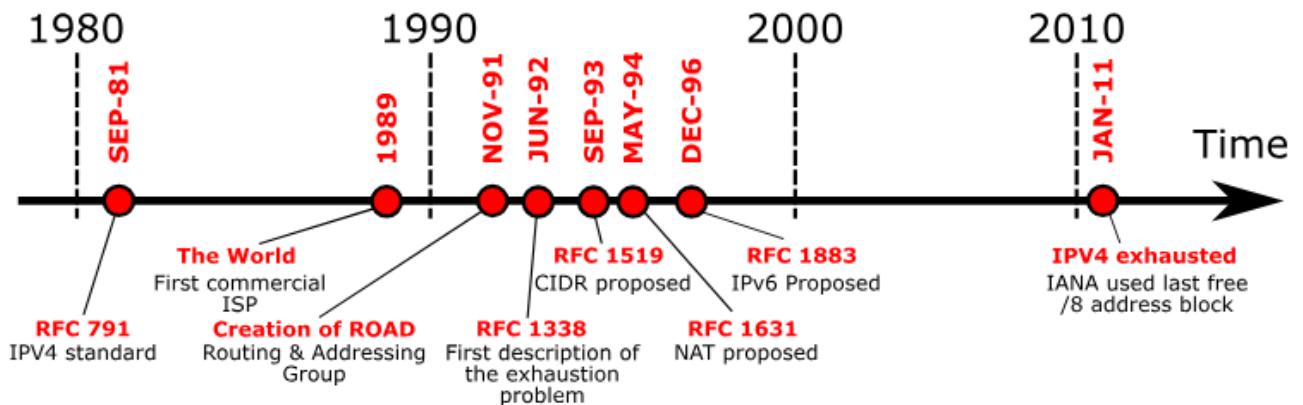
Osservazione. Spesso gli ISP assegnano indirizzi IP privati, con un ulteriore modulo NAT sui suoi router. Ciò crea ulteriori complicazioni...



Curiosità Storiche.

Nel 1991, gli indirizzi IP pubblici si stavano consumando *molto rapidamente*; quindi iniziarono le preoccupazioni e le discussioni, sviluppando il modello IPv6 nel 1993. Tutti si attesero una diffusione relativa rapida di IPv6, tuttavia sono passati circa 25 anni e sono usati "poco", siccome gli indirizzi IPv4 si erano depletati solo nel 2010.

Cosa è successo? In concorrenza si sviluppo il framework del NAT tra il 1993 e 1994, rallentando significativamente il *consumo di indirizzi IP pubblici*. Si preferì di usare quindi il NAT.



Network Point-To-Point

X

Network Point to Point (fondamenta): casi d'uso, tech stack delle network point to point. Mezzi fisici, protocollo PPP e struttura dei frame.

X

0. Voci correlate

- Fondamenti su Internetwork
- Fondamenti su Network
- Modello Fisico dell'Internet

1. Network Point to Point

I collegamenti "*point to point*" sono delle *network WAN* che collega solo *due nodi*.

Esempio. Collegamenti con ISP

Il protocollo IP richiede che ogni *network abbia una network number*, quindi il *network point-to-point* deve avere *network number* e *indirizzo IP alle estremità*. Quali tipi di network number vanno bene?

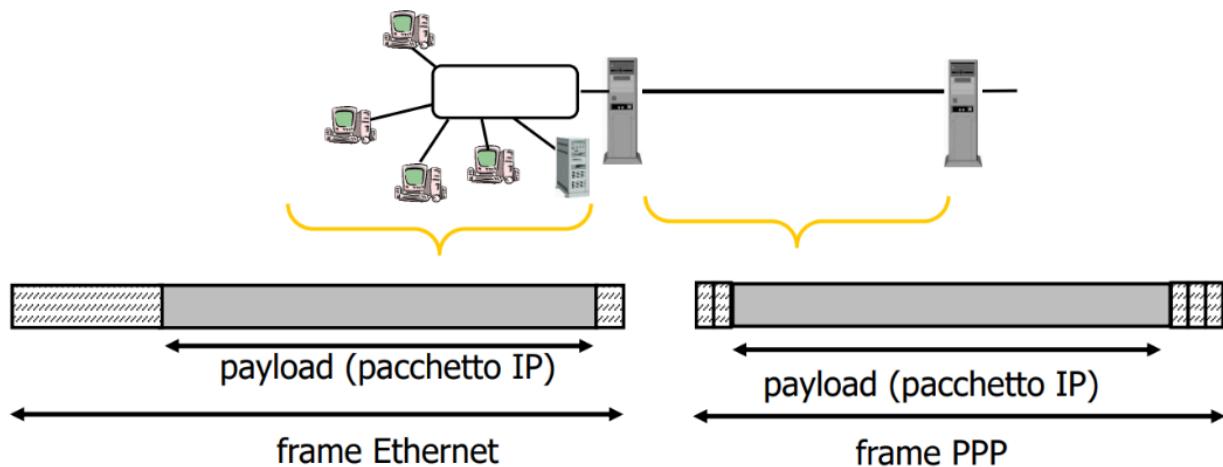
- Network number/31: *NO!* Ricordiamo che gli indirizzi host con tutti uno o zero non sono usufruibili da parte di nodi
- Network number/30: Due host number usufruibili, 01 e 10.
- Network number/29: Abiamo ancora più host number, ma sarebbe uno "spreco" di indirizzo IP...

Si evince che idealmente un *network number di una network point-to-point* sia di tipo /30.

Tecnologia Network PTP. Vediamo addesso la tecnologia per implementare le network point-to-point, quindi le sue proprietà, mezzi fisici, e protocolli.

- **Proprietà.** Una network point-to-point è tipicamente *connectionless, message-oriented* e *unreliable*.
- **Mezzi Fisici.** Molti mezzi fisici, tra cui linee telefoniche, linee dedicate o fibre ottiche
- **Formato Frame.** Il protocollo più comune per i *frame di network point-to-point* è il protocollo PPP. Ogni frame è formato da *due byte + payload IP+ 3 byte*, con MTU di 1500 byte. In particolare:
 - Il primo byte è **7E** e serve per indicare l'inizio del frame
 - Il secondo byte è **21**, indica che il frame è protocollo IP
 - L'ultimo byte è lo stesso del primo, indica la fine
 - Gli altri byte servono per motivazioni di natura elettrica (*checksum*)

Osservazione. Nel frame PPP non si indica (ovviamente) gli indirizzi IP del destinatario o del mittente.



X

Internet at Home: fondamenti sui collegamenti residenziali. Dispositivo "router"/"modem": ruoli. Schema logico di un "modem", scenario tipico (NAT + DHCP). Osservazioni sugli "hot spot" dei dispositivi mobile. Osservazione: ogni casa è una "organizzazione" (spazio di indirizzamento) autonoma/o. Osservazione: stratificazione di NAT. Nota importante: reti Wi-Fi pubblici, pericoli.

X

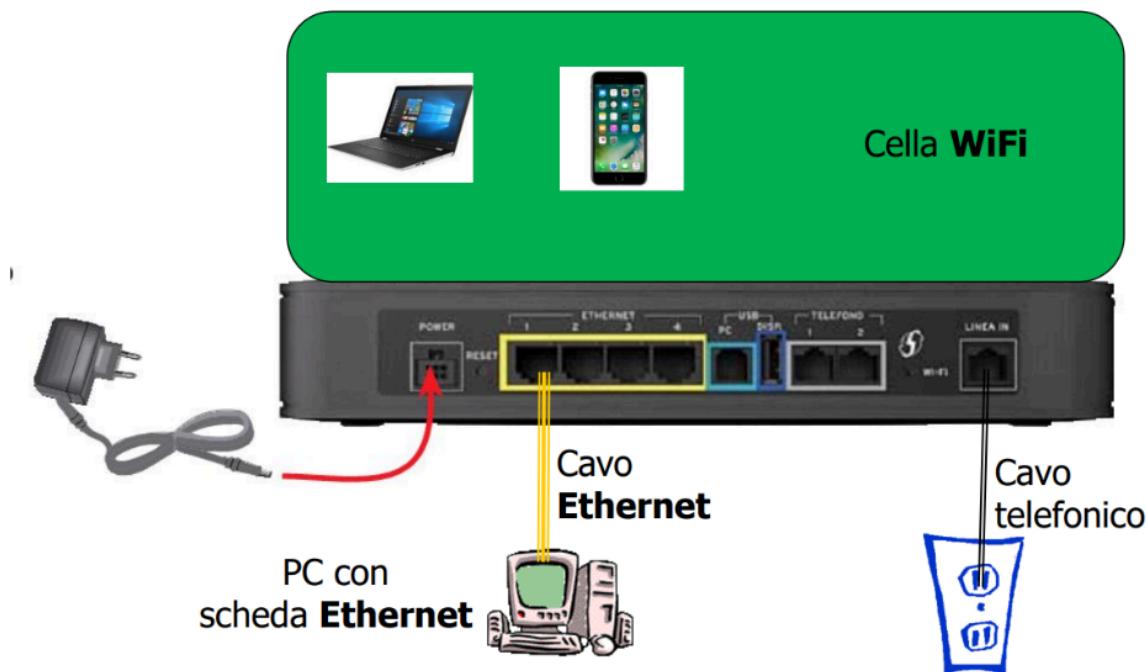
0. Voci correlate

- Indirizzi IP Pubblici e Privati
- Network Point to Point

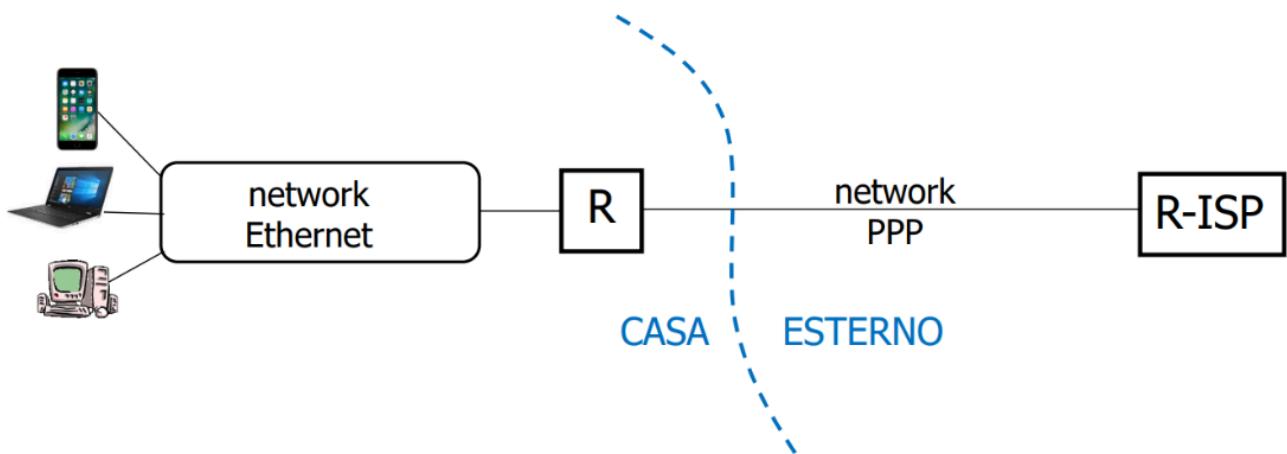
1. Internet at Home: Il "modem"

Nella maggioranza dei casi, a casa si usa un *dispositivo* (uno "scatolotto") che ha molte funzioni, diverse tra loro. Nel gergo comune la si riferisce come *"router"*, *"access point"* o *"modem"* (tutti termini tecnicamente inappropriati...). Essa compie le seguenti funzioni:

- *Router*
- *Ethernet Switch*
- *Ethernet Access Point*
- *NAT*
- *DHCP*
- *Firewall*
- *Web server, per configurazione*

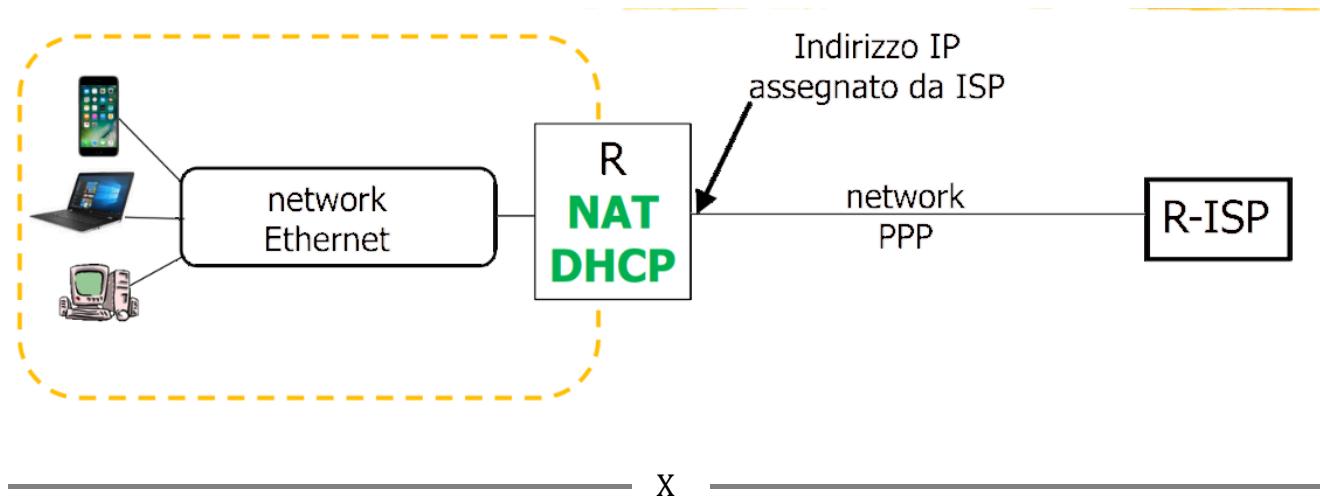


Da un punto di vista logico, abbiamo quindi un *router* che gestisce i collegamenti *all'interno della casa*, ed essa comunica con il router dell'ISP mediante una network PPP.



L'assegnazione degli indirizzi IP viene fatta *in maniera dinamica*. In particolare, dal lato ISP si ha che R-ISP è anche un server DHCP che assegna un indirizzo IP al *router*.

Di conseguenza si ha *solo un indirizzo* IP... come sarebbe possibile collegare più dispositivi? Lo si fa usando l'interfaccia *NAT + DHCP* all'interno del router! Notiamo che quindi la gestione degli indirizzi IP privati è *autonoma*, ossia l'ISP non ne è a conoscenza.

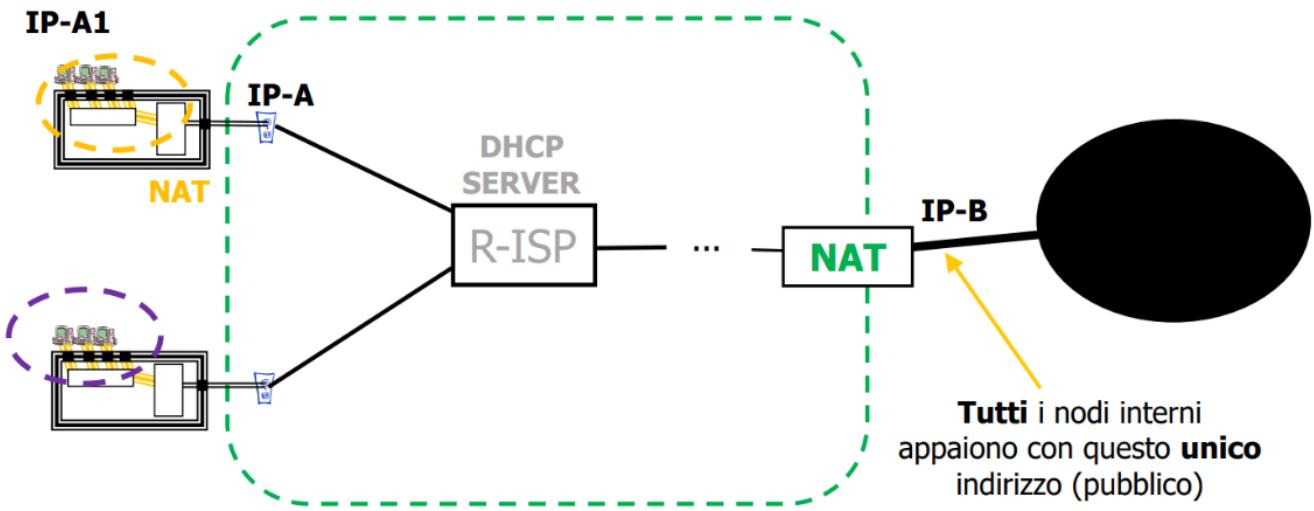


2. Osservazione sui Collegamenti Residenziali

Osservazione. (*Hot Spot*)

Alcuni dispositivi sono muniti di funzionalità "*hot spot*", ovvero possono disporre di un "*nuovo network ethernet Wi-Fi*" a cui poter collegarsi. Questa funzionalità è simile a quanto visto prima, solo che il *dispositivo stesso* ha un modulo NAT+DHCP per permettere i collegamenti. Il provider non ne è a conoscenza di questo collegamento.

Osservazione. In un certo senso, ogni "*casa*" è uno spazio di indirizzamento autonomo (NAT). Ma dall'esterno si hanno *indirizzi IP univoci* dal punto di vista dell'ISP!



Osservazione. Di conseguenza, collegarsi ad una "rete WiFi di casa" vuol dire ottenere un indirizzo IP dal DHCP server "*di casa*", e se inviamo pacchetti dall'esterno sembrerà che siano inviati dalla "*casa*". Quindi è importante nella pratica configurare WiFi con *autenticazione*, altrimenti una qualsiasi persona può collegarsi nella stessa cella e *osservare o modificare* il traffico, o trasmettere "*dati sospetosi*"...

Firewall: motivazione, esempi storici, principio fondamentale. Definizione di Firewall, dove si collocano i Firewall. Formato (intuitivo) delle regole Firewall, algoritmo Firewall. Osservazione: le connessioni TCP sono bidirezionali, conseguenze sulle regole Firewall. Notazione delle regole Firewall, terminologia "traffico entrante" e "traffico uscente". Esempi (esercizi). Considerazione sui "firewall veri".

0. Voci correlate

- Comunicazione tra Processi
- Fondamenti su Internetwork
- Proxy HTTP

1. Motivazioni Firewall

Nella Cybersecurity, alcune **vulnerabilità** (errori nei programmi che possono essere sfruttati per effettuare attacchi) permettono a degli attaccanti di **installare programmi** e fare ciò che vogliono. Questa vulnerabilità si chiama "**RCE**".

In particolare, l'attaccante può **costruire un messaggio** contenente un programma P (che sfrutta la vulnerabilità RCE), trovare un server con quella vulnerabilità e poi inviare il messaggio e "**infettare**" il server. Ancora peggio, può creare un "**for loop**" sugli indirizzi IP su Internet e scansionare quindi **tutti gli indirizzi IP** e infettarli tutti (tutti i server pubblici sono a rischio). Ancora molto peggio, il programma può "**autopropagarsi**", ossia il programma stesso effettua la scansione (in questo caso, il programma è un "**virus worm**"); quindi anche i nodi interni alle organizzazioni sono a rischio...

Facciamo un paio di esempi concreti:

- SQL slammer (2007) infettò 75.000 vittime nei primi 10 minuti di esecuzione
- Il ransomware WannaCry (maggio 2017) è uno dei casi più notevoli di sempre e infettò oltre 200.000 virus nelle prime 12 ore (e poi il 3/4 dell'Internet a fine giornata), causando un danno di ordini di grandezze di miliardi di dollari. Il virus sfruttò una vulnerabilità dei programmi eseguiti su porta 445 (SMB).

Q. E' necessario che i **nodi server** siano accessibili da "**qualunque parte**" dell'Internet (mondo)?

Con la teoria vista, sì. Tuttavia, mediante questi esempi diventa chiaro che abbiamo il seguente principio:

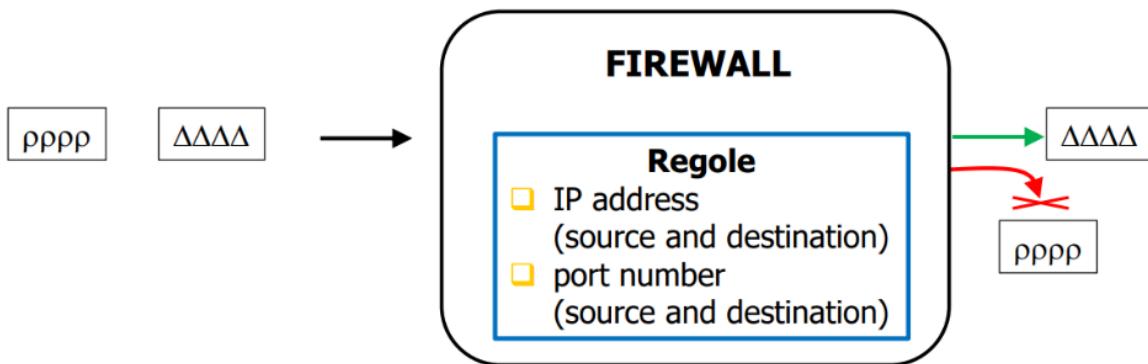
"Nessuna organizzazione del mondo permette tutto il traffico interno-esterno, e anche interno-intero in certi casi, in quanto ogni traffico permesso rappresenta un rischio. Idealmente, si permettono solo traffici permessi."

X

2. Firewall: Definizione e Notazioni

Definizione. Il *firewall* è un *dispositivo software* con due compiti:

- Analizzare il traffico IP
- Permettere o bloccare il traffico, a seconda delle regole configurate



Il firewall si può trovare sia sui *router di frontiera* che sui *dispositivi personali*. In un certo senso, il ruolo di firewall è quello di proteggere il "*ambiente interno*", ossia il dispositivo personale se si tratta di un firewall sui dispositivi personali, o l'organizzazione se si tratta di un firewall sui router di frontiera.

Oggi è *praticamente indispensabili*.

Il firewall si basa sulle "*regole*", che cosa sono precisamente?

Definizione. Le *regole firewall* sono *funzioni booleane a quattro parametri*: IP-src, PORT-src, IP-dst e PORT-dst. Per ogni parametro, possiamo usare gli operatori $=$, \wedge , \vee , $\in [\dots , \dots]$ per determinare se un certo valore sia ammissibile o meno (in realtà ci sono più operatori, ma per noi va bene).

Vediamo adesso come il firewall faccia a determinare se scartare o permettere del traffico IP.

ALGORITMO.

- Quando riceve un *pacchetto IP* con (IP-src:port-src, IP-dst:port-dst):
 - Per ogni regola nelle regole configurate:
 - Se regola(IP-src:port-src, IP-dst:port-dst) ha valore Vero: permettere il traffico
 - Altrimenti (termina il ciclo):
 - Scartare il pacchetto

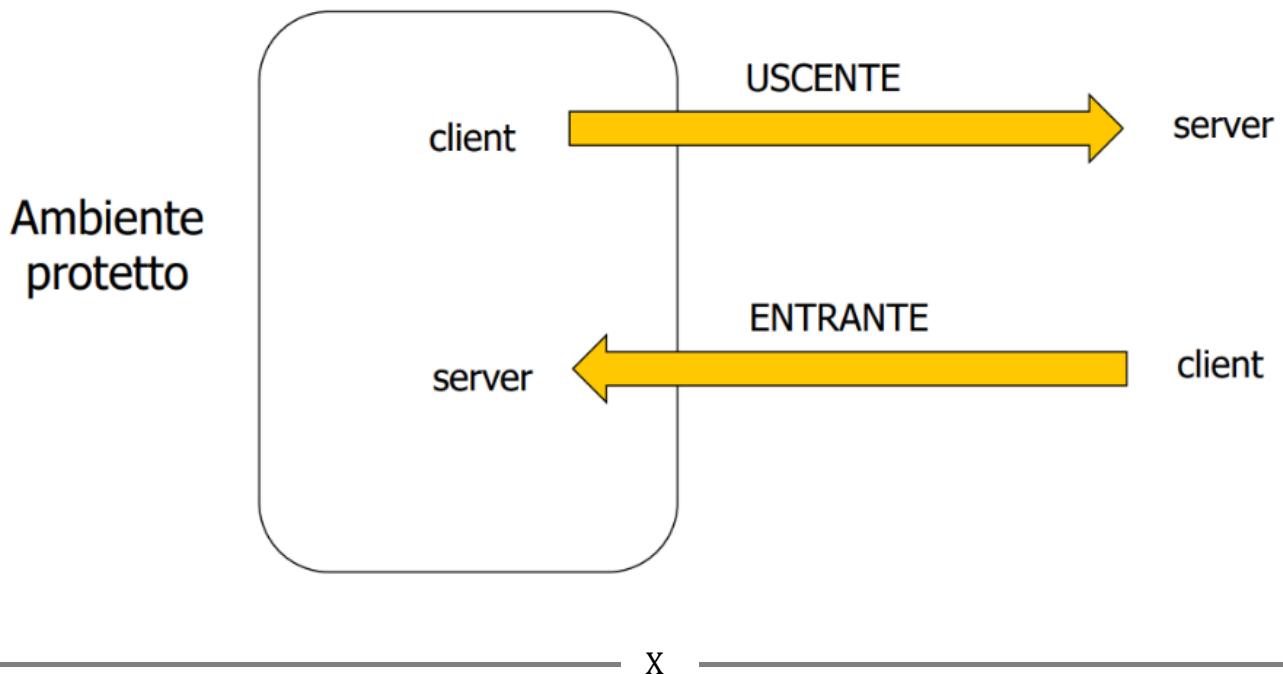
Osserviamo che quindi il firewall tende più a *proibire* che *permettere*, siccome i traffici permessi vanno specificati esplicitamente.

Osservazione. Essendo le connessioni TCP *bidirezionali*, ogni regola firewall che regola traffico TCP (poi encapsulato in IP) deve avere un suo "*coniugato*". In parole poche, devo permettere sia *richieste* che *risposte*.

Notazione. Per indicare la *configurazione* di un firewall, scriviamo una tabella dove:

- Ogni riga è una regola
- Ci sono 4 colonne, che sono i parametri delle regole firewall; quindi IP-src, PORT-src, IP-dst e PORT-dst.
- Ogni riga può avere opzionalmente un nome (vedremo dopo).
- Per indicare che un "*qualsiasi valore è accettabile*", si scrive "?". In un certo senso, indica che non possiamo conoscere quel valore a priori.

Osservazione. Il firewall di frontiera analizza solo "*traffici uscenti o entranti*". Nel gergo tecnico, per "*traffico uscente*" si intende la comunicazione client-server dove client sta nell'ambiente protetto; viceversa, per "*traffico entrante*" intendiamo la comunicazione server-client il server è collocato all'interno dell'ambiente. Ancora più semplicemente, basta chiedersi "*Chi apre la connessione TCP?*"; se è il nodo all'interno, è uscente; altrimenti è entrante.



2. Esempi ed Esercizi

Esempio. (*Proxy HTTP*)

Ci chiedevamo come un router di frontiera di un'organizzazione fosse effettivamente in grado di bloccare traffico HTTP non provenienti dal proxy. Si imposta il *firewall* ammettendo solo traffico HTTP uscente da parte del proxy. Ovvero,

name (optional)	IP-src	IP-dst	port-SRC	port-DST
HTTP Outbound	IP_PROXY	?	?	80/443
Conjugate	?	IP_PROXY	80/443	?

Osservazione. Per calcolare la regola "*coniugata*" basta scambiare i campi. Si sconsiglia fortemente di farlo, piuttosto è meglio ragionare sulla regola da scrivere; in questo modo, si può rilevare degli errori scritti nella regola precedente.

Per gli esercizi seguenti, supporre di stare all'interno di un'organizzazione.

Esercizio. Permettere invio mail da MS dell'organizzazione

Esercizio. Permettere dall'esterno il NS dell'organizzazione

Esercizio. Permettere al NS di risolvere nomi (contattare altri NS)

Esercizio. Permettere il prelievo mail

Esercizio. Permettere traffico HTTP uscente solo da una network N, sia Netnum(N) il suo network number.

Esercizio. Vedere esercizio P2 dell'elenco degli esercizi svolti.

Traccia. Ragionare prima su quali traffico DNS sono necessari, per ogni task da compiere; poi fare la cosa analoga per traffico HTTP e SMTP.

Esercizio. Supponiamo la seguente coppia di regole

name (optional)	IP-src	IP-dst	port-SRC	port-DST
rule_1	IP-MS	?	?	110
rule_2	?	IP_MS	110	?

Ha senso? Che tipo di regola è (usare la nomenclatura "outbound" o "inbound").

N.B. Negli esercizi, si considera più grave scrivere regole "*trop poco permissive*", invece di scrivere regole "*trop restrittive*".

X

3. Considerazioni su Firewall Reali

Facciamo un paio di considerazioni sui "*firewall reali*".

- In realtà la configurazione delle regole sono *molto complicate*, ma più potenti e flessibili. Esempio: Firewall su Windows

- Esistono più operandi per le regole

Inoltre, nella realtà, diventa difficile modellare le vere necessità per la configurazione delle regole, ossia quanti e quali server esistano, quanti client esistano e di cosa hanno bisogno ed eccetera...

Quindi una configurazione del firewall rappresenta una "*approssimazione*" delle necessità reali, e si può entrare in due casistiche:

- *Regole troppo restrittive*: alcuni programmi non funzionano e gli utenti si lamentano. Per motivi di carattere economico, è la più frequente (anche se dal punto di vista tecnico, è molto peggio)
- *Regole troppo permissive*: Funziona tutto, ma rimangono rischi non necessari.

Adesso facciamo un paio di pensieri sul firewall lato endpoint. *Windows Firewall* è uno dei programmi più indispensabili di sempre e permette *numerose* possibilità di configurazione senza costo aggiuntivo. Tuttavia, il profilo "*default*" permette praticamente tutto, e quindi non protegge niente.

Questa è una cosa che non dovrebbe succedere, siccome si ha uno strumento *indispensabile* per la difesa del proprio dispositivo ma non lo si usa. Il motivo a cui si riconduce principalmente è caratterizzato da *esigenze economiche e pratiche*, ovvero gli utenti non sanno configurare il firewall e non vogliono imparare o assumere qualcuno di farlo.

Poi questo rappresenta uno dei drammi odierni, siccome *molti dispositivi* presenteranno delle vulnerabilità RCE; tutti i dispositivi sono questi suscettibili ad attacchi come visti prima.