**Appunti CCNA**

***CAP. 1***

**Modello TCP/IP**: 5 Livelli -> Fisico -> Data Link -> Network -> Trasporto -> Applicazione

**Modello ISO/OSI**: 7 Livelli -> Fisico -> Data Link -> Network -> Trasporto -> Sessione -> Presentazione -> App.

**TCP/IP Protocolli per Livello**:

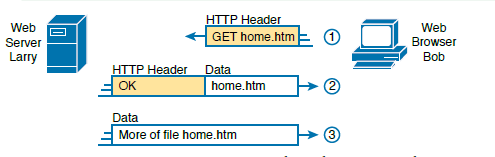
Applicazione: http, pop3, smtp

Trasporto: tcp, udp

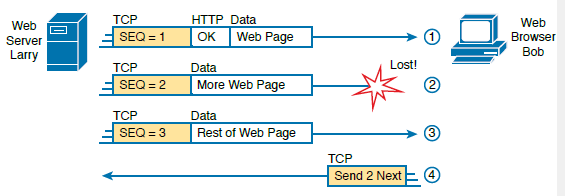
Network: ip, icmp

Data Link e Fisico (considerati insieme): ethernet, 802.11 (wi-fi)

HTTP overview:



TCP overview: nel TCP header contiene un num. Sequenziale che permette a chi riceve il mex di capire se qualche pacchetto è andato perduto. Se cosi fosse viene ri-richiesto e rimandato. (il tcp fornisce un servizio di **error recovery** per il protocollo http del lvl superiore -> Adjacent Layer interaction).

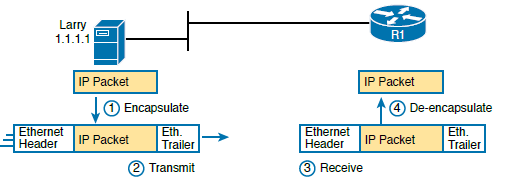


TCP/IP Same Layer/Adjacent Layer:

|  |  |
| --- | --- |
| Same Layer interaction | Due pc usano un protocollo per comunicare con lo stesso layer sull’altro pc. Il protocollo crea un header che dice ad ogni pc cosa fare. |
| Adjacent layer interaction | Su un singolo pc il layer sotto mette a disposizione un servizio per il layer sopra. Il sw o hw implementano la richiesta dei layer più alti verso quelli più bassi. |

**FUNZIONAMENTO IP PROTOCOL**: mette davanti al **segmento** del lvl trasporto un ip header che contiene l’ip source e destination creando un ip **packet**.

Funzionamento Data Link e Fisico:



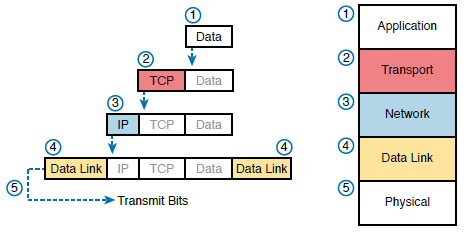
**Step 1**. Larry incapsula l’ip packet con l’ethernet e trailer creando ethernet frame.

**Step 2**. Larry tasmettem fisicamente i bit usando l’elettricità.

**Step 3**. R1 fisicamente riceve il segnale, e ricrea gli stessi bit interpretando i segnali.

**Step 4**. R1 scapsula l’ip packet dall’ethernet frame rimuovendo header e trailer.

**TCP/IP INCAPSULAMENTO**:



**Terminologia TCP/IP**:

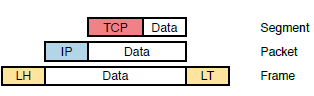
Liv. Appl. -> I dati sono dati.

Liv. Trasp. -> I dati sono segmenti. (segment)

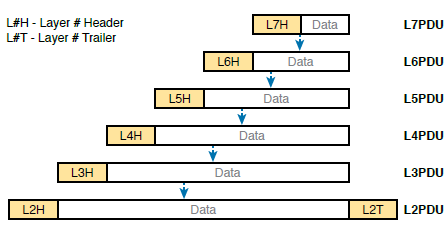
Liv. Network -> I dati sono pacchetti. (ip packet)

Liv. Data -> I dati sono frame.

Il lvl **Trasp. e Network** hanno un **header**, mentre **Data** ha **header e trailer**.

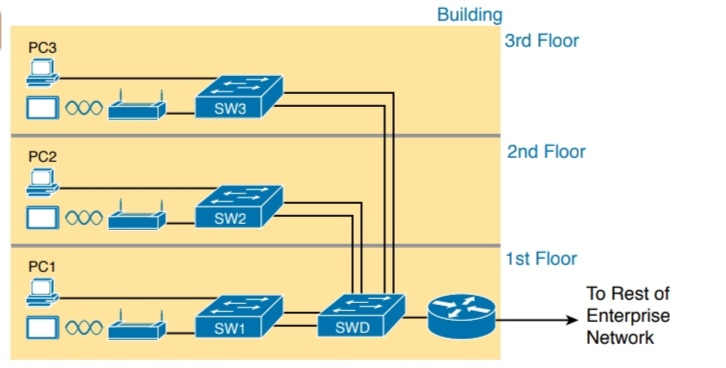


**Terminologia ISO/OSI**:



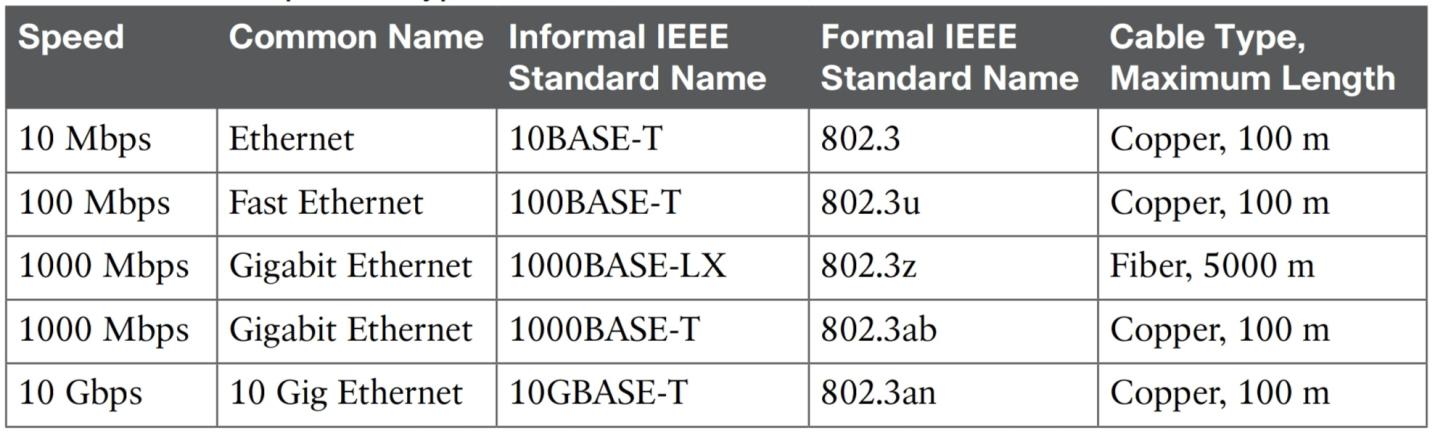
***CAP. 2***

**Typical Enterprise Lan**:



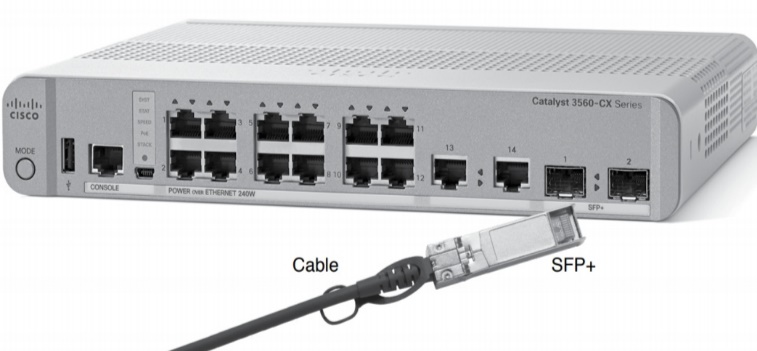
Come possiamo osservare, è presente uno switch per piano a cui sono collegati i dispositivi dello stesso piano. Tutti gli switch sono collegati al Distribution Switch (SWD) al piano terra, collegato a sua volta al router connesso al resto del network.

**Tipi di Standard Ethernet**:



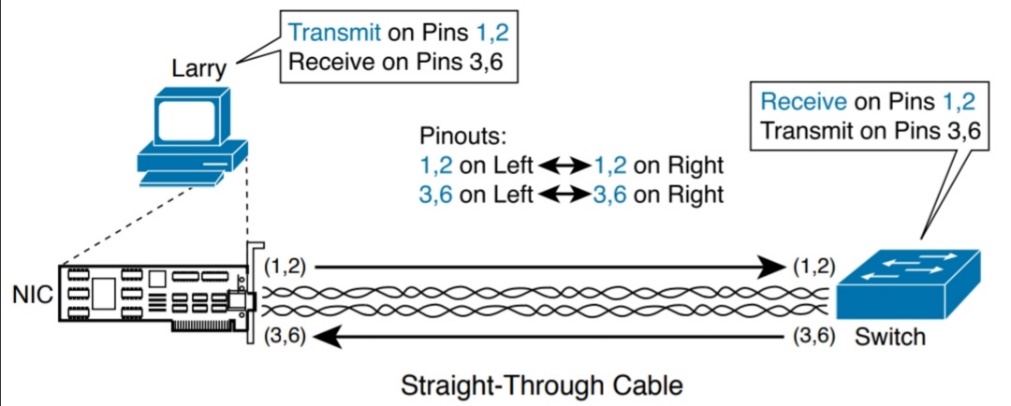
\*collegamenti in rame si fanno con cavi UTP. Se è in rame al fondo “T” sennò un suffisso che include “X”.

Molti cavi UTP usano RJ-45 su entrambe le estremità. l’RJ-45 ha 8 pin per connettore.

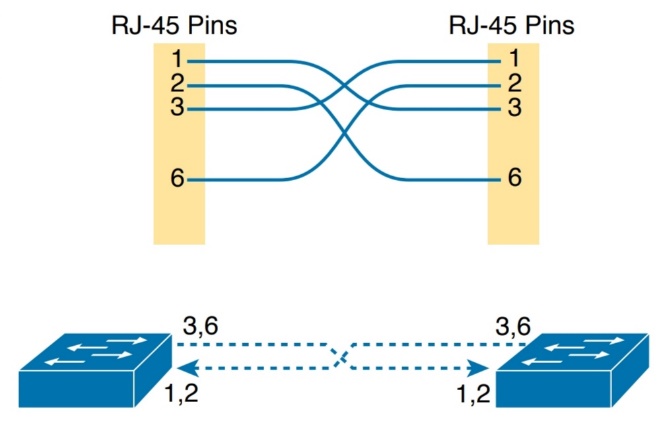
**Gigabit Ethernet Interface Converter** (GBIC): l’originale form factor per un ricetrasmettitore per interfacce gigabit. Più largo degli SFP.

**Small Form Pluggable** (SFP): il rimpiazzo del GBIC usato su interfacce gigabit è più piccolo e prende meno spazio sulla scheda network (NIC) o sullo switch.

**Small Form Pluggable Plus** (SFP+): uguale all’SFP ma usato sulle 10Gb/s interfacce. (il + si riferisce alla maggiore velocità rispetto SFP).

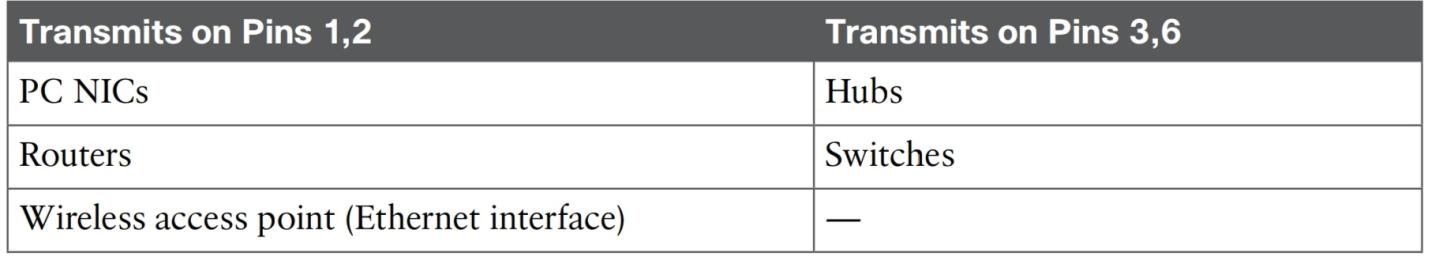


**Straight Through Cable**: gli STC trasmettono per le NIC sui pin 1 e 2 e ricevono sui pin 3 e 6. Gli switch sono configurati per ricevere sui pin 1 e 2 e trasmettere su 3 e 6, per permettere la comunicazione.



**Crossover Cable:** nel caso in cui tu voglia collegare due dispositivi che trasmettono e ricevono sugli stessi pin avrai bisogno di un CC, come nell’immagine.

**10 BASE-T e 100 BASE-T pin utilizzati**

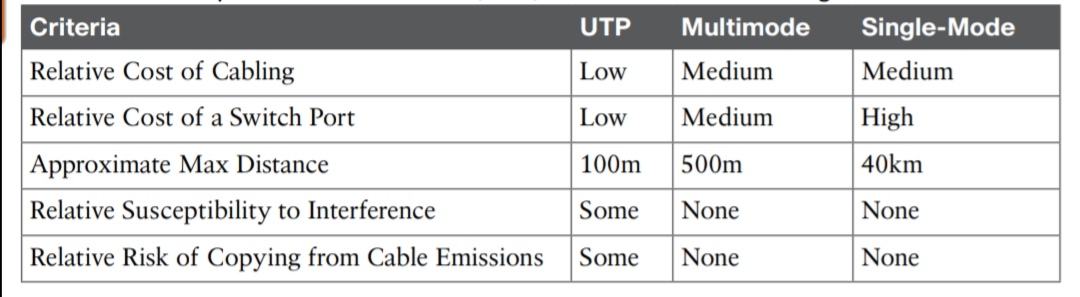


**TIPI DI FIBRA**:

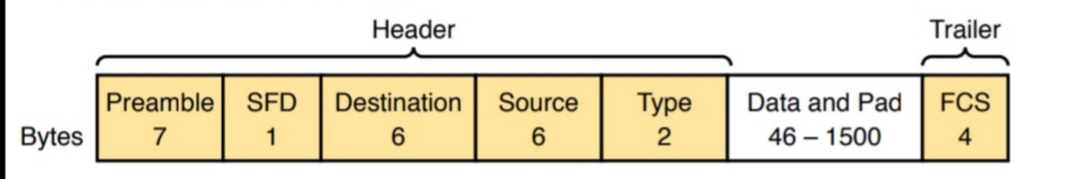
Esistono due tipi di cavi in fibra: multimode fiber e single mode fiber.

Multimode: meno costosa, copre distanze nettamente inferiori (400m), cablaggio medio costoso, SFP.

Single mode: più costosa, copre fino a 40km, cablaggio medio costoso, SFP+.



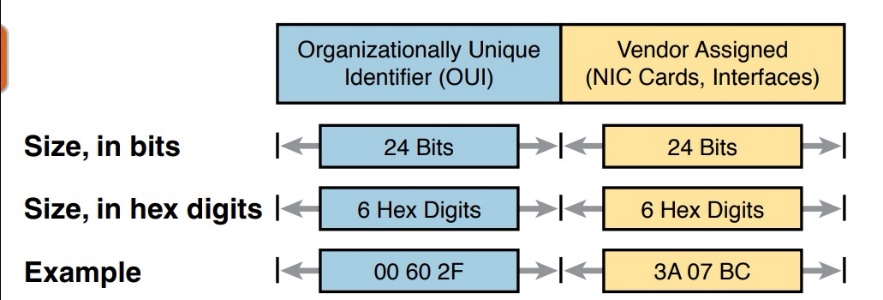
**Vista in dettaglio degli Ethernet Header e Trailer**



|  |  |  |
| --- | --- | --- |
| **Nome** | **Byte** | **Description** |
| Preamble | 7 | Usato per la sincronizzazione. |
| Start Frame Delimiter (SFD) | 1 | Significa che il prox byte è l’inizio del Destination MAC Address. |
| Destination MAC Address | 6 | Identifica il MAC del destinatario. |
| Sorce MAC Address | 6 | Identifica il MAC della sorgente. |
| Type | 2 | Definisce il tipo di protocollo (ipv4 o ipv6) |
| Data and Pad | 46-1500 | Dati dal lvl più in alto di solito un L3PDU. |
| Frame Check Sequence (FCS) | 4 | Fornisce un metodo per il NIC che riceve, per determinare se ci sono stati errori di trasmissione. |

**Indirizzamento Ethernet**

Gli indirizzi ethernet, anche chiamati **Media Access Control** (MAC), sono lunghi 6 byte. Per convenzione i computer scrivono i MAC come un numero a 12 cifre esadecimale.

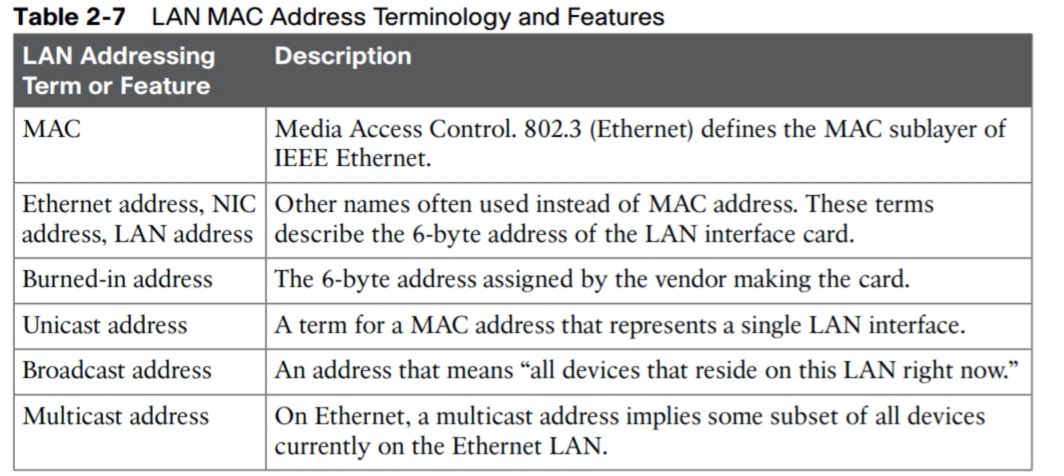
La maggior parte dei MAC rappresentano un singolo NIC o porta ethernet. Questi tipi di indirizzi son chiamati Unicast MAC Address.

Gli indirizzi MAC vengono assegnati a tutti i device ethernet dai produttori, i quali possiedono un codice di 3 byte unico per ogni azienda fornitogli dalla IEEE, chiamato OUI. Gli altri 3 byte li assegnano in modo univoco da loro, creando il MAC Address.

Oltre agli indirizzi Unicast l’ethernet usa anche i **Group Addresses**, che identificano più di una LAN Interface Card. Un frame inviato ad uno di questi indirizzi verrà consegnato a un piccolo gruppo di device connessi alla LAN oppure a tutti. Infatti la IEEE definisce due categorie di Group Addresses:

**Broadcast Address**: i frame inviati a questo indirizzi vengono inviati a tutti i device connessi alla LAN

**Multicast Address**: i frame inviati a un multicast address verranno copiati e inviati a un gruppo di device della LAN.



**Error Detection con FCS:** il mittente applica una complessa formula matematica al frame prima di inviarlo registrando il risultato nel campo FCS, il ricevente applica la stessa formula così riesce a capire se ci sono stati errori durante l’invio del frame e in caso lo scarta.

Half Duplex: il dispositivo deve aspettare ad inviare se sta ricevendo i dati, non può inviare e ricevere contemporaneamente.

Full Duplex: può.

**CSMA/CD**: algoritmo di Collision Detection utilizzato nell’half duplex. Si divide in 3 step.

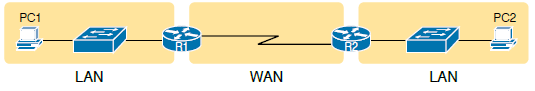
1. Un dispositivo che deve inviare un frame ascolta che il canale sia libero.
2. Quando è libero, manda.
3. Il mittente ascolta il canale mentre spedisce per capire se avvengono collisioni, nel caso tutti i nodi che stanno spedendo dati agiscono cosi:
4. Prima inviano un Jamming signal che avvisa tutti i nodi della collisione.
5. Tutti scelgono indipendentemente un random time da aspettare prima di riprovare l’invio.
6. Ripartono allo step 1.

***CAP. 3***

**Wide Area Networks** (WAN):

Definisce gli standard del lvl Fisico e i protocolli del lvl Data per comunicare su lunghe distanze.

Si divide in due tipi: **leased-line WAN e Ethernet WAN**.

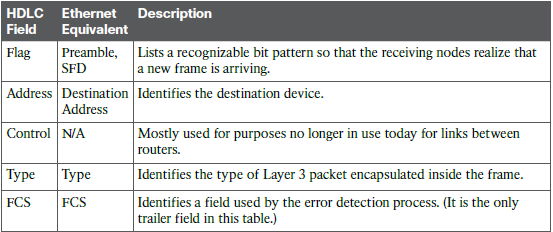
**Leased Line**

PC connesso allo switch crea una LAN, i due router fra loro collegano le LAN creando una WAN.

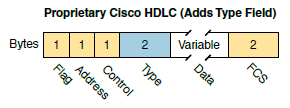
La velocità della Leased Line per spedire i bits è già prestabilita, funziona in full-duplex.

**HDLC Data-Link Dettagli**

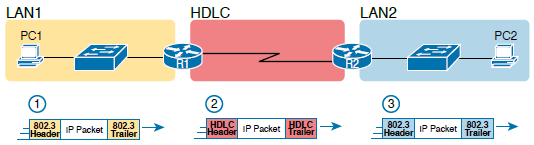
Leased Line garantisce di spedire i bit fisicamente (Lvl Fisico), ma non definisce un protocollo di Lvl Data, per questo motivo alcune organizzazioni hanno creato due protocolli per controllare le LeasedLine: **High-Level Data Link Control** (HDLC) e **Point-to-Point Protocol** (PPP).

**Cose simili tra HDLC e Ethernet**

I **protocolli Data Link** hanno lo stesso ruolo: **controllare il corretto delivery** dei dati sul lvl Fisico.

Es. Protocollo Ethernet Data Link usa un “**destination address field**” per sapere dove spedire e un **FCS** per sapere se i dati sono arrivati correttamente. HDLC funziona in maniera **simile**, l’unica differenza è che HDLC ha meno da fare lavorando in una topologia P-t-P, cioè quando un frame viene mandato da un router, può solamente andare nell’altro router.

HDLC è uno standard di International Org. for Standardization (ISO). ISO **non** ha implementato un Type field, così **Cisco** ha modificato l’HDCL **implementandolo** come in figura. (<)

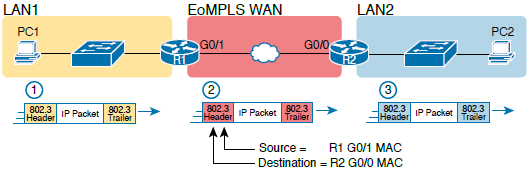
**Logica dei router a Livello Data Link**

**Step 1**. PC1 incapsula il packet in un ethernet frame che ha MAC Address di R1.

**Step 2**. R1 toglie l’ethernet e incapsula il packet nell’HDLC header e trailer e spedisce a R2.

**Step 3.** R2 toglie l’HDLC ed incapsula il packet nell’ethernet frame che ha il MAC del PC2, e glielo manda.

In sintesi, una leased line con HDLC crea una WAN fra due router così che possano spedire dati tra le LAN.

**EoMPLS: Ethernet over Multiprotocol Label Switching** è unatecnologia che può essere usata per creare un servizio Ethernet al cliente.

**Nota**: gli 802.3 H&T sono diversi in ogni step, come spiegato negli step.

**Step. 1:** PC1 incapsula il packet in un ethernet frame che ha il MAC Address di R1.

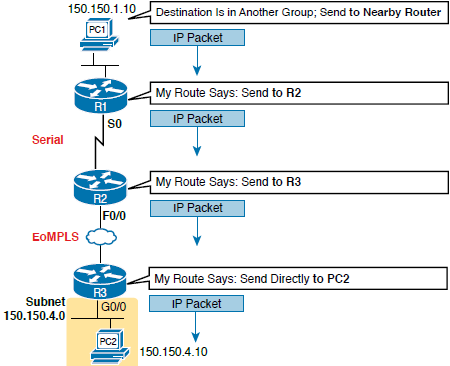
**Step. 2**: R1 rimuove l’ethernet frame ed incapsula il packet con il suo ethernet frame. La destinazione è il MAC di R2, e la sorgente è il MAC di R1. R1 spedisce tramite il servizio di EoMPLS a R2.

**Step. 3**: R2 toglie l’ethernet frame dal packet, lo incapsula con il suo ethernet frame che ha il MAC di PC2, e glielo manda.

**IP Routing**

Tanti modelli di protocollo sono esistiti negli anni, ma il **TCP/IP regna**. A livello **Network** TCP/IP, esistono **due** opzioni sullo lo stesso protocollo per far funzionare il **livello network**: **IP Version 4 (IPv4) e IP Version 6 (IPv6).** Sia IPv4 che IPv6 forniscono le stesse funzioni, ma con qualche dettaglio diverso.

In particolare, il lvl network specifica come i packets devono viaggiare end-to-end su un TCP/IP network.

**Routing Livello Network**

I router e gli end-user (host) collaborano per l’IP Routing. Il sistema operativo dell’host ha il software TCP/IP, gli host usano il sw per decidere dove mandare i packet, spesso al router vicino.

In questo esempio (<) PC1 analizza l’indirizzo di destinazione e capisce che PC2 NON è nella sua stessa LAN. La logica di PC1 gli dice di mandare il packet al router più vicino, chiamato PC1’s Default Router.

Tutti i router hanno una tabella chiamata ***IP routing table*** che contiene gli IP networks e IP subnets. Quando un router riceve un packet, compara l’indirizzo di destinazione del packet con le entrate nella routing table e crea un match. Questo match gli servirà per sapere dove spedire il packet la prox volta.

**Come Network Layer Routing usa LANs e WANs**

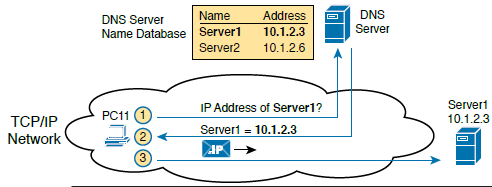
Il lvl network ignora la trasmissione fisica dei dati, per spedirli si rivolge ai protocolli del lvl Data Link che a loro volta si riferiscono al lvl Fisico. Il lvl Data aggiunge una H&T al packet, creando un frame, prima di farlo fisicamente spedire.

**Step. 1**: Data Link Frame Check Sequence (FCS) per essere sicuri che il frame non abbia errori, se ci fossero si scarta il frame.

**Step. 2**: si rimuove la vecchia H&T del Data Link e si lascia solo l’IP packet.

**Step. 3**: compara l’indirizzo di destinazione del packet con la routing table, per trovare il miglior percorso.

**Step. 4**: incapsula il packet con nuovi Data Link H&T e spedisce il frame.

**DNS e ARP**

**Domain Name System** (DNS): PC11 deve connettersi a Server1, così l’user scrive Server1 riferendosi al nome. **Step 1**: PC11 manda un mex al DNS Server.

**Step 2**: DNS Server risponde con l’IP del Server1.

**Step 3**: PC11 può spedire un packet all’indirizzo 10.1.2.3

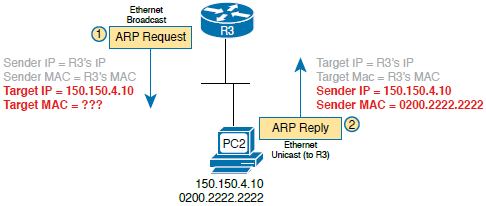
Nota: un DNS non sa tutti i nomi che matchano gli indirizzi, tutti i DNS lavorano insieme scambiandosi dati.

**Address Resolution Protocol** (ARP): Nelle interfacce ethernet, come fa un router a sapere quale MAC usare per la destinazione? Usa l’ARP.

Nelle ethernet LAN, quando un host o router necessita di incapsulare un packet in un nuovo ethernet frame, sa tutto su come fare la header, tranne il MAC di destinazione.

Il TCP/IP introduce l’ARP come metodo con cui ogni host o router possono dinamicamente imparare il MAC di un altro host o router sulla stessa LAN.

Come funziona l’ARP?

E’ un protocollo che include una ARP Request che chiede “è questo il tuo IP? perfavore rispondi con il tuo MAC”.

ARP Reply invece lista l’IP che matcha con il MAC.

Nota che gli host/router ricordano la loro ARP Table, tenendola costantemente aggiornata.

(comando arp -a da prompt si vede la arp table)

PING **(Packet Internet Groper):**

Funzionamento: Il Ping usa Internet Control Message Protocol (ICMP), manda un ICMP echo reques*t* ad un altro IP, il pc ricevente dovrà rispondere con un ICMP echo reply*.* Se la risposta va a buon fine, è tutto ok.

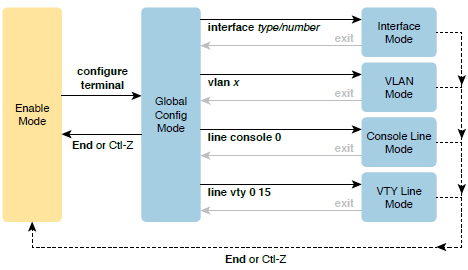
**CAP. 4**

**Telnet & SSH:** metodi per accedere alla CLI da remoto

Telnet: trasmette in chiaro, non crypta.

SSH: Crypta tutto incluse le password.

**Modalità CLI**



**I 4 principali tipi di memoria per gli Switch:**

**RAM:** a volte chiamata DRAM, è usata dallo switch per memorizzare la **running configuration***.*

**Flash Memory**: un chip all’interno dello switch oppure memory card, che contiene il SO di Cisco e che viene utilizzata dallo switch per recuperare la stessa al momento del boot. Può essere usata per contenere altri file, per esempio copie di backup di configurazione.

**ROM**: contiene un programma boot-strap che viene caricato quando lo switch viene acceso. Questo programma prende il SO dalla Flash Memory e lo carica nella RAM.

**MVRAM:** contiene la **startup configuration** che viene usata quando lo switch viene acceso o riavviato.

|  |  |
| --- | --- |
| **Comandi** | **Scopo** |
| undebug all  no debug all | Disabilita tutti i debug in corso. |
| reload | Riavvia lo switch, da enable in su |
| copy startup config running-config | Sovrascrive la running config con la startup |
| copy running-config startup-config | Sovrascrive la startup con la running config |
| show running config | Mostra il contenuto della running config |
| write erase / erase nvram: / erase startup-config | Cancella la startup config |
| quit | Disconnette l’user dalla CLI |
| show startup config | Già lo sai |
| enable | Da i superpoteri |
| Disable | Te li toglie |
| Configure terminal | Solo Mod Enable: attiva la configuration mode |

**CAP. 5**

**Vista sulla logica degli Switch**

Ilcompito principale degli switch è di inviare dati attraverso il network, per farlo svolgono 3 azioni:

1. Decidono quando inviare il frame o quando filtrarlo (non inviarlo), basandosi sul MAC di destinazione.
2. Si preparano ad inviare i frame imparando ed esaminando i MAC delle source di ogni frame ricevuto.
3. Inviano una sola copia del frame alla destinazione, creando un ambiente senza loop tra gli altri switch del network usando l’STP (Spanning tree protocol)

**Known Unicast Frame**

Quando si invia un frame lo switch controlla il MAC di destinazione e lo compara in quelli della MAC Address Table che gli dice su quale porta farla uscire per arrivare alla destinazione voluta.

**Unknown Unicast e Broadcast Frame**

Quando allo switch arriva un frame con MAC Add di destinazione a lui sconosciuto, invia copie del frame su tutte le porte eccetto quella da cui l’ha ricevuto. Questo è il **flooding**. L’idea è che se non sai dove mandarlo, lo mandi a tutti. Sarà il dispositivo corretto a rispondere, lo switch lo salva nella MAC add table, gli altri lo scartano.

**Evitare Loop usando STP**

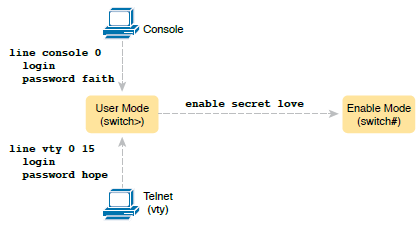
La terza feature primaria di uno switch è la prevenzione dei loop, che si evitano con STP.

**Come funziona STP**? Per evitare i loop l’STP blocca alcune porte random dall’inviare frame di modo che rimanga solo un percorso attivo tra ogni segmento di LAN.

**Aging Time**: timer utilizzato all’interno della MAC table, di default 300s, associato ad ogni entries che viene rinnovato ogni volta che la entries viene utilizzata. Se il timer scade, la entries viene cancellata.

|  |  |
| --- | --- |
| **comandi** | **scopo** |
| Show mac address-table | Mostra la MAC table |
| Show mac address-table dynamic | Mostra tutti I MAC imparati dinamicamente |
| Show mac address-table dynamic vlan “*vlan-id”* | Mostra tutti I MAC imparati dinamicamente in quella vlan |
| Show mac address-table dynamic address “*mac* *address*” | Mostra il MAC dinamicamente imparati con quel MAC address |
| Show mac address-table dynamic interface “*interface id*” | Mostra tutti I MAC dinamicamente imparati associati a quell’interfaccia |
| Show mac address-table count | Mostra il numero di entries nella MAC table, e il num totale di slot vuoti della table |
| Show mac address-table aging-time | Mostra l’aging timeout dei MAC inattivi |
| clear mac address-table dynamic | Pulisce la table dei MAC imparati dinamicamente |
| Show interfaces status | Crea un elenco delle interfacce con i relativi stati e le info operative per ognuna. |

**CAP. 6**

**Sicurezza per gli Switch**

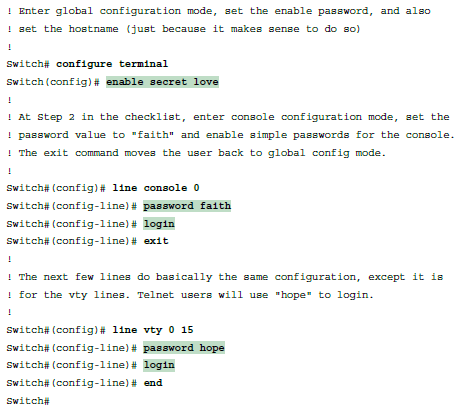
Ogni user di default è abilitato ad entrare nello switch in tutte le mode e configurare lo switch. Per ovviare a questa problematica, si possono settare password e user come nell’esempio.

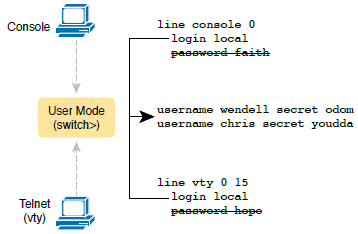
**enable secret** permette di settare una password criptata.

**Settare la Console Password:**

1. Usare **line con 0** per entrare in configuration mode.
2. Usare **password *pwd-value*** per settare la password.
3. Usare il comando **login** per abilitare la console pwd.

Per settare la **Telnet** pwd usare il comando **line vty 0 15** per entrare in configuration mode, il resto è uguale al punto 2 e 3 sopra.

**Come settare la console e telnet password:**

**Configurare Switch con Username Locali**

1. Usare il comando:

**username** *nome* **secret** *password*

per aggiungere un utente locale allo switch.

1. Configura la console per usare gli user/pwd locali:

A. usare **line con 0** per entrare in configuration mode.

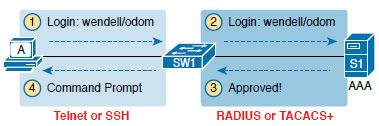
B. usare **login local** per abilitare la console al username e pwd, con un check nella lista di utenti locali.

C. (opzionale) si può usare il comando

**no password** per rimuovere tutte le shared pwd, per tenere sicura la config dello switch.

1. Configura Telnet per usare gli user/pwd locali:
2. Usare **line vty 0 15** per entrare nella vty configuration mode
3. e C. sono uguali al punto 2.

**Authentication, Authorization and Accounting Server (AAA)**

Molte volte capita di dover **cambiare** password o **disabilitare** utenti, e via dicendo. Cisco usa un server esterno, chiamato AAA.

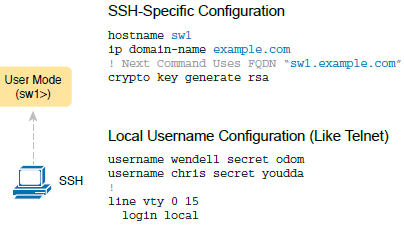
**Come funziona**?

Quando si usa AAA per l’autenticazione, lo switch o router manda un messaggio al server AAA chiedendo se l’username e password sono abilitati, l’AAA risponde come nell’immagine.

La connessione tra **user** e **switch** è in **Telnet o SSH**, mentre tra **switch** e **server** **AAA** si comunica in **Radius o Tacacs**+, protocolli che criptano le pwd mentre viaggiano nel network.

**Accesso Remoto Sicuro in Secure Shell (SSH)**

Telnet ha lo svantaggio di far scambiare messaggi in chiaro, comprese le pwd. SSH invece cripta tutti i dati trasmessi tra client e server, proteggendoli.

SSH usa il **Fully Qualified Domain Name** (FQDN) dello switch per creare la SSH encryption key, FQDN viene creato dall’hostname e domain name dello switch. L’ultimo comando **crypto key generate rsa** genera la SSH encryption keys.

(opzionale) Siccome Telnet è poco sicuro, si può decidere di abilitare solo SSH nella *vty line configuration* con il comando **transport input ssh.**

Di **default** gli switch sono settati con **transport input all** o **transport input none**, in quel caso devi configurare tu quello che vuoi. Inoltre supportano **entrambe le versioni** di **SSH**, per decidere di abilitare solo la v2 ad esempio, si usa **ip ssh version 2** in global configuration mode.

Esistono due comandi principali che forniscono info su SSH nello switch, il primo per vedere lo status del server SSH e si usa **show ip ssh,** il secondo è **show ssh,** lista info sui client SSH al momento connessi allo switch.

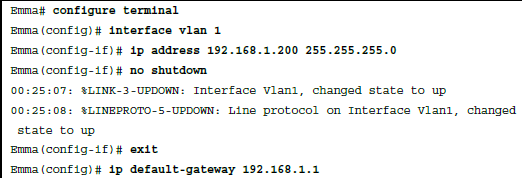
**Configurare IPv4 sullo Switch**

**Step 1.** Usare **interface vlan 1** (dando per scontato sia sulla vlan1) in global config mode per settare la VLAN 1 configuration mode.

**Step 2**. Usare **ip address** *ip-address mark* per assegnare un IP e una maschera.

**Step** **3**. Usare **no shutdown** per abilitare la VLAN 1.

**Step 4**. Usare ip **default-gateway** *ip-addres* in global config mode per configurare il default gateway.

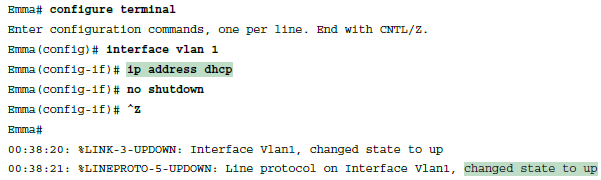
**Step 5**. (opzionale) usare **ip name-server** *ip-address1 ip-address2*… per configurare lo switch nell’usare il DNS per matchare i nomi coi loro rispettivi IP Address.

**Abilitare/Disabilitare Interfaccia su Switch**

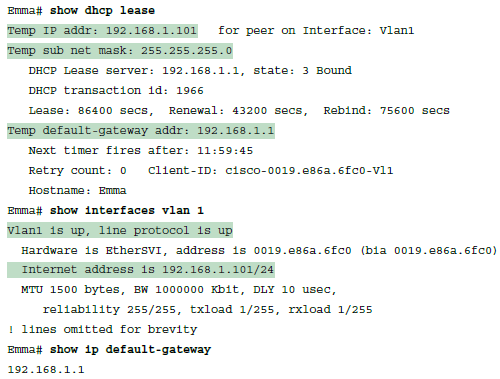
Per **abilitare** una interfaccia su uno switch, si usa **no shutdown**. Per **disabilitarla** invece **shutdown**.

**Configurare uno Switch per imparare il suo indirizzo IP con DHCP**

Uno switch può usare anche il **Dynamic Host Configuration Protocol** (DHCP) per imparare dinamicamente le sue impostazioni di IPv4. Praticamente tutto quello che devi fare è dire allo switch di usare il DHCP nell’interfaccia e abilitare l’interfaccia.

Step. 1: si entra nella VLAN 1 config mode usando **interface vlan 1**, e si autorizza l’interfaccia con **no shutdown**.

Step 2: Si assegna un indirizzo IP e una mask usando **ip address dhcp**.

**Verificare IPv4 sullo Switch**

Esistono diversi modi per verificare la configurazione IPv4.

Si può dare **show running-config** che mostra la config corrente, oppure dando **show interfaces vlan*****x*** vengono mostrati l’indirizzo IP e la mask, infine si può usare **show dhcp lease** per vedere i parametri temporanei dhcp (nota che lo switch non memorizza la config dhcp).

Se imposti l’indirizzo IP staticamente verrà sempre listato, se invece lo sceglie il DHCP dinamicamente ed il DHCP fallisce nell’assegnarlo, il comando **show interfaces vlan 1** non lo mostrerà, questo comando tra l’altro non ti dice se l’IP è statico o dinamico.

Infine, quando **sbagli** un comando sulla CLI il DNS si impegna per cercare quel comando nei propri server, e ci mette 1 min di timeout per ridarti il controllo della CLI. Per disabilitare ciò si usa **no ip domain-lookup**.

**Logging synchronous**: dice allo switch di aspettare che tu abbia finito per dare lo show.

**Exec-time**: setta un timer “inactivity timeout”, dopo un tot di inattività scelto dall’utente IOS chiude la sessione dell’utente.

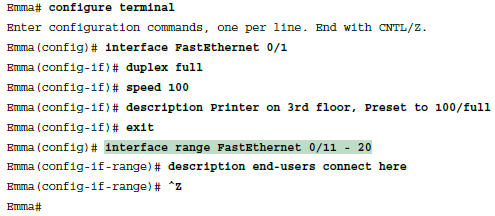
**CAP. 7**

IOS usa il termine “**interfaccia**” riferendosi alle porte fisiche del device usate per spedire e ricevere dati.

Ogni interfaccia può essere configurata con diverse impostazioni, ogni interfaccia ha le sue.

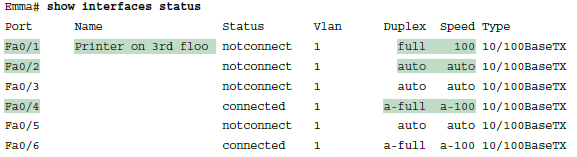
Esistono tre tipi principali di configurazioni per interfaccia (porta): **velocità (**speed**), duplex e description**.

**Configurazione Speed, Duplex e Description**

I dispositivi che supportano diverse velocità autonegoziano quale usare fra loro, per il miglior risultato.

In ogni caso si può settare: ***duplex*** {auto | half | full} e ***speed*** {auto | 10 | 100 | 1000}.

La **description** ti permette di aggiungere una descrizione all’interfaccia. Semplice.

Nell’immagine le regole settate valgono **solo** per l’interfaccia FastEthernet 0/1, e nessun’altra.

Per vedere i dettagli di ogni interfaccia si usa **show interfaces status**.

Anche se ci fosse solo una riga di output.

Per la Fa0/1: lo “status” è impostato su **notconnect** nel senso che al momento non sta lavorando.

Per la Fa0/2: default config, speed e duplex sono su “auto”, significa che cercherà di negoziare la velocità e la connesione, quando la porta sarà “up”.

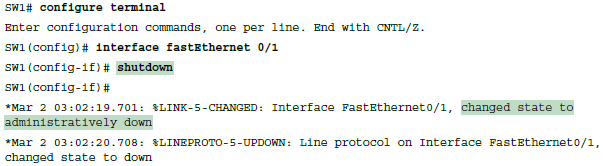
Per la Fa0/4: similmente a Fa0/2 è settata di default, la a- davanti su speed e duplex significa che sono parametri già autonegoziati.

**Configurare più interfacce con “Interface Range”**

Nella penultima figura si nota un comando **interface range** che si usa per risparmiare tempo nella configurazione delle interfacce. Il comando, nell’esempio, dice all’IOS di applicare quella configurazione dall’interfaccia FastEthernet 0/11 alla 0/20.

**NB**: Questo è fattibile solo se le interfacce sono dello stesso tipo e sono numerate consecutivamente.

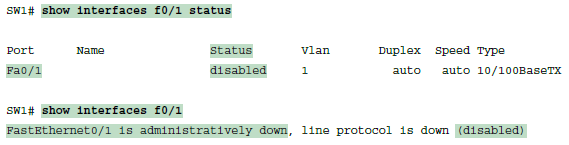
**Controllare le interfacce “amministrativamente”**

E’ necessario poter decidere quali interfacce devono essere **up** e quali **down** senza il bisogno di farlo fisicamente, per questo Cisco usa due comandi per abilitare o disabilitare una porta: **no shutdown e shutdown.**

Nell’immagine (>) SW1 sta lavorando sulla F0/1, l’user si connette alla console e la **disabilita** con “**shutdown**”, la console restituisce con un messaggio in cui avvisa che la porta è “amministrativamente” down.

Per **riportarla su** l’unica cosa che devi fare è lo stesso procedimento ma con “**no shutdown**”.

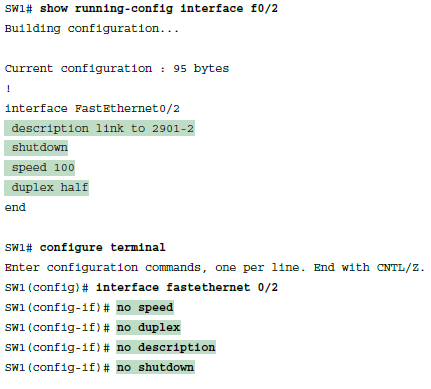
Il comando **show interfaces status** lista un solo output, quando la porta è down l’output è “**disabled**”.

Invece il **show interfaces** nella riga sotto lista molte righe di output (tagliate dallo screen) per interfaccia, dando molti più dettagli.

**Rimuovere Configurazioni con il No Command**

Se hai settato una configurazione con determinati comandi, alle volte mettendo “**no**” davanti a quel comando si può cancellare quella configurazione.

Per esempio se hai settato la **speed 100**, dando **no speed** la velocità torna in default (**auto**). Stessa logica per il duplex e per la description.

Nell’immagine (<) si vede la porta F0/2 dello switch SW1 configurato con **speed 100, duplex full e description link to 2901-2, poi shutdown**.

(<) Qui la configurazione viene tutta tolta.

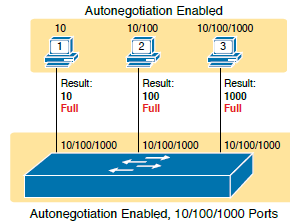


Infatti quando si da il **show running-config** (>) il comando non restituisce nulla.

**NB**: Tipicamente il comando **show running-config** e **show startup-config** **NON** mostrano la configurazione di default, il fatto che l’output sia vuoto è segno che la configurazione è tornata in default.

**Autonegoziazione**

Come funziona? La IEEE ha uno standard (**802.3u**) dove due device si mandano segnali elettrici per capire a quale frequenza possono comunicare.



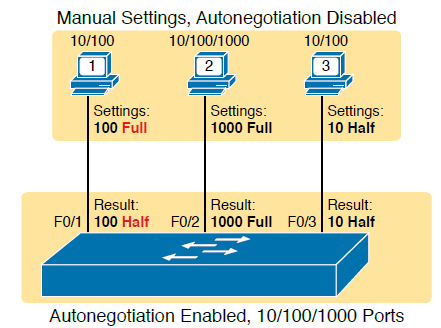
Per esempio, nel primo caso l’interfaccia dello switch chiede al PC1 se possono andare a 1000Mbps ma la NIC risponde che al max va a 10Mbps, di conseguenza scelgono la maggior velocità supportata da entrambi e la miglior duplex possibile. Idem per PC2 e PC3.

Quando l’autonegoziazione viene disabilitata su un dispositivo che viene collegato ad un altro device che invece ce l’ha attiva, la connessione tra i due potrebbe non funzionare o funziona male.

La IEEE definisce alcune regole che i device devono seguire quando **l’autonegoziazione fallisce**:

1. usare la velocità più **bassa** possibile.
2. se la tua **velocità è > 100** usare il **full duplex**, se è **< o =** usare **half**.

Gli switch possono prendere decisioni migliori rispetto alle regole imposte sopra, perché possono captare la velocità usata dagli altri device quando l’autonegoziazione fallisce. Di conseguenza si comportano in modo leggermente diverso:

1. captano la velocità dell’altro device senza l’autonegoziazione, ma se falliscono, rispettano la regola.
2. uguale al 2. sopra.

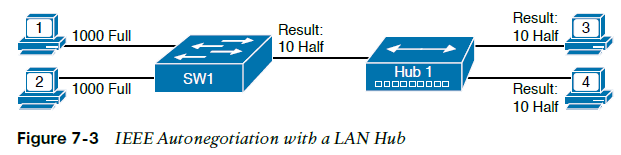
Nel primo caso, lo switch ascolta i segnali elettrici del collegamento e capisce che il PC1 va a **100Mbps**, di conseguenza imposta la velocità a 100Mbps e, ovviamente, usando le **IEEE autonegoziation default** setta in **half duplex**.

Invece nel secondo caso usa gli stessi step logici del primo caso, ma sceglie **full-duplex** perché la velocità è **1000Mbps**.

Nel terzo caso sceglie 10Mbps perché capta la velocità del PC3, va in **half-duplex**.

Il PC1 mostra uno sfortunato risultato, un **duplex mismatch** poiché i due nodi comunicano alla stessa velocità ma il PC1 usa una full-duplex che non permette l’utilizzo della **CSMA/CD** logic e quindi manda i frame continuamente. Lo switch, invece, è in half-duplex che permette l’uso della collision detection.

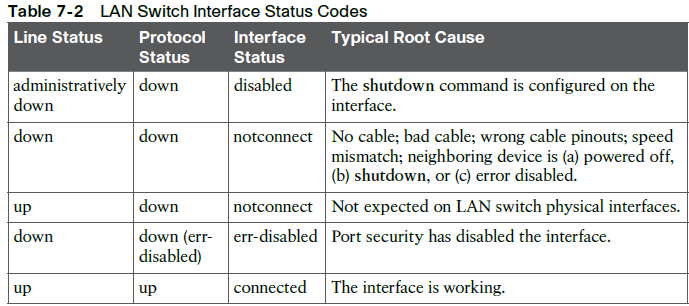
Come risultato lo switch crederà che sia avvenuta una **collisione** mentre non sarà effettivamente successo nulla. Lo switch di conseguenza interromperà la trasmissione, e inizierà a rinviare i frame, e cosi via. Infine il collegamento funziona, ma di merda.

**NB**: gli hub non usano l’autonegoziazione quindi tutti i device che lo usano funzioneranno a 10Mbps ed in half-duplex perché gli hub usano la CSMA/CD.

**Interface Status Code e Nonworking States**

I comandi **show interface** e **show interface description** elencano i due **status code** chiamati **line status e protocol status**.

**Line status** si riferisce generalmente a come funziona il Layer 1, mentre il **protocol status** si riferisce al Layer 2.



**Problemi del Layer 1 su interfacce up/up**

Gli switch possiedono diversi interface counter visibili con il comando **show interface.**

I counter possono aiutare ad identificare diversi problemi che avvengono sull’interfaccia in connect state.

**Runts**: frame che non raggiungono la minima dimensione richiesta, possono essere causati da collisioni.

**Giants**: frame che eccedono la massima dimensione di un frame.

**Input error**: è la somma di altri counter, tipo runts, giants, no buffer, CRC, frame ecc.

**CRC**: frame ricevuti che non hanno passato il controllo FCS, possono essere causati da collisioni.

**Frame**: frame ricevuti che possiedono un formato illegale, possono essere causati da collisioni.

**Packets output**: numero totale dei packets inviati dall’interfaccia.

**Output errors**: numero totale di packets che la porta switch ha provato a trasmettere, ma senza successo.

**Collisions**: contatore di tutte le collisioni avvenute prima della trasmissione dell’ultimo byte del frame in invio.

**Late collisions**: una sotto categoria di collisioni che accade dopo la trasmissione del 64 byte del frame.

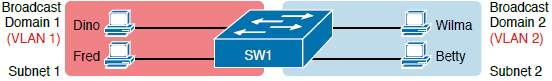
**CAP. 8**

**Virtual LAN e differenze con Virtual-LAN**

**LAN**: include tutti i device, server, switch, router, cavi e access point in una location. Insomma, include tutti i **device nello stesso dominio di broadcast**.

La **logica di uno switch** è che tutte le **interfacce** sono nello **stesso dominio** di broadcast, di conseguenza quando si spedisce un **frame**, viene inoltrato a **tutte** **le altre porte**. Dunque, per creare due LAN separate, devi avere **due switch**, come in figura.

Il vantaggio delle **VLAN** invece, è di poter compiere lo stesso obiettivo nella figura sopra, ma con un unico switch, assegnando le varie interfacce ad un dominio piuttosto che all’altro.

Ogni singolo dominio creato dallo switch è chiamato **Virtual LAN**. Nella figura sotto lo switch ha creato due VLAN separate, di conseguenza Dino non potrà mai spedire un frame a Wilma o Betty. In breve, un **mex di broadcast** verrà ricevuto solo dai device nella stessa **VLAN**.

**Vantaggi delle VLAN in breve:**

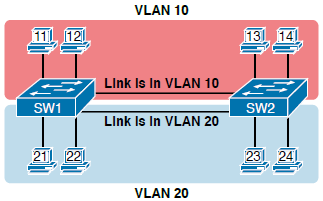
1. aumenta la prestazione dell’host, riduce il numero di device che ricevono il mex di broadcast.

2. aumenta la sicurezza, riduce il n. di host che ricevono quel frame. Inoltre applica delle policies per VLAN.

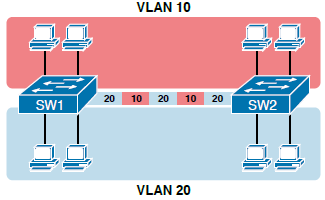
3. crea design più flessibili ed organizzati.

**Creazione Multiswitching VLAN con Trunking**

Il **trunk** è il metodo che permette di far passare **più VLAN dalla stessa porta**, ottimizzando tutto il sistema, l’opposto di trunk è **access**, dove è configurata **una sola VLAN** per interfaccia.

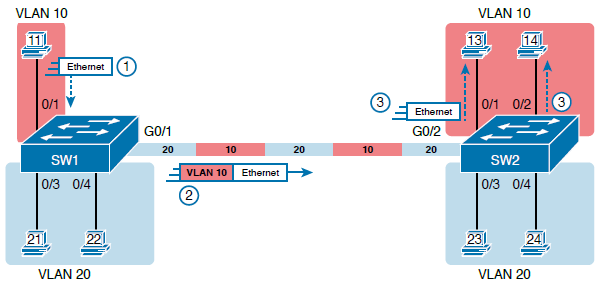
Il trunking porta lo switch ad usare un processo chiamato **VLAN Tagging**, ovvero lo switch sorgente aggiunge un ulteriore header al frame prima di spedirlo in trunk. In questa header viene specificato il **VLAN ID**, così il destinatario sa a quale VLAN appartiene il frame.

In figura (<) il multiswitching VLAN funziona, ma è privo di trunk, di fatti per supportare le VLAN avrai bisogno di un collegamento fisico (physical link) per ognuna di esse. Se dovessi creare 10 o 20 VLAN, avrai bisogno di 10 o 20 cavi, collegati alle 10 o 20 porte di ogni switch.



**Concetto di VLAN Tagging**

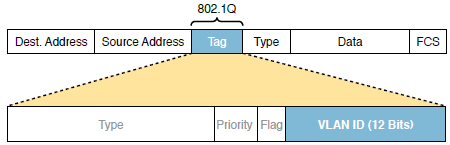
Il VLAN trunking crea un link tra gli switch che può supportare quante VLAN vuoi. Essendo in trunk, gli switch trattano il link come parte di ogni VLAN, e nel frattempo il link si occupa di superare il traffico fra le varie VLAN. (>)

**In breve**: il trunking permette agli switch di mandare i dati su multiple VLAN dallo stesso link fisico, aggiungendo una header identificativa al frame.

1. PC11 deve spedire in broadcast sulla VLAN 10, così manda a SW1 che aggiunge un header VLAN ID.
2. Viene spedito in trunk e ricevuto da SW2.
3. SW2 rimuove la header e spedisce a tutti i device della VLAN 10.

Altro esempio: PC 21 deve spedire in broadcast in VLAN 20. SW1 spedirà subito a Fa0/4 che sono nella stessa VLAN e poi manda a Gi0/1 con l’header aggiuntiva. SW2 toglie la header e spedisce solo a Fa0/3 e 4.

**Protocolli Trunk: 802.1Q e ISL**

La **IEEE ha creato l’802.1Q** il più usato come protocollo trunk, **Cisco ha creato ISL** anni prima. Cisco supporta entrambi i protocolli nei propri switch.

Entrambi i protocolli aggiungono al frame il VLAN ID ma con qualche differenza: **802.1Q** aggiunge un extra 4-byte nel ethernet header del frame originale, il VLAN ID è di 12 bits.

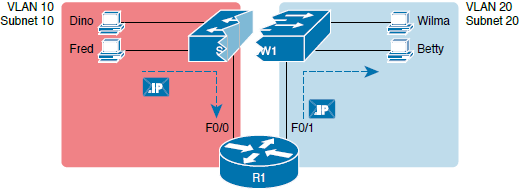
Gli switch Cisco spezzano il **range** di VLAN ID, che supporta **212** VLAN cioè un range di 1-4094 (con due valori riservati 0 e 4095), in **normal range** che va **1 a 1005** e possono essere usati da **tutti** gli switch **ed extended range** che va da **1006 a 4095**  e possono essere usati solo da **alcuni** switch.

L’802.1Q definisce un **VLAN ID speciale** per ogni trunk, chiamato ***Native VLAN***. Per definizione l’802.1Q non aggiunge una header al frame nella VLAN nativa, ovvero quando uno switch riceve un frame privo di 802.1Q header, lo switch ricevente sa che il frame è parte della VLAN nativa.

NB: Ovviamente entrambi gli switch devono essere d’accordo su quale VLAN è quella nativa.

**Scambio dei dati tra VLANs**

Se crei una VLAN che contiene più VLAN avrai bisogno di far comunicare tutti i device fra loro.

Ed è per questo che per far comunicare le VLAN tra loro si deve usare un **router** (come in figura <) o uno **switch Layer 3** che performi come un router.

Lo **switch** così può mandare dati **dentro** **una VLAN**, e il **router** può mandare i dati **tra le subnet**.

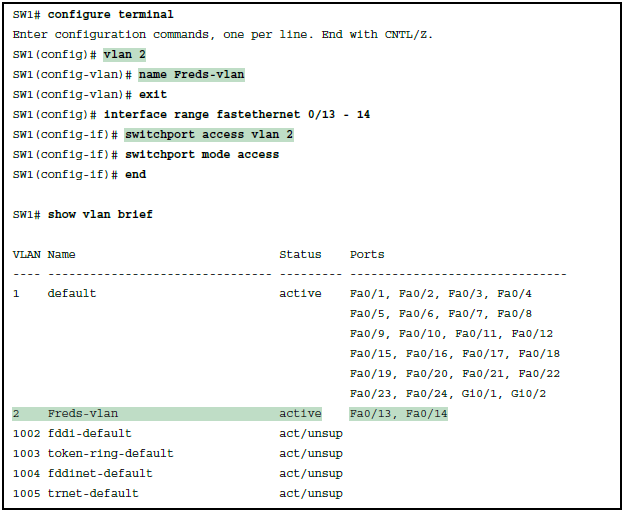
Nell’immagine, semplicemente, Fred dalla VLAN 10 routa ip packet che deve andare da Betty nella VLAN 20, lo switch manda due diversi Layer 2 ethernet frame: uno da Fred a R1 F0/0 e l’altro da R1 F0/1 a Betty.

Il sistema funziona, ma ci sono diverse altre soluzioni per soddisfare questo schema.

**Creare una VLAN ed Assegnare la VLAN ad una Interfaccia**

Per creare una VLAN lo switch deve essere configurato di modo da credere che esista realmente, in più deve avere le interfacce in access e non in trunk, oppure trunk che supportano le VLAN:

Step. 1: **Configurare una nuova VLAN**

1. Da configuration mode, usa **vlan** *“vlan-id”* in global config mode per crearla e muovere l’utente in VLAN config mode.
2. (opzionale) Comando **name** *“name”* in VLAN config mode per listare un nome per la VLAN, se non è configurata sarà VLAN”*ZZZZ”* dove ZZZZ è il numerico della VLAN.

Step. 2: **Accessi Interfaccia**

1. Usare **interface** “*numero*” in global config mode per passare in interface config mode per ogni interfaccia desiderata.
2. Usare **switchport access vlan** “*id-number”* in interface config mode per specificare il numero di VLAN associate a quell’interfaccia.
3. (opzionale) **switchport mode access** setta l’interfaccia sempre in access.

NB: le porte di default vengono settate dallo switch sulla VLAN 1.

**Configurazione VLAN Trunking**

La configurazione può essere predefinita oppure essere negoziata tra gli switch:

1. **Il tipo di trunking**: IEEE 802.1Q, ISL, oppure scegliere quale usare (per gli switch che supportano entrambi).
2. **Administrative mode**: sempre in trunk, mai in trunk, oppure scegliere quando trunkare.

Gli switch che supportano sia 802.1Q che ISL per negoziare usano il Dynamic Trunking Protocol (DTP), se entrambi supportano entrambi i protocolli, useranno l’ISL, sennò useranno quello che hanno in comune.

Per configurare il tipo o autorizzare il DTP a negoziare si usa:

**switchport trunk encapsulation {dot1q | isl | negotiate}**

**Comandi Trunk Administrative Mode con la Switchport Mode**

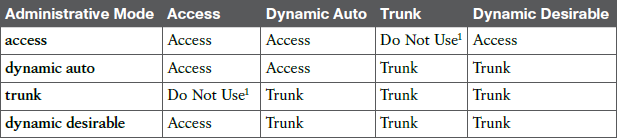
**access** > agisce sempre in access, mai in trunk

**trunk** > agisce sempre in trunk

**dynamic desirable** > inizia la negoziazione e scelgono se usare il trunk o meno

**dynamic auto** > aspetta passivamente che un altro switch mandi un mex per negoziare, lui risponde e negoziano

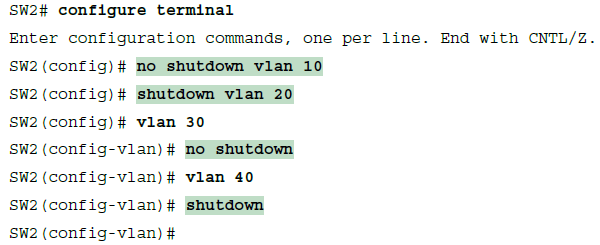
NB: Se due switch vogliono comunicare ma sono entrambi in **dynamic** **auto** nessuno dei inizierà la negoziazione. Quando uno switch viene cambiato in **desirable** mode, allora iniziano a negoziare.



Per disabilitare la negoziazione del DTP usare **switchport nonegoziate**.

**Access VLANs Undefined or Disabled**

Gli switch non mandano frame per VLAN che

1. Non sono configurate o non sono state imparate dal VTP
2. La VLAN è conosciuta ma è disabilitata (shut down)

Per il punto **A** se una VLAN non è configurata, basta configuarla. Per fare un check delle VLAN usare **show vlan** che mostrerà tutte le VLAN conosciute dallo switch, più lo status di alcune, **active** o **act/lshut.**

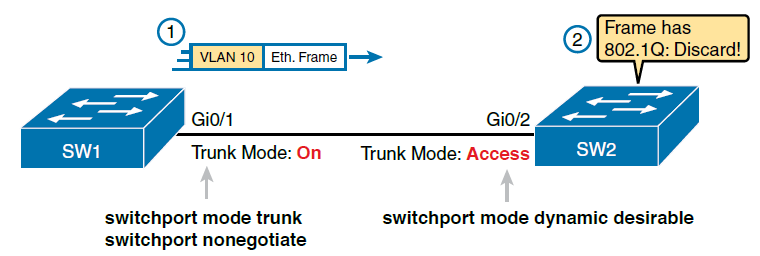
Per il punto **B** invece, dare shutdown a una VLAN significa **disabilitarla** su quello switch, così lo switch **non manderà frame** a quella VLAN.

**Trunking Mismatch**

Il trunking può essere configurato correttamente di modo che due switch siano in trunk, ma alle volte possono crearsi problemi: se uno switch è in trunk e l’altro no, oppure se nessuno dei due è in trunk.

La **configurazione sbagliata** più comune è settare entrambi gli switch su **switchport mode dynamic auto**, gli switch aspetteranno entrambi passivamente che l’altro device proponga di trunkare.

**Un altro esempio**: uno switch in trunk spedisce il frame via link, i frame crossano il link ma l’altro switch NON è in trunk, di conseguenza scarterà tutti i frame che hanno una VLAN tag nell’header. In questo caso le porte risultano entrambe up/up o connected.

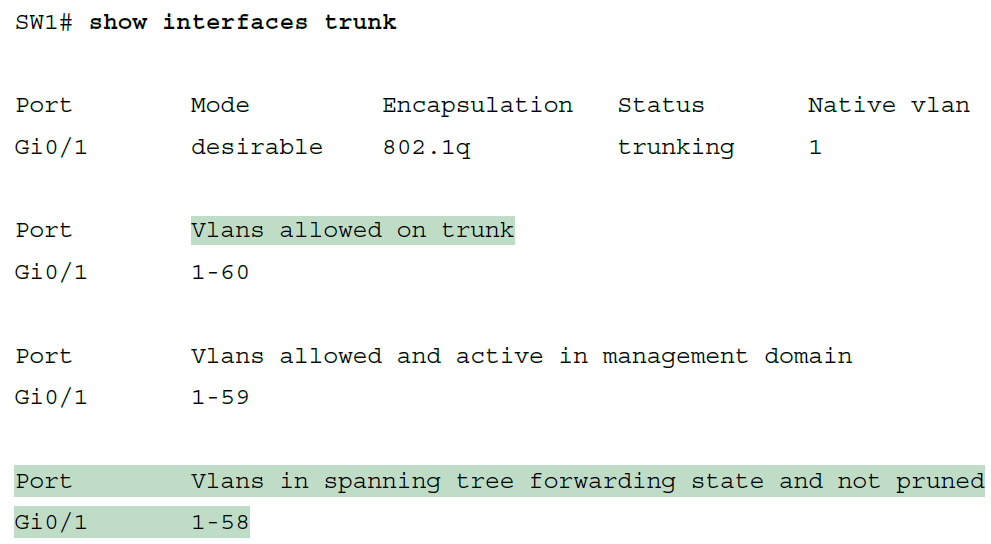
L’immagine mostra la configurazione sbagliata: SW1 abilita il trunk con **switchport mode trunk** ma disabilita anche il **Dynamic Trunk Protocol** usando **switchport nonegoziate**. SW2, a sua volta, avendo il DTP attivo cerca di negoziare, ma SW1 ce l’ha disattivato, dunque SW2 non potendo negoziare sceglie di **non trunkare**.

In sostanza, SW1 potrebbe mandare un frame sulla VLAN 10 (per esempio), ma SW2 vedrebbe tutti i frame che arrivano con la **802.1Q header** come “**illegali**”, perché SW2 tratta la sua porta, la Gi0/2 come una **access port**, così **scarta** tutti i frame con la **802.1Q header**.

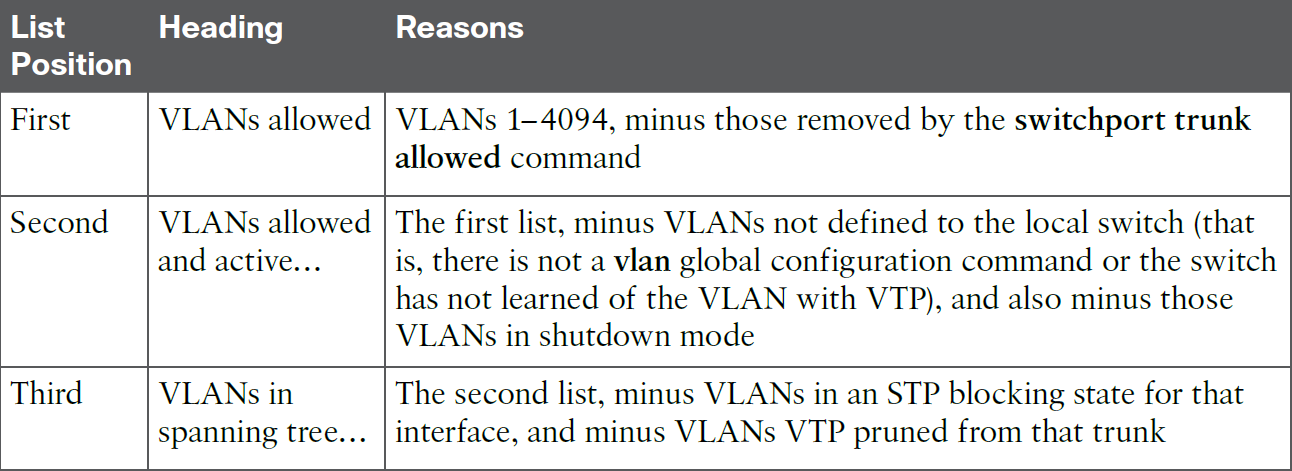
Per fare un check e prevenire queste situazioni i comandi migliori sono **show interfaces trunk** e **show interfaces switchport**. Sappi che gli switch non ti allertano su questi errori di configurazione.

**Limitare traffico dati nelle interfacce VLAN**

Per limitare il traffico dei dati su uno switch si usa **switchport trunk allowed “*vlan-id”.***

**Esempio**: in uno switch dove sono state configurate la VLAN 1 fino a 100, di default lo switch permetterebbe il traffico su tutte e 100 le VLAN; se volessimo **limitare** questo traffico dalla **VLAN 1 a 60** useremmo **switchport trunk allowed vlan 1-60.**

Il comando **show interfaces trunk** crea 3 liste con criteri differenti in base allo stato delle VLAN, come nell’immagine. (>)

Le liste di VLAN con lo **show interface trunk**:

**Mismatch Native VLAN: VLAN Hopping**

E’ possibile settare la VLAN nativa ad altre VLAN con il comando **switchport trunk native vlan “*id-vlan”.***

Quando uno switch ha una native VLAN settata in modo diverso dagli switch “vicini”, ciò farà saltare il frame da una VLAN ad un'altra.

**Esempio**: se SW1 spedisce un frame su Native VLAN 1 su una trunk 802.1Q, SW1 ovviamente non aggiungerà una VLAN header al frame, com’è normale per le native VLAN. SW2 riceverà il frame senza header e penserà che il frame è parte della sua native VLAN, ma se SW2 è configurato con Native VLAN 2, cercherà di spedire il frame sulla VLAN 2.

Questo è il **VLAN Hopping**, mandare un frame destinato ad una VLAN in un’altra VLAN.

**CAP. 9**

**Spanning Tree Protocol (STP) e Rapid STP (RSTP)**

1Senza meccanismi come **STP o RSTP** una LAN con link ridonanti farebbe **loopare** i frame per un tempo **indefinito**. Con STP/RSTP attivo gli switch bloccano alcune porte dall’invio di frame.

Le interfacce da bloccare sono scelte dall’STP ed RSTP tenendo a mente **due goals**:

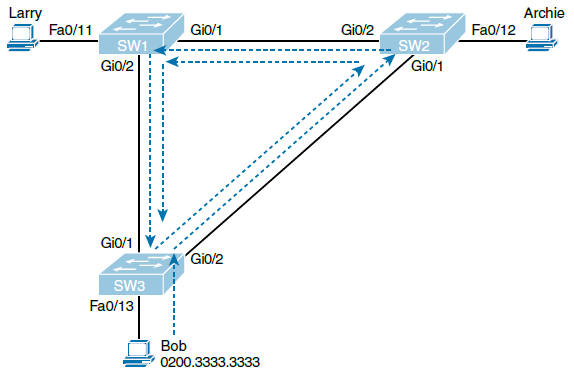
1. Vengono bloccate alcune porte, ma non tutte, i device della VLAN possono comunque mandare frame a tutti gli altri device della VLAN.
2. I frame hanno vita breve e non loopano chissà quanto.

STP/RSTP prevengono il loop dei frame aggiungendo un check su ogni interfaccia prima che lo switch la usi per mandare o ricevere frame.

Se STP/RSTP forwarding state è attivo sulla porta, l’uso è normale. Invece se la porta è su STP/RSTP bloking state blocca tutto il traffico non spedendo ne ricevendo su quella porta.

**Broadcast storm**: **qualunque** tipo di ethernet frame **loopa** nella LAN per un **tempo indefinito**. Il broadcast storm può saturare tutti i collegamenti con copie di un solo frame, impattando significativamente la end-user performance.

**MAC Table Instability:**

 **Esempio**: Bob manda un frame allo SW3, quindi la MAC Table Entry associata a Bob è Fa0/13. A causa del loop di questo frame che verrà mandato dallo SW1, SW3 e anche indietro a SW3 sulla Gi0/1, lo SW3 aggiornerà la sua MAC Table perché arriverà una copia del frame con una source MAC Address a lui conosciuta, ma diversa da quella attualmente salvata, causando un aggiornamento della entry per quel MAC.

La nuova MAC Table entry per lo SW3 è Gi0/1.

A questo punto lo SW3 non può più comunicare con Bob, e se arrivasse un frame destinato a lui lo SW3 lo manderebbe erroneamente sulla Gi0/1 creando ancora più congestione.

**Multiple Frame Trasmission:** un effetto collaterale dei looping frame nel quale multiple copie di un frame, vengono spedite all’host rischiando di causare ulteriore problemi.

**Spanning Tree Protocol: Come funziona**

STP previene il loop mettendo ogni porta degli switch in uno stato: **forwarding state o blocking state**.

Nel primo funziano normalmente, mentre nel secondo i frame non vengono processati.

Per risolvere il problema del loop nell’immagine prima, STP avrebbe bloccato una porta. Se con quella porta bloccata non si fosse trovato il giusto destinatario usando il flooding ci sarebbe stato un **STP convergens,** un processo tramite il quale gli switch colletivamente decidono di cambiare lo stato delle porte per ritentare il flooding.

L’algoritmo STP crea uno **spanning tree** di interfacce che inviano frame. La struttura dell’albero di interfacce che inviano frame crea un singolo percorso da e verso ogni collegamento ethernet.

Il processo usato dall’STP, chiamato **spanning tree algorithm** (STA), sceglie quali interfacce vengono messe in stato forwarding, tutte le altre ovviamente vengono messe in blocking.

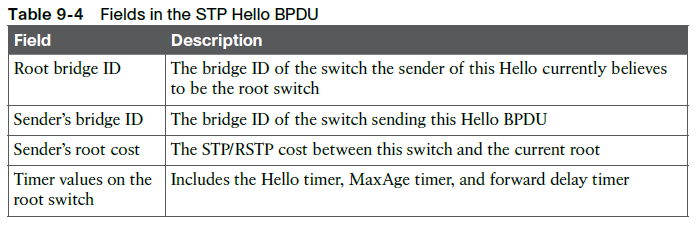
STP usa **3 criteri** per mettere l’interfaccia in forwarding:

1. STP elegge un root switch. STP mette tutte le interfacce attive in forwarding perché è lo switch principale.
2. Qualunque altro switch considera quali delle sue porte ha il costo amministrativo minore, nel percorso fra se stesso e il capo switch. Questo è il **root cost**. Questa porta viene definita **root port**, e viene messa in **forwarding**.
3. Molti switch possono essere collegati allo stesso ethernet segment, ma basandoci sul fatto che il link connette due device, un link avrà al max due device. Con due switch connessi fra loro, quello con il root cost minore è messo in forwarding state. Quello switch è il “**designated switch**”, e la porta dello switch connessa al segmento, è la **designated port** (DP).

**STP bridge ID & Hello BPDU**

Lo STA inizia con l’elezione di uno switch come root switch. Per decidere quale degli switch sarà eletto come root tutti gli switch in collegamento fra loro si scambiano degli **STP messages** contenenti il **bridge ID**.

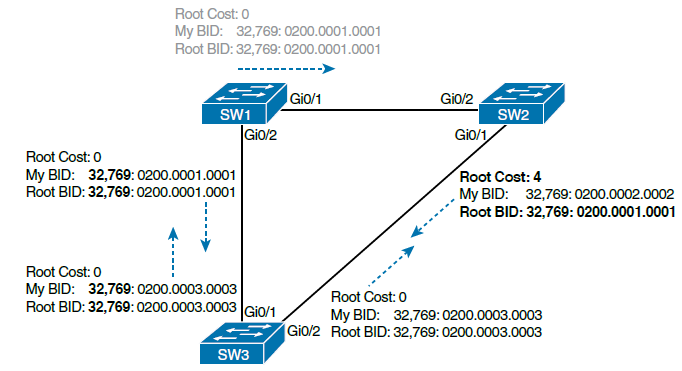
**STP/RSTP bridge ID** è un valore unico per ogni switch lungo da 8 byte. E’ formato da 2 byte del **priority field** e 6 byte del **system ID**, quest’ultimo basato sul MAC Address di ogni switch, il che rende il BID unico per ognuno.

STP/RSTP usa mex chiamati **Bridge Protocol Data Unites** (BPDU), anche chiamati **configuration BPDU**, i quali vengono usati dagli switch per scambiarsi info fra loro. Il BPDU più comune è chiamato **Hello BPDU**, che include molti dettagli, tra cui il source BID. Analizzando i BID gli switch possono capire quale fra loro ha mandato quale Hello BPDU.

**Eleggere il root Switch**

Per eleggerlo gli switch si basano sul BID all’interno delle BPDU. Il root switch sarà quello con il BID minore.

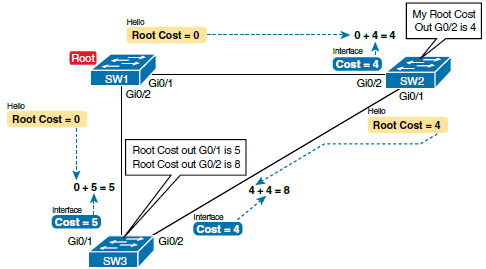
Dato che il BID si divide in due parti, si inizia guardando il **priority field:** essenzialmente quello con la priorità più bassa diventa root. Per esempio se uno switch ha di priorità 4000 e un altro 8000, il primo diventa root. Se avviene un **pareggio,** si usa il **system ID**, allora si guarda lo switch con il MAC più basso. Per esempio se uno switch ha come MAC 020000000000 e un altro ha 081111111111, il primo diventerà root.

**Il processo di elezione** inizia con tutti gli switch che chiedono di diventare root, mandando Hello BPDU con all’interno la propria BID come root BID. Se uno switch riceve un BPDU che ha un BID migliore del suo, cioè minore, quello switch smette di candidarsi e inizia ad inviare il BPDU dell’altro.

In breve funziona come le elezioni politiche, il candidato meno popolare si ritira e supporta il candidato più popolare. Quando tutti gli switch saranno d’accordo su chi ha il BID minore si avrà un root switch.

Nella figura (>) vince SW1, perché ha il BID minore.

**Scelta della root port per ogni switch**

Dopo che è stato eletto il root switch, gli switch devono scegliere una propria **root port**, ovvero la porta che ha il minor costo per raggiungere il root switch.

Nella figura (>) SW3 ha due possibili percorsi per raggiungere il root switch. Il **costo totale è la somma dei costi** di tutte le switch port che il frame attraverserà per **raggiungere** il root switch.

Come si nota il root cost del percorso diretto tra SW3 e SW1 è 5, mentre l’altro percorso è 8 (4+4). SW3 prenderà quindi la propria Gi0/1 come root port, perché ha il costo minore.

Gli switch capiscono quale porta è meglio scegliere aggiungendo il proprio **STP port cost,** (che è un valore intero assegnato ad ogni porta con lo scopo di fornire un unità di misura che permetta all’STP di scegliere quale interfaccia aggiungere alla topologia STP/RSTP) ai root cost elencati all’interno di ogni Hello BPDU ricevuti.

In caso di **pareggio** del **root cost** per due o più percorsi, lo switch applica questi tre spareggi:

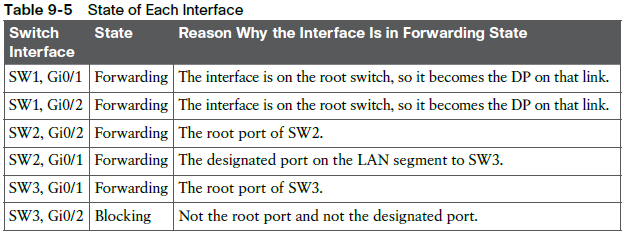
1. Si guarda al vicino con il minor Bridge ID
2. Si guarda al vicino con la minor priority port
3. Si guarda al vicino con il minor internal port number

In pratica ogni switch port ha un costo, se lo dicono, scelgono quello minore, e sbam.

**Scelta della Designated Port per ogni Segmento LAN**

L’ultimo step dell’STP/RSTP nella scelta della topologia STP/RSTP è scegliere la **Designated Port** (DP). La DP di ogni segmento LAN è la **porta** di uno switch che ha il **costo minore della LAN**. In pratica, lo switch con il **minor costo** per raggiungere il **root switch**, diventa la **DP del segmento**.

Nella figura precedente, ad esempio, il root cost di SW2 è 4 ed SW3 è 5, in questo caso vince SW2 facendo diventare la sua Gi0/1 la DP.

Tutte le **DP** vengono sempre messe in **forwarding state**. Se i costi dovessero pareggiare, si passa al BID più basso. SW2 nell’esempio precedente avrebbe ugualmente vinto, avendo il BID più basso.

L’unica porta che non ha bisogno di essere in forwading è la Gi0/2 di SW3. L’immagine (>) riassume la situazione.

**STP, come funziona quando il network è stabile**

Un STP root switch manda un Hello BPDU ogni 2 secondi di default. Ogni nonroot manda l’Hello a tutte le DP, ma solo dopo aver cambiato alcuni item dell’Hello.

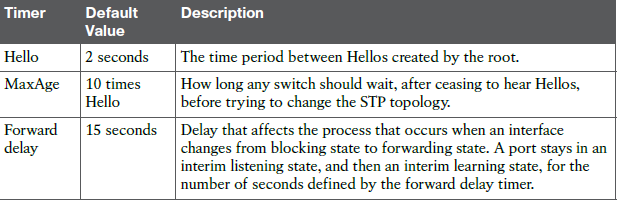
Per esempio, un Hello con un timer di 2 secondi sul root switch, gli altri switch lo cambieranno e rimanderanno dalle DP, così tutti gli switch continueranno a riceverlo ogni 2 secondi.

Il tutto si riassume nei seguenti punti:

1. Il root crea e manda un Hello BPDU, con root cost = 0, alle interfacce in forwarding.
2. I nonroot ricevono l’Hello sulle loro root ports, dopo aver cambiato mettendoci il loro BID come BID sorgente e listato il loro root cost, lo switch manda l’Hello alle DP.
3. Step 1 e 2 si ripetono fino a quando qualcosa cambia.

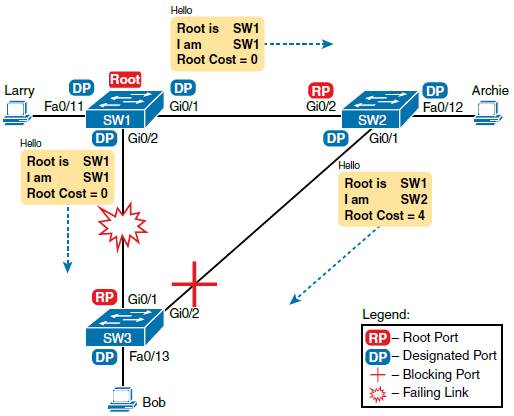
**Se non riceve la BPDU**?

Quando uno switch **non riceve la BPDU**, sa che c’è un **problema** nel network. Tutti gli switch confidano di **ricevere** periodicamente l’Hello come **strategia** per essere sicuri che il **path al root switch sia funzionante**. Dunque se vede che non arriva, oppure arriva con dettagli diversi, si attiva il **processo di cambio del spanning-tree topology.**

**STP Timers che gestiscono STP Convergence**

Per varie ragioni, il processo di STP convergence necessita di 3 timers, come si vede in tabella.

Se uno switch non riceve l’Hello BPDU nel “Hello time” , lo switch continua normalmente. Ma, se l’Hello non arriva neanche dopo il MaxAge, lo switch inizia a cambiare la STP topology. In questo caso lo switch aspetterebbe 20 secondi prima di reagire.

Dopo che il MaxAge è finito, lo switch rifà le sue “STP choices”, basandosi sulle Hello BPDU ricevute dagli altri switch. Rivaluta quali degli switch dovrebbe diventare il root.

Nella figura (>) si riassume il tutto:

**Scelta del nuovo switch root**:

SW3 non riceve più l’Hello sulla sua RP Gi0/1, quindi reagisce.

SW2 non ha bisogno di reagire perché continua a ricevere l’Hello periodica sulla sua Gi0/2.

SW3 capisce che non riceverà più Hello sulla sua Gi0/1.

Ora SW3 può iniziare a reagire, e rivalutare chi per lui è il root. SW3 continua a ricevere l’Hello da SW2, a sua volta ricevuti da SW1.

SW1 continua ad avere il BID minore, quindi rivince le elezioni.

**Scelta della nuova root port**:

Adesso SW3 rivaluta la scelta della sua RP. SW3 riceve l’Hello solo su Gi0/2 anche se blocking, ma riceve. A prescindere dal root cost, Gi0/2 diventa la RP.

**Scelta della nuova DP**

In questo esempio niente da fare, DP rimane Fa0/13 perché nessun altro switch è connesso a quella porta.

**Cambiare Stati delle Interfacce con STP (Roles & States)**

STP usa l’idea di ruoli e stati (**roles and states**).

**Roles**, come **root port o designated port**, si usano per analizzare la topologia della LAN.

**States**, come **forwarding o blocking**, dice ad uno switch se spedire o ricevere frame.

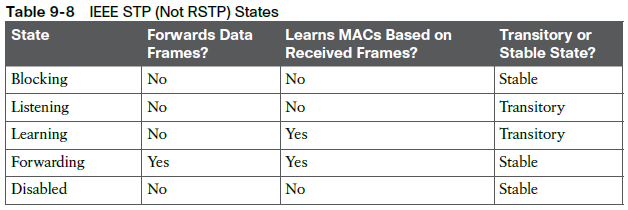
Quando STP converge, uno switch sceglie nuovi ruoli per le porte, che determinano nuovi stati.

Una **porta in forwarding** può **subito** essere messa in **blocking**, mentre **viceversa** no, lo switch mette la porta prima in due **intermediate states:**

* **Listening**: tipo come il blocking, la porta non può mandare frames. Lo switch rimuove le entries della MAC Table non più usate, i quali potrebbero essere il motivo dei loop.
* **Learning**: le interfacce in questo stato continuano a non mandare frame, ma lo switch inizia ad imparare i MAC Address dei frame ricevuti su quell’interfaccia.

Uno switch mette la porta da **blocking > listening > learning > forwarding**.

Per passare da uno stato all’altro (listening e learning) ci vogliono 15 secondi, in totale 30. Se uno switch deve aspettare anche il MaxAge (di dafault 20 sec), in totale verranno 50 secondi di ritardo.

La tabella (>) riassume i vari states delle interfacce.

**STP comparata a Rapid STP (anche detta IEEE 802.1w)**

**STP e RSTP** lavorano in modo **uguale** sotto parecchi aspetti:

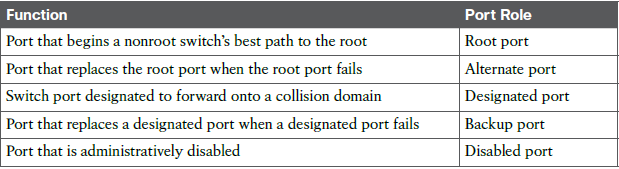
* Eleggono un root switch usando le stesse regole e tiebreakers.
* Scelgono le root port con le stesse regole.
* Eleggono DP su tutti i LAN segment con le stesse regole e tiebreakers.
* Mettono le porte in **forwarding o blocking**, RSTP chiama quest’ultimo ***discarding state***.

Con tutte queste somiglianze, perché la IEEE ha dovuto inventare la versione Rapid? La differenza sta nella convergenza, mentre STP ci mette 50 secondi, RSTP ce ne mette solo 10.

**RSTP funziona cosi**:

* Aggiunge un meccanismo dove lo switch può rimpiazzare la root port, senza dover aspettare per metterla in forwarding state.
* Aggiunge un meccanismo che rimpiazza la DP, senza dover aspettare per raggiungere il forwarding.
* Diminuisce il timer di attesa per alcune casistiche.

Per esempio, quando nell’STP un link funziona ma lo switch, per qualche motivo, non riceve più l’Hello che riceveva prima, con STP dovrebbe aspettare 10 volte l’Hello timer di 2 sec (quindi 20 secondi di default), con **l’RSTP aspetta solo 3 volte l’Hello timer**, in più mentre aspetta può mandare mex in cui chiede ai vicini cos’è successo.

RSTP usa il termine ***alternate port*** per indicare le porte che possono essere usate come root port, quando quest’ultima non funziona.

***Backup port*** è una porta di backup per la DP.

Un’altra differenza significativa è che l’STP fa creare al root l’Hello che poi viene spedita tra gli switch, con **l’RSTP ogni switch crea indipendentemente la propria Hello** BPDU.

**RSTP e il ruolo della Alternate Ports**

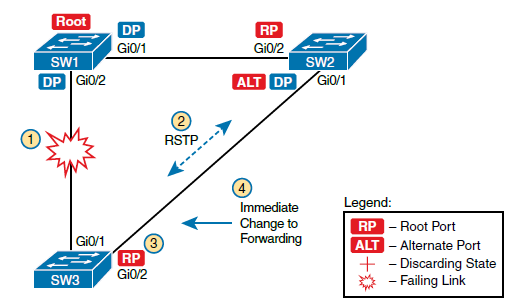
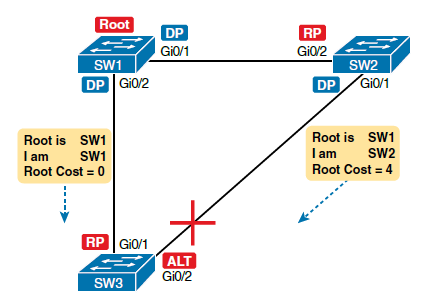
Con STP ogni nonroot sceglie una porta come RP. RSTP segue la stessa logica con le stesse regole, ma in più permette di scegliere altre RP, chiamate **alternate ports**.

Una alternate port è la seconda miglior scelta alla root port. Quando la root port non riceve, lo switch attua cosi:

1. root port viene passata in disabled port (b) forwarding in discarding (equivalente di blocking in STP)

Senza aspettare timers o altro, cambia i ruoli e gli stati per l’alternate port: ora è root port ed in forwarding.

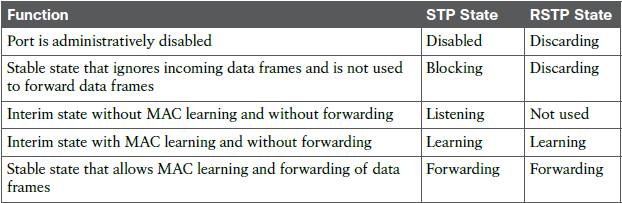
La nuova root port non ha bisogno di aspettare in altri stati, tipo learing, passa diretto in forwarding.

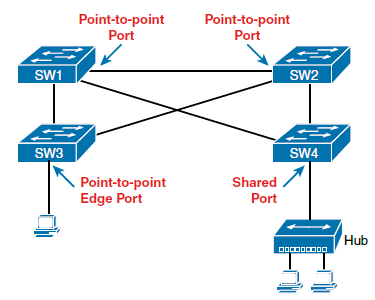


**‑‑‑‑‑‑‑>**

1. Il link fra SW1 e SW3 non va, la root port di SW3 fallisce.
2. SW3 e SW2 scambiano messaggi RSTP per conferma che la alternative port di SW3 diventerà la root.
3. SW3 Gi0/1 viene disabilitata e la Gi0/2 immediatamente messa root, senza attesa.

**Stati e Processi di RSTP e STP**



**Tipi di Porte RSTP**

Porte che connettono due switch fra loro e non sono al “bordo” del network si chiamano **point-to-point ports**, mentre porte che connettono ad un endpoint device, come un PC o server, sono chiamate **point-to-point edge ports, o solo edge ports**. Infine le **shared ports** sono porte che connettono un hub, come nell’immagine.

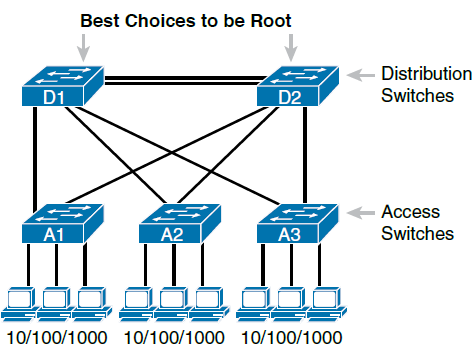
**Features Opzionali STP: EtherChannel, PortFast e BPDU Guard**

**EtherChannel**: combina multipli segmenti paralleli di uguale velocità (fino a 8 segmenti) tra gli stessi pair di switches, come nell’immagine.

**PortFast:** permette allo switch di passare da blocking a forwarding, bypassando listening e learning.

**BPDU Guard**: serve per proteggere le porte da attacchi del tipo un utente che entrando dalle interfacce, passa dagli switch, diventa root e può creare problemi. Il Guard interviene bloccando le porte dove riceve BPDU, questa feature è molto utile nelle interfacce che dovrebbero essere usate solo in access

**CAP. 10 – Capire RSTP attraverso la configurazione**

Negli switch Cisco odierni si usa RSTP, arrivano **già configurati** di default senza bisogno di ulteriore configurazione. Anche se alcuni switch usano RSTP e altri STP, possono **intercomunicare** per creare uno **spanning tree funzionante**.

Nella figura (>) è mostrato il classico modello di una LAN.

I **distribution** vengono collegati ad altri switch per favorire la “distribuzione” dei frame nella lan, vengono settati come root.

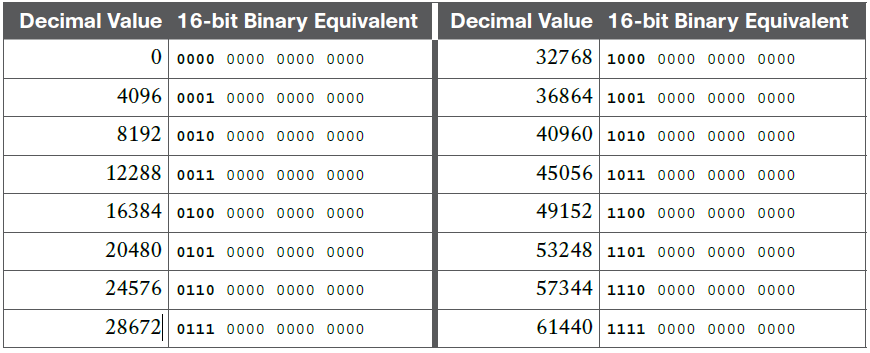
Gli **access** invece connettono agli endpoint device.

**Bridge ID e System ID extension**

Originariamente il **BID** era formato da **2 byte di priority field e 6 byte per MAC Address**, ora invece per supportare l’utilizzo delle **VLANs** all’interno dell’MSTP (Multiple STP), il **priority field** viene separato in due campi, **uno di 4 bit considerato il nuovo priority field, seguito da 12 bit chiamato system id extension**, che rappresenta il VLAN ID.

I Cisco switch permettono di configurare il BID ma solo i 4 bit, con il comando:

**spanning-tree vlan “*vlan-id*” priority “*x*” Tabella valori configurabili come priority**

che richiede un numero decimale tra 0 e 65535, ma non tutti i numeri in questo range funzionano, devono per forza essere **multipli di 4096**.

Questo MSTP consente di configurare due switch come più papabili per diventare root switch, con i comandi:

**spanning-tree vlan “x” root primary**

**spanning-tree vlan “x” root secondary**

Questi due comandi impongono allo switch di prendere una decisione basandosi sul priority value, ma immagazzinando il valore scelto come priority value del comando **spanning-tree vlan “x” priority “*value***”.

Usando il primo comando, lo switch diventerà root e autoimposterà la sua priority a 24576 o 4096.

Nel caso si usi il terzo comando per settare la priority, se quest’ultima è minore di quella del root switch, quello switch diventerà root.

**Differenze fra RSTP e RPVST+**

RSTP e il RPVST+ usano i meccanismi del RSTP, ma RPVST+ usa questi meccanismi per ogni singola VLAN, cosa che non fa RSTP.

I due metodi differiscono come segue:

1. RSTP crea un solo tree (CST), mentre RPVST+ crea un tree per ogni VLAN.
2. RSTP manda un set di mex (BPDU) per tutto il network, senza tener conto del numero di VLAN. Mentre RPVST+ manda un set di mex per ogni VLAN.
3. RSTP e RPVST+ usano due differenti destination MAC Address: RSTP usa il multicast MAC Address 0180.C200.0000 (IEEE standard), mentre RPVST+ usa il multicast MAC Address 0100.0CCC.CCCD (Cisco).
4. Quando si trasmettono mex su un VLAN trunk, RSTP manda i mex alla Native VLAN senza header. RPVST+ invece manda ogni VLAN message alla propria VLAN con un header.
5. RPVST+ aggiunge un extra type-length value (TLV) al BPDU che identifica VLAN ID, mentre RSTP non lo fa perché ignora le VLAN.
6. Entrambi hanno un priority field di 16 bit, di cui 12 dedicati al system ID extension. RSTP setta questo valore a 0000.0000.0000 perché ignora le VLAN, mentre RPVST+ usa VLAN ID.

**Port Cost**: con il comando **spanning-tree [*vlan x*] cost *y*** permette di settare il cost per le interfacce.

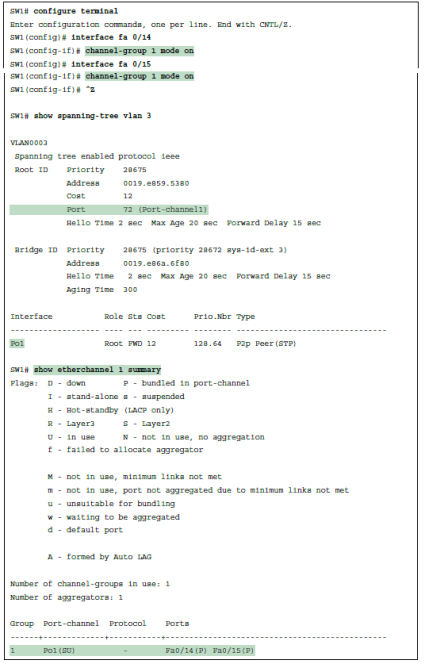
Puo essere usato sia per tutte le VLAN che per una specifica VLAN di quella porta. Cambiare i costi, cambia anche il root cost degli switch, il quale impatta sulla scelta della RP e DP.

**EtherChannel**

Come già visto gli switch trattano i multipli collegamenti paralleli fra loro, come uno solo, chiamato EtherChannel. Quest’ultimo serve per bilanciare il traffico in tutti i link.

**Configurare un EtherChannel**

Per configurarlo tutte le porte desiderate devono far parte dello stesso channel, usando su ogni interfaccia il comando **channel-group** con la parola chiave **mode** **on** e tutte con lo stesso numero.

Per displayare lo status si sa **show etherchannel**.

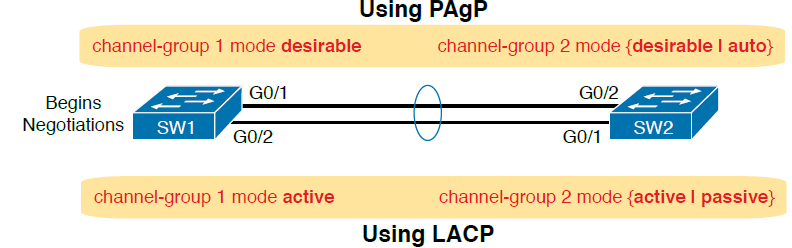
Per configurarlo si seguono questi due step:

1. Usare il comando **channel-group “*number*” mode on** nella interface config mode, per ogni porta che deve far parte del channel.
2. Usare lo stesso numero per tutti i comandi sullo stesso switch, ma il channel group degli switch vicini può essere diverso.

**Configurare dinamicamente EtherChannel**

Oltre alla configurazione manuale, i Cisco switch supportano una configurazione che usa protocolli dinamici per negoziare quando un link deve diventare parte di un EtherChannel o no. Questi protocolli inviano mex verso gli switch vicini scoprendo quando un link passa i controlli, se li passa il link viene aggiunto all’EtherChannel, altrimenti messo in down state e non usato (se non passa i controlli).

Gli switch supportano due protocolli: **Cisco Port Aggregation Protocol** (PAgP) e **Link Aggregation Control Procol** (LACP) della IEEE, entrambi svolgono lo stesso compito con una piccola differenza.

Il **LACP** supporta **più link in un channel**, ovvero **16**, il **PAgP massimo 8**. Inoltre il LACP può tenere attivi solo 8 link contemporaneamente, gli altri 8 rimangono in attesa di essere usati nel caso uno degli attivi fallisca.

Per configurare i protocolli si usa il comando **channel-group** con le parole chiavi “**desirable” e “auto**” per abilitare il PAgP e “**active” e “passive**” per abilitare il LACP. Usando desirable e active lo switch inizierà la negoziazione.

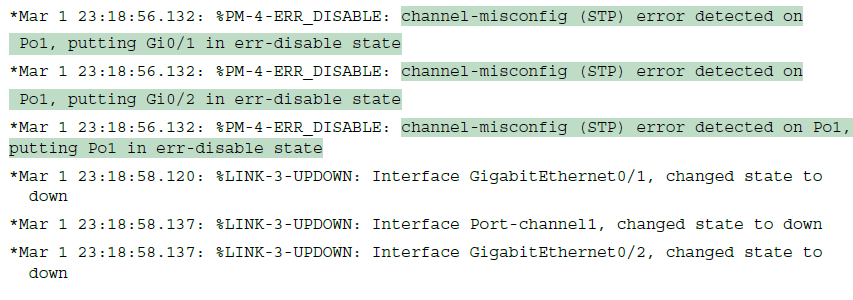
**Configurazione porte degli EtherChannel**

Quando si usa il comando channel-group tutte le altre impostazioni devono essere settate correttamente, altrimenti lo switch potrebbe decidere lo stesso di non usare l’interfaccia all’interno dell’EtherChannel. Prima di usare un’interfaccia nell’EtherChannel lo switch compara le configurazioni con quelle già esistenti all’interno del channel, questa nuova interfaccia deve essere settata nello stesso modo delle interfacce già esistenti, altrimenti lo switch configurerà l’interfaccia come parte del channel ma non la userà e la mette in non-working.

Lista degli items che lo switch checka sulle interfacce:

* Speed
* Duplex
* Operational asses o trunking state (o tutte in access o tutte in trunk)
* Se la porta è in access, l’access VLAN
* Se la porta è trunk, la lista delle VLAN abilitate e la Native VLAN
* Impostazioni STP

In più gli switch checkano le impostazioni degli switch vicini, per farlo usano il LACP o il PAgP, oppure il CDP se la configurazione è manuale.

Nell’esempio (>) SW1 e SW2 cercano di usare due link in un EtherChannel come nella figura precedente. Prima di configurare l’EtherChannel alla porta G0/2 SW1 è stato dato un diverso port cost rispetto alla G0/1. Nell’immagine è il risultato del giusto comando, ma avendo port cost diversi le porte vengono messe in disable state.

Perché? SW1 rileva un differente port cost tra le sue porte, inoltre mette il port channel in error disable state. Come risultato il port channel non funziona e quindi le interfacce non sono operative.

Per risolvere questo problema bisogna riconfigurare i settings in modo che le porte abbiamo lo stesso cost, inoltre le port channel e interfacce devono essere **shutdown** e poi **no shutdown**. (spente e riaccese)

**NB**: quando si da il shutdown al port channel anche le interfacce vengono shutdownate, idem per no shutdown.

**Opzioni Configurazione EtherChannel Load Distribution**

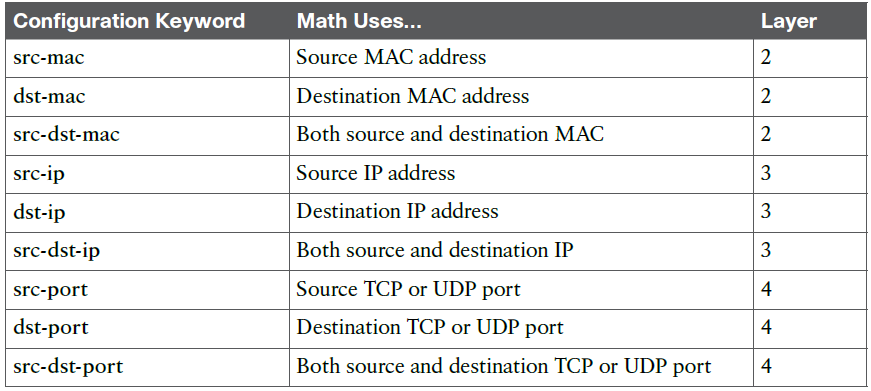
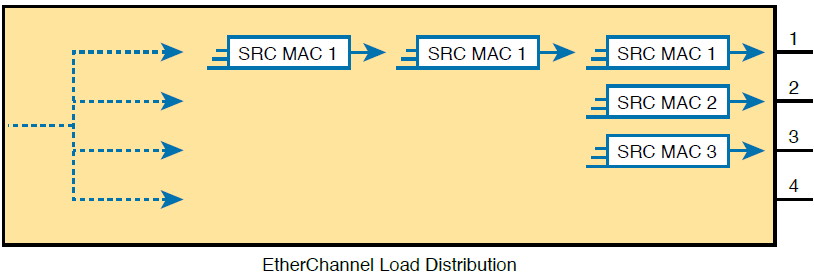
La Load Distribution basa le sue scelte sui valori numerici trovati negli header di livello 2, 3, 4 per ogni frame. Il processo può essere configurato attraverso il comando **port-channel load-balance** “*metod”,* che permette la scelta del metodo da usare; a seconda del metodo scelto l’algoritmo farà il suo match su campi diversi (alcuni switch supportano metodi basati su MAC o MAC e IP).

Tabella dei metodi >>

I vari algoritmi di distribuzione hanno alcuni obiettivi in comune:

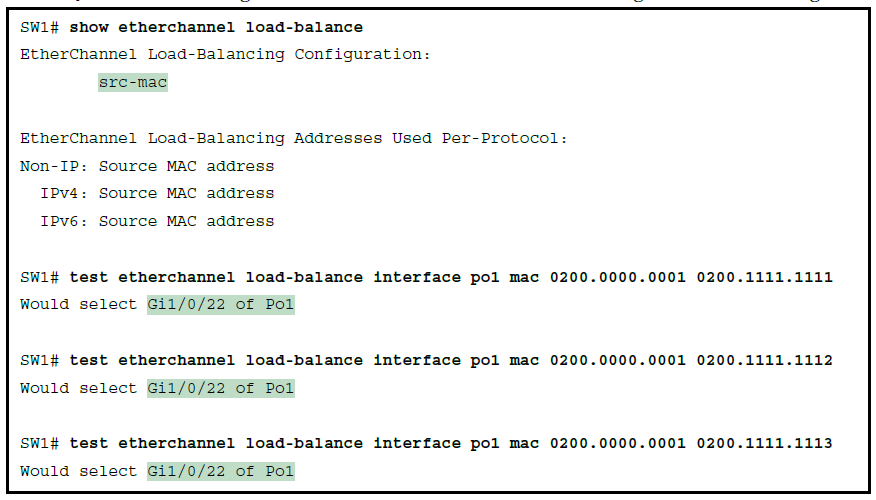
1. Fare in modo che tutti i messaggi che fanno parte di un flusso della stessa applicazione, vengano inviati attraverso lo stesso link.
2. Integrare l’algoritmo di load distribution all’interno di hw ASIC.
3. Usare tutti i link attivi dell’EtherChannel.
4. Tenendo presenti i vincoli degli altri obiettivi, bilanciare il traffico dei link attivi.

Per esempio con il metodo di load distribution **src-mac** tutti i frame con lo stesso MAC Address scorrerando nello stesso link, come nell’immagine.

Cisco fornisce diversi metodi così che ci si possa fare un idea su quali dei campi esaminati possiedono la più grande varietà di valori. Maggiore è la varietà, migliore è l’effetto di bilanciamento del traffico, e minore è la chance di mandare quantità di traffico sproporzionate nei link.

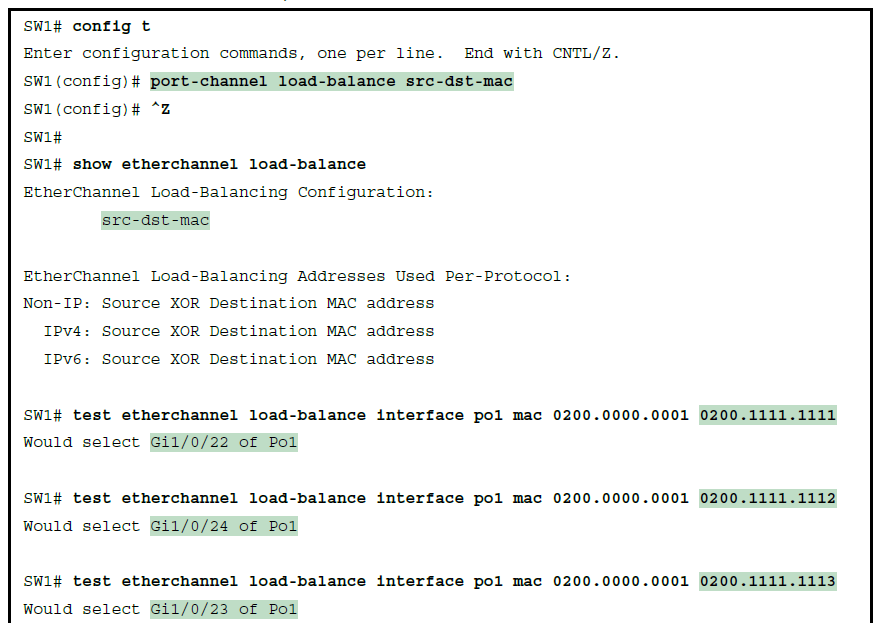
**Gli effetti dell’algoritmo di Load Distribution**

Con il comando **show etherchannel load-balance** viene mostrato il metodo che si sta usando.

Con il comando **test etherchannel load-balance interface “*id***” “valore mac/valore ip in base al metodo”, chiede allo switch prendere in considerazione alcuni address o porte e di rispondere alla domanda “su quale link spedisci?”.

Nell’immagine, tutti 3 i test mostrano la stessa interfaccia di uscita (Gi1/0/22), poiché si sta usando il metodo src-mac, che quindi non considera i 3 diversi destination MAC address inseriti.

Invece in questo caso vediamo che le interfacce di uscite sono diverse perché sono diversi anche i source MAC address.

In quest’ultimo caso invece è stato cambiato il metodo in **src-dst-mac** e di conseguenza prendendo in considerazione anche il destination MAC address le porte di uscita sono diverse.

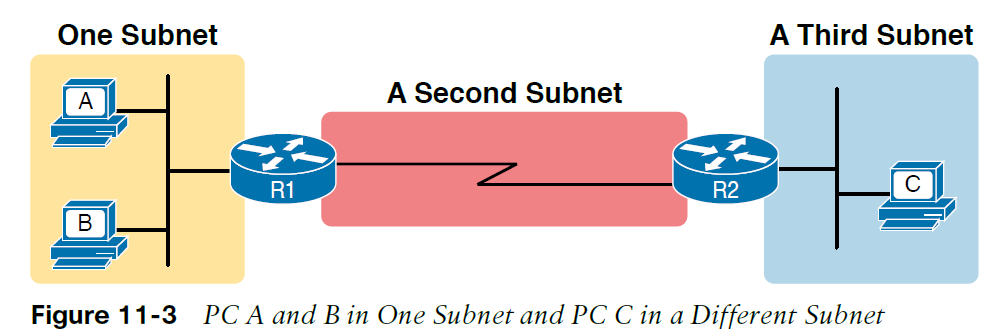
**CAP. 11 – SUBNETTING**



Esistono 4 domande che posso riassumere come analizzare il subnetting e l’addressing per ogni network:

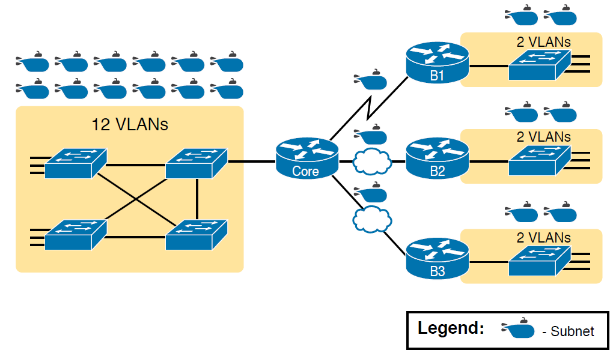
1. Quali host devono essere raggruppati in una certa subnet?
2. Quante subnet richiede il network?
3. Quanti host IP richiede ogni subnet?
4. Useremo una one size subnet per semplicità, o no?
5. **Regole sul perché determinati host sono in una determinata subnet**

Ogni device che si connette ad un network ha bisogno di un indirizzo IP, che viene assegnato con regole standard. IP Addressing raggruppa con regole ogni blocco di indirizzi IP in gruppi, chiamati subnet.

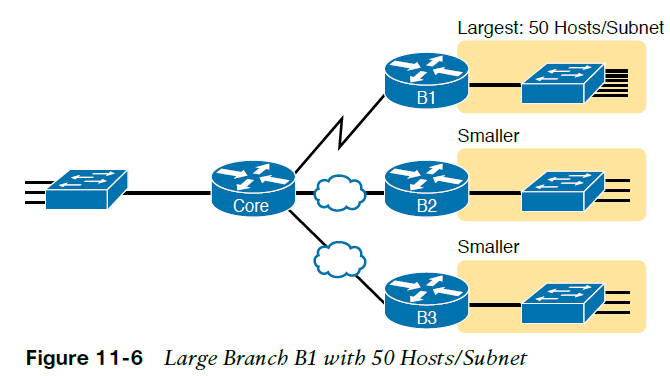
Le regole sono:

1. Gli indirizzi nella stessa subnet **NON** sono separati da un router.
2. Gli indirizzi in diverse subnet sono separati da **ALMENO UN ROUTER**., come nell’immagine. (>)

NB: siccome i router mandano frame da una subnet all’altra, faranno parte di più subnet e di conseguenza saranno configurati con più indirizzi IP, uno per interfaccia.

2. **Determinare il numero di subnet**

Per determinare il numero di subnet hai bisogno di avere informazioni dal plan, come nell’immagine.

3. **Determinare il numero di host per subnet**

E’ l’ingegnere che deve specificare quanti hosts ci vogliono, in questo caso (>) l’ha specificato solo per B1, perché fin quando B2 e B3 stanno sotto i 50 di B1 non c’è bisogno di specificare.

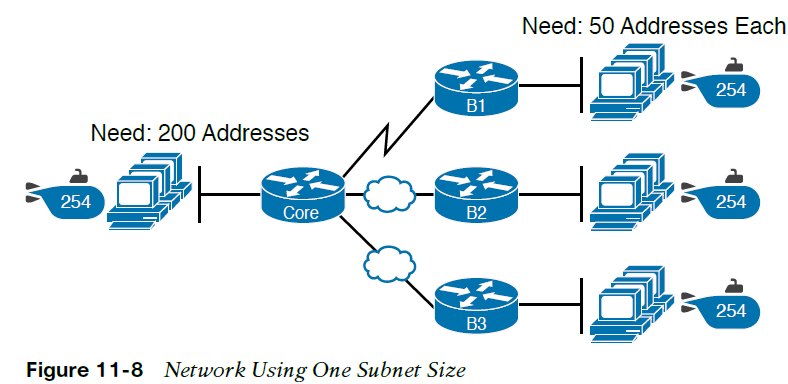
**Specificare la grandezza di una Subnet**

L’ingegnere assegna alla subnet una **subnet mask**, e quella maschera definisce la grandezza della subnet.

Gli indirizzi dunque vengono definiti con 2x indirizzi, meno due, quello numericamente più basso che è il **subnet number (numero identificativo della subnet)**, e quello numericamente più alto che è il **subnet broadcast address**, come risultato ci sono **2x - 2** indirizzi usabili per subnet.

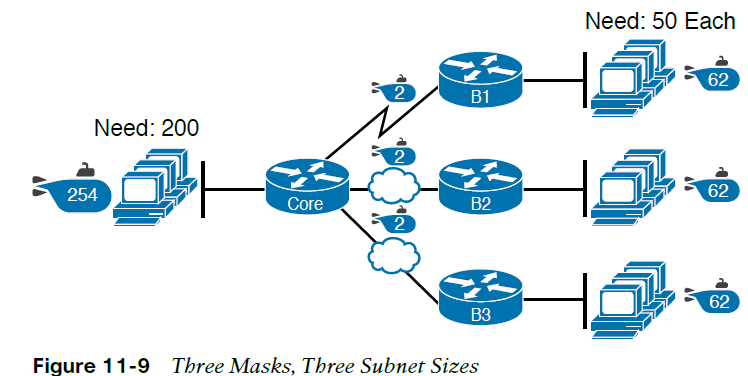
**One Size Subnet**

Per usare la single-size subnet, devi usare la stessa maschera per tutte le subnet di quel network.

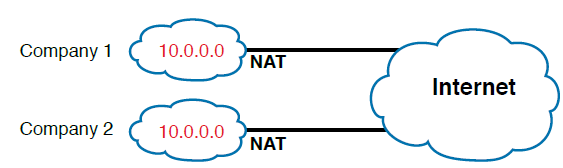
Ma quale maschera usare? Bisogna scegliere la maschera che riesce a supportare la subnet che richiede il maggior numero di indirizzi IP, come nell’immagine. (>)

La LAN a sinistra richiedono 50 address ma la main LAN ne richiede 200, per realizzare ciò 27 -2 = 126 dunque non è sufficiente, ma avendo 8 bits a disposizione, 28 – 2 = 254, risulta più che sufficiente.

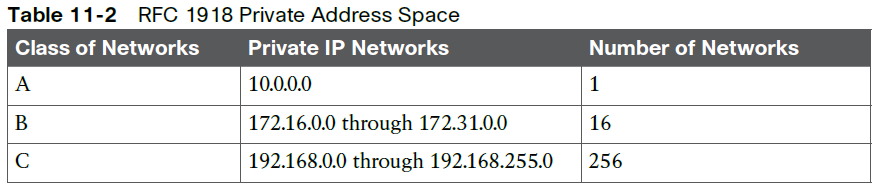
**Pregi e Difetti** – rende il lavoro più semplice allo staff, usano tutti una ed una sola maschera, però d’altra parte è un alto spreco di indirizzi IP. Nell’immagine le LAN a sx richiedono 50 indirizzi, ma ognuna ne supporta a prescindere 254.

**Multiple Size Subnet**

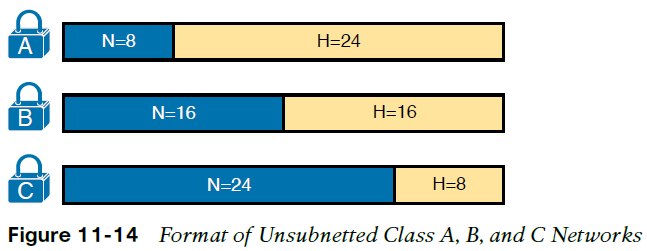
In questo caso si avranno diverse maschere per ogni subnet. Differenti maschere vuol dire differenti numeri di host. Nella figura precedente venivano sprecati indirizzi, la figura di fianco (>) mostra l’alternativa del Multiple Size, con 3 diverse maschere per ognuna delle 3 richieste.

**Indirizzi IP Privati – NAT**

Il Network Address Translation (NAT) è un metodo che permette a più aziende di usare lo stesso indirizzo ip privato, come nell’immagine.



Tutte le aziende usano il NAT ormai, la tabella definisce i range di indirizzi privati che un azienda può avere.

**Le classi di indirizzi**

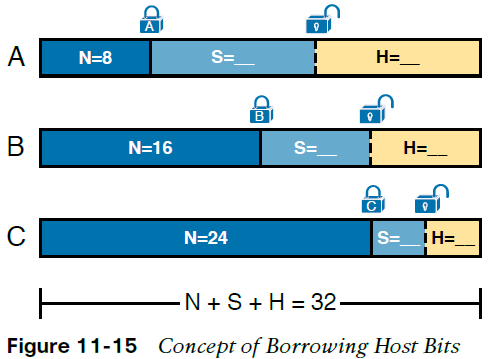
Ogni classe ha un formato di indirizzo IP totale di 32 bit.

■ **Class A:** 224 – 2 = 16,777,214

■ **Class B:** 216 – 2 = 65,534

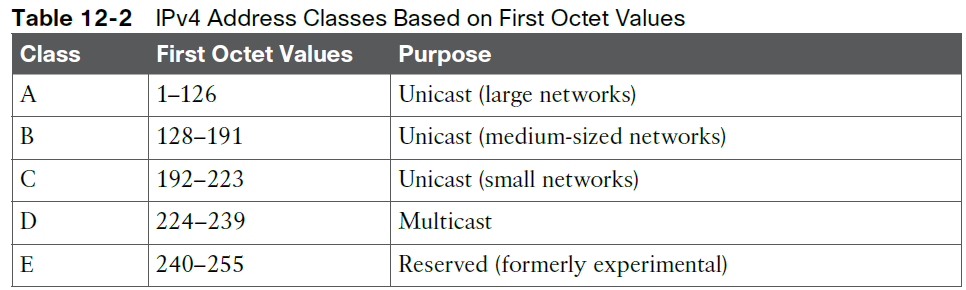
■ **Class C:** 28 – 2 = 254

N = Bit parte network. H = bit parte host.



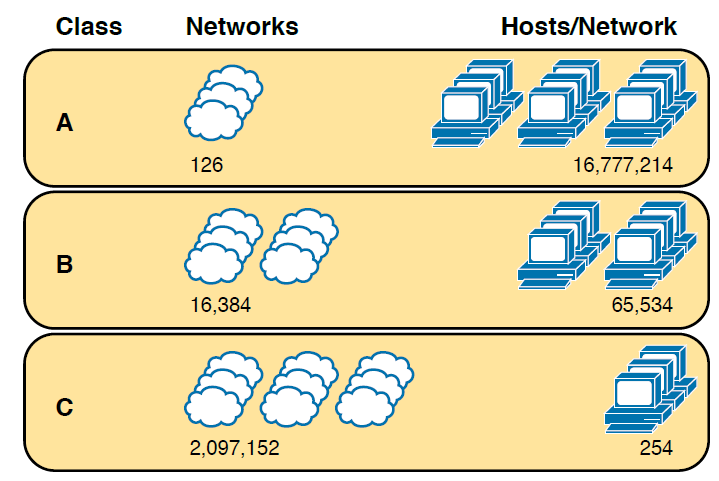
Poiché il formato degli indirizzi deve rimanere 32 bit, per identificare la subnet viene presa una parte dei bit disponibili per gli host, ed assegnata alla parte subnet, come nell’immagine. (>)

2^s deve essere maggiore del numero di subnet richieste, idem per gli host, 2^h – 2 deve essere maggiore del numero di host richiesti per ogni subnet.

**CAP. 12 – Classi di indirizzi**



NB: Gli indirizzi Classe A che iniziano con 0 o 127, sono riservati. B e C non hanno indirizzi riservati.

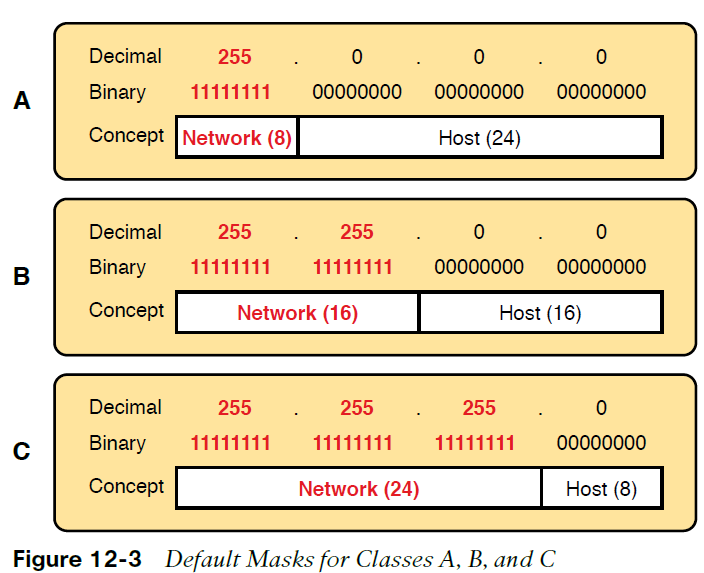


Classe A: 126 network, 16mln di host per network.

Classe B: 16384 network, 65k hosts per network.

Classe C: 2mln network, 254 host per network.

Default Mask di ogni Classe A, B, C.

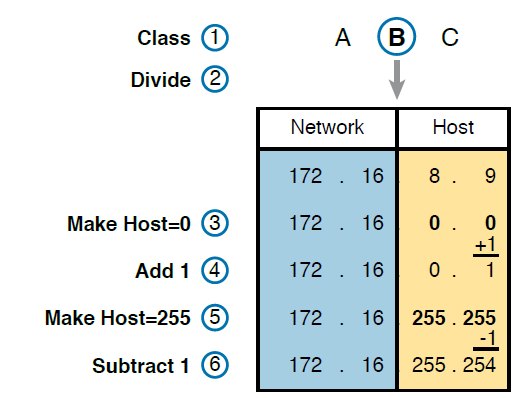


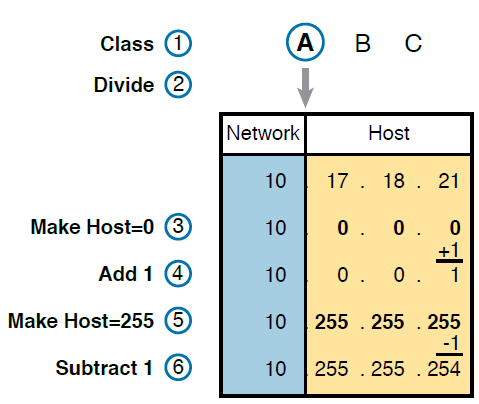
**Network ID e Broadcast Address**

Il Network ID è il numero numericamente più basso del network, cioè il primo. L’indirizzo IP usabile per primo è quello subito dopo il Network ID. Il Broadcast Address è il numericamente più alto del network, l’ultimo. L’indirizzo IP usabile per ultimo è quello prima del Broadcast Address.

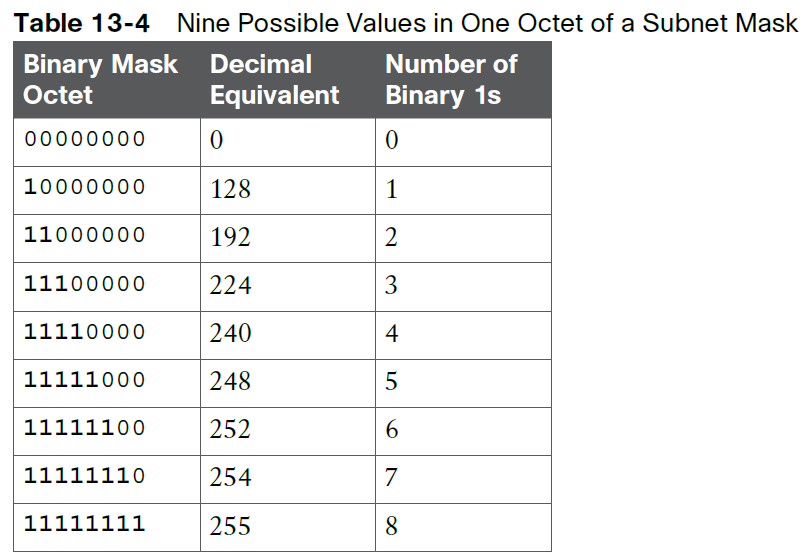
**Come trovare Broadcast, Network ID, Primo Indirizzo IP e L’ultimo Indirizzo** **IP**

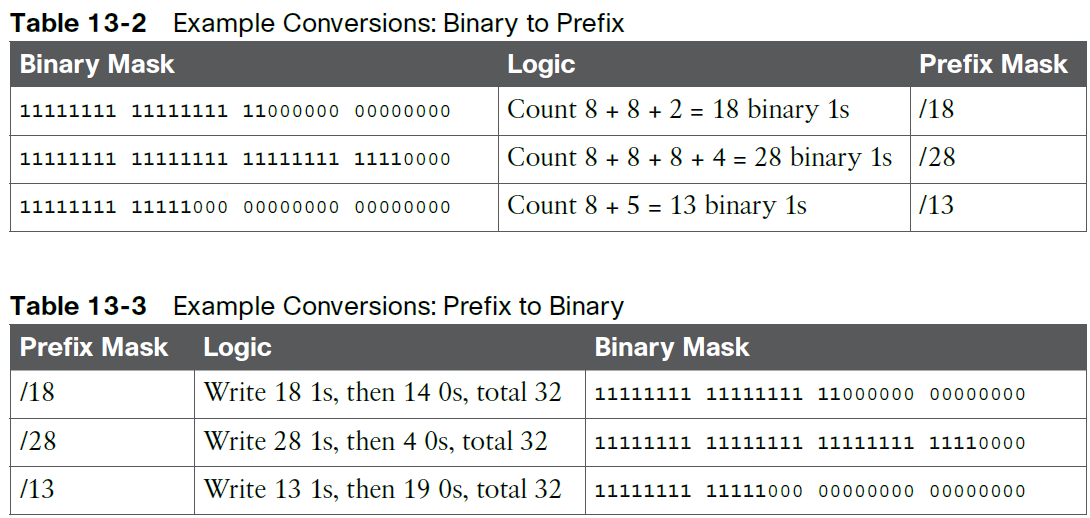
1. Determinare la classe dell’indirizzo basandosi sul 1° ottetto
2. Dividere gli ottetti della parte host da network basandosi sulla classe
3. **Network ID** > Cambiare tutti gli ottetti della parte host a zero.
4. **Primo Indirizzo >** aggiungere 1 all’ultimo ottetto del Network ID.
5. **Broadcast Address >** cambiare tutti gli ottetti della parte host e metterli a 255.
6. **Ultimo Indirizzo >** sottrarre 1 dal quarto ottetto del Broadcast Address.

 **es. 10.17.18.21 (Classe A) es. 172.16.8.9 (Classe B)**

****

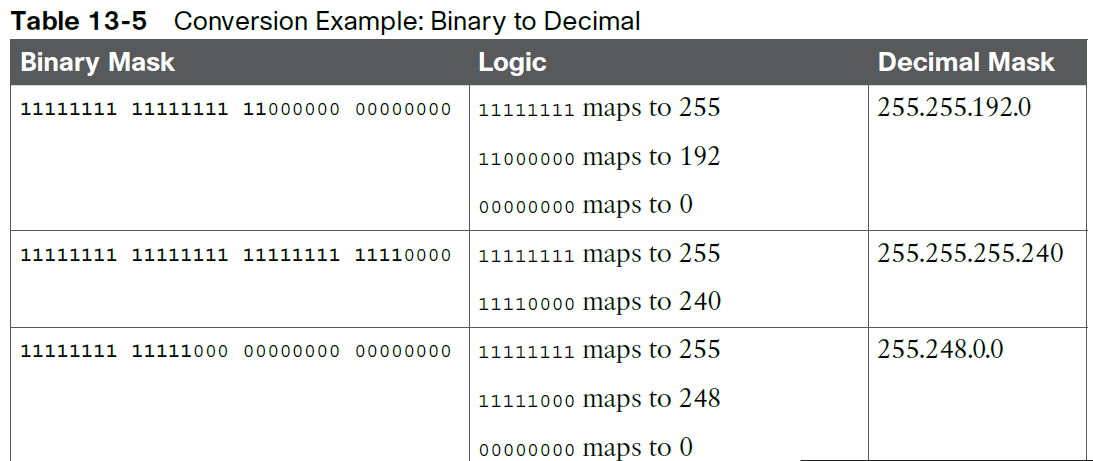
**CAP. 13 – Binary**



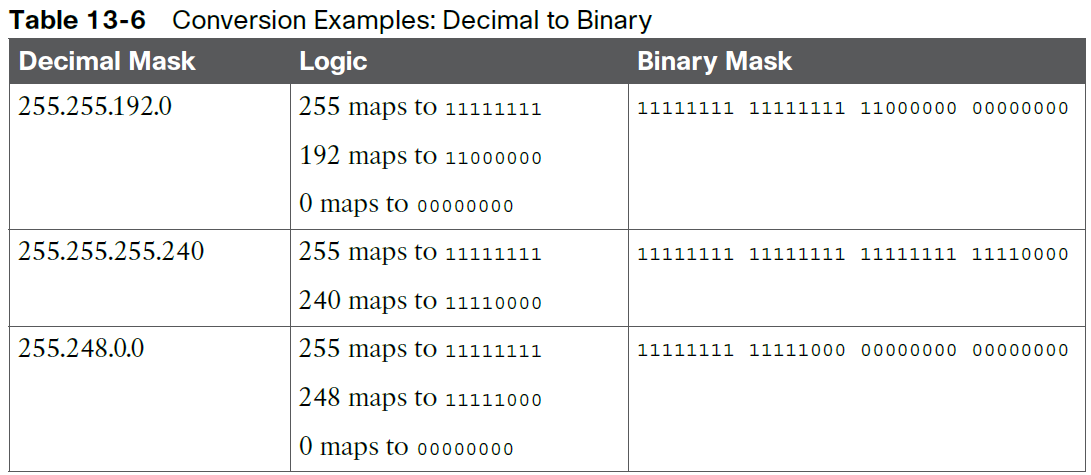


Conversione Binary > Prefix (/)

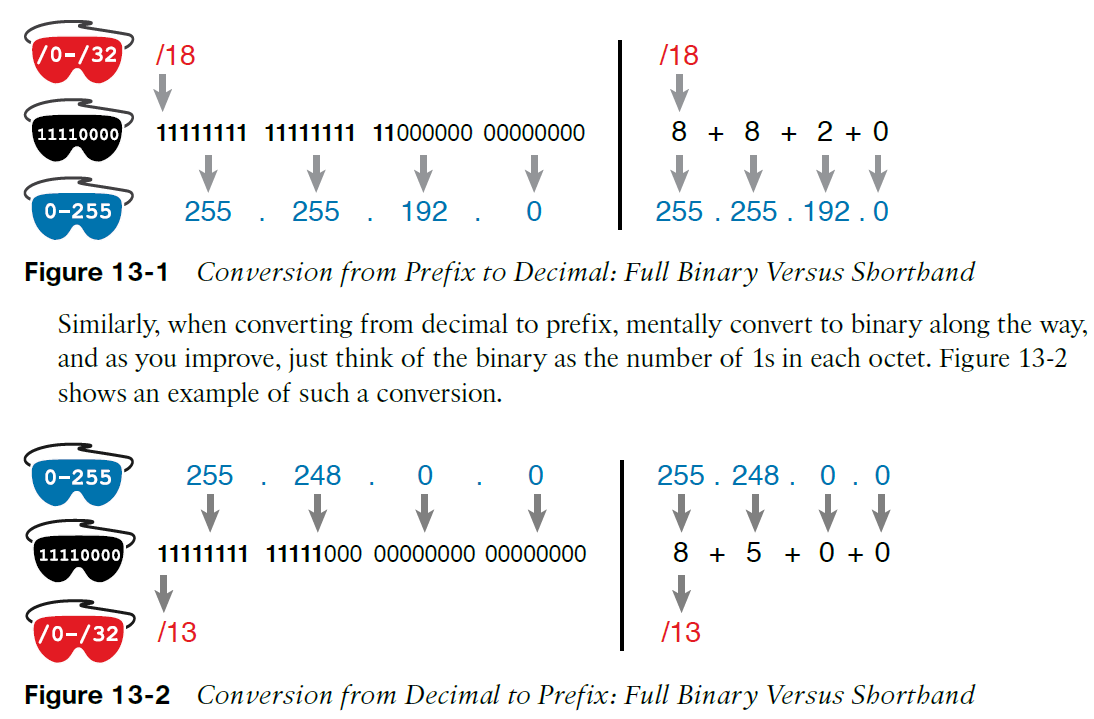
Conversione Prefix (/) > Binary



Conversione Binary > Decimal

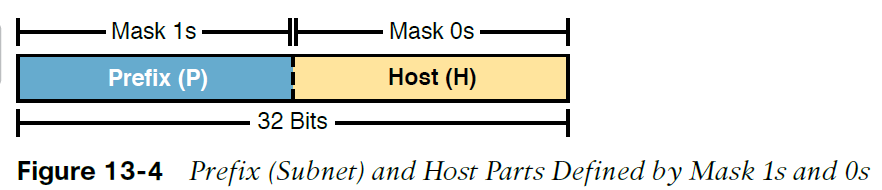


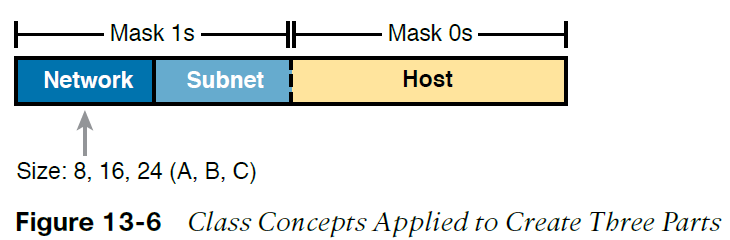
Conversione Decimal > Binary



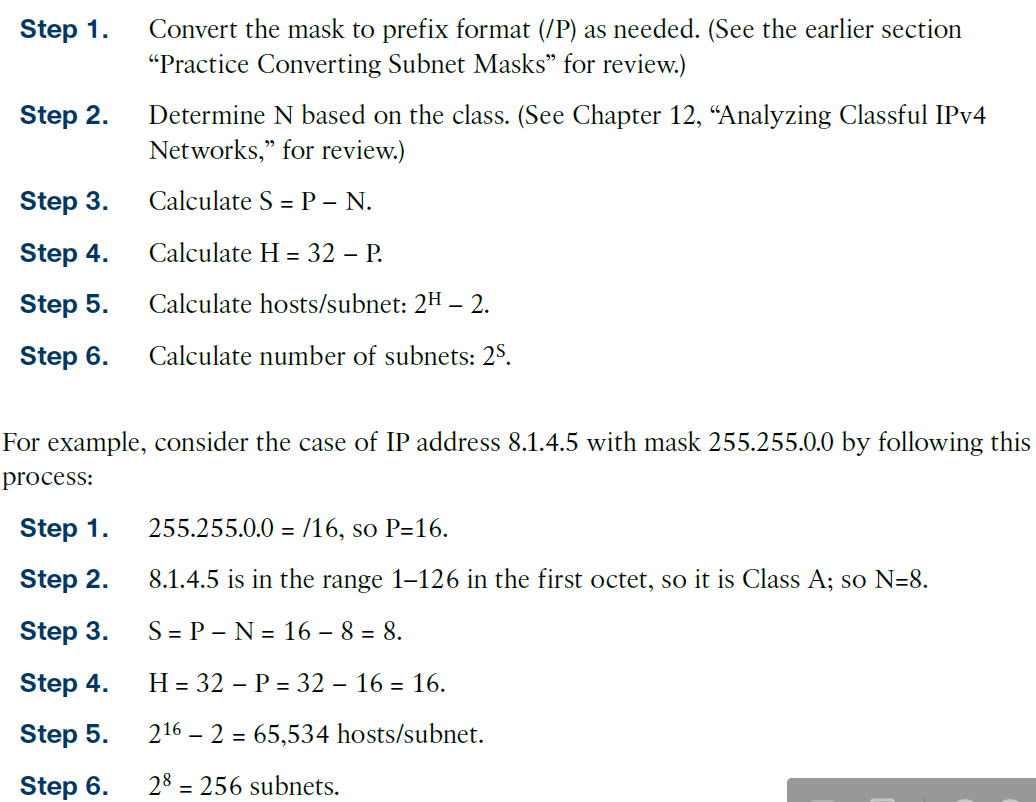
Conversione Prefix > Decimal

Conversione Decimal > Prefix

**Classless Addressing**, il concetto secondo cui un IPv4 address ha due parti: la prefix part e la host part.

 **Classfull Addressing**, il concetto secondo cui un IPv4 address ha 3 parti: network, subnet e host.

**Step per trovare Mask, /P, S, H**



Un altro esempio:

