Software di interconnessione
Point-to-Point Protocol

Il protocollo PPP (*Point-to-Point Protocol*) definisce un metodo standard per trasportare datagrammi generati da protocolli di livello *network* su link punto-punto.
Il protocollo PPP è costituito da 3 componenti principali:

- Un modo per incapsulare datagrammi di livello superiore.
- Un protocollo per il controllo del link LCP (*Link Control Protocol*) per instaurare, configurare e testare connessioni a livello *Data-Link*.
- Un insieme di protocolli di controllo della rete NCP (*Network Control Protocol*) per selezionare e configurare diversi protocolli di livello *Network*.

Il protocollo PPP è stato progettato per semplici link che consentono il trasporto di pacchetti tra due *peer*. Questi link sono *full-duplex* e si assume che consegnino i pacchetti nello stesso ordine con cui sono stati spediti.
L'incapsulamento del protocollo PPP consente simultaneamente il *multiplexing* di differenti protocolli di livello *Network* sullo stesso link. Tale meccanismo è stato appositamente progettato per essere compatibile con la totalità dell'*hardware* esistente.
Per essere sufficientemente versatile e portabile rispetto ad una varietà di ambienti di esecuzione, il protocollo PPP fornisce un meccanismo LCP. Il protocollo LCP viene utilizzato per accordarsi sulle opzioni di incapsulamento, per definire il formato e la lunghezza dei pacchetti, per individuare errori e/o cattive configurazioni e per terminare un link.
L'incapsulamento PPP è utilizzato per distinguere i datagrammi di protocollo superiore che possono essere trasferiti attraverso il protocollo PPP. I pacchetti del protocollo PPP sono delimitati da due *flag*, uno iniziale ed uno finale e da tre campi (*Protocol*, *Information* e *Padding*). Il significato dei campi è il seguente:

1. *Protocol* - Il campo *Protocol* è costituito da uno o due *byte* ed indica il protocollo di livello superiore a cui è destinato il datagramma contenuto nel campo *Information*. Il *byte* più significativo di questo campo viene trasmesso per primo.
2. *Information* - Il campo *Information* è costituito da zero o più *byte*. Tale campo contiene il datagramma di livello superiore che deve essere trasmesso attraverso il protocollo PPP. La lunghezza massima per il campo *Information*, incluso il campo *Padding* ma escluso il campo *Protocol*, viene definita *Maximum Receive Unit* (MRU), il cui valore di *default* è 1500 *byte*. Tale valore può essere modificato in base a degli accordi tra i due *host* che si trovano agli estremi del link PPP.
3. *Padding* - Il campo *Padding* serve per completare il campo *Information* in modo tale che si arrivi al valore MRU che è stato stabilito dai due *host* che comunicano attraverso il protocollo PPP.

L'instradamento o routing

La funzione fondamentale dell'instradamento (*routing*) consiste nell'inoltro (*forwarding*) di pacchetti ed avviene generalmente in modalità *store-and-forward* (memorizza ed inoltra). La necessità di ricevere completamente il pacchetto prima di ritrasmetterlo introduce un tempo di latenza pari al tempo di trasmissione.
Le tecniche fondamentali di inoltro, che differiscono per il metodo di analisi del problema instradamento, sono le seguenti:

- ***Routing by network address***. L'indirizzo di un sistema, che deve essere univoco sulla rete, è scritto direttamente nel pacchetto. Gli IS (*Intermediate System*) usano tale indirizzo come chiave di ricerca nella loro tabella di instradamento e determinano lungo quale cammino il pacchetto debba essere ritrasmesso. Tale tecnica è usata nei *transparent-bridge* (livello OSI

2), e in IP. È in generale adottata dai protocolli non connessi.
- ***Label swapping***. È generalmente usata nei protocolli connessi e trova applicazioni in ATM. Ogni pacchetto è marcato con una *label* che serve come chiave in una tabella di instradamento sull'IS. L'IS, prima di ritrasmettere il pacchetto, sostituisce la *label* con una nuova *label*. Le *label* devono quindi essere univoche solo all'interno di un dato link. Se il protocollo è connesso, le *label* altro non sono che gli identificativi delle connessioni.
- ***Source routing***. È una tecnica usata tramite una opzione del protocollo IP (per esempio, dai *bridge Token Ring*). Nel *source routing* la lista degli IS da attraversare, è scritta nel pacchetto dal nodo mittente, che lo chiede ad un IS o lo scopre con meccanismi di *route location*.

La tecnica presa in esame in questa trattazione sarà la prima, poiché è quella adottata negli schemi di instradamento IP, e quindi integrata nei protocolli e nei *router* IP.

Routing - Definizioni

Con la dicitura rete fisica si indica un insieme di calcolatori aventi le interfacce di rete attestate su una stessa sottorete, in cui una particolare tecnologia di trasporto assicura la connessione.

Una rete logica è l'insieme delle interfacce, a cui è stato assegnato lo stesso indirizzo di *subnet*, che possono comunicare senza dover passare attraverso un *router* (instradatore). Tale condizione viene detta di *routing* implicito. IP assumeva originariamente una corrispondenza biunivoca tra reti fisiche e logiche; realizzazioni più moderne ammettono anche più reti logiche nella stessa rete fisica.

Il *routing* tra reti logiche diverse è esplicito ed è gestito dai *router* tramite tabelle di instradamento.

IP adotta i concetti di destinazioni dirette e indirette nella sua logica di *routing*.

Un *host* diretto è una stazione collegata direttamente alla rete ed al *router* della rete, mentre un *host* indiretto è un *host* di destinazione situato su una rete diversa da quella dell'*host* di origine; questo significa che il datagramma deve essere inviato ad un *router* intermedio prima di essere consegnato all'*host* di destinazione.

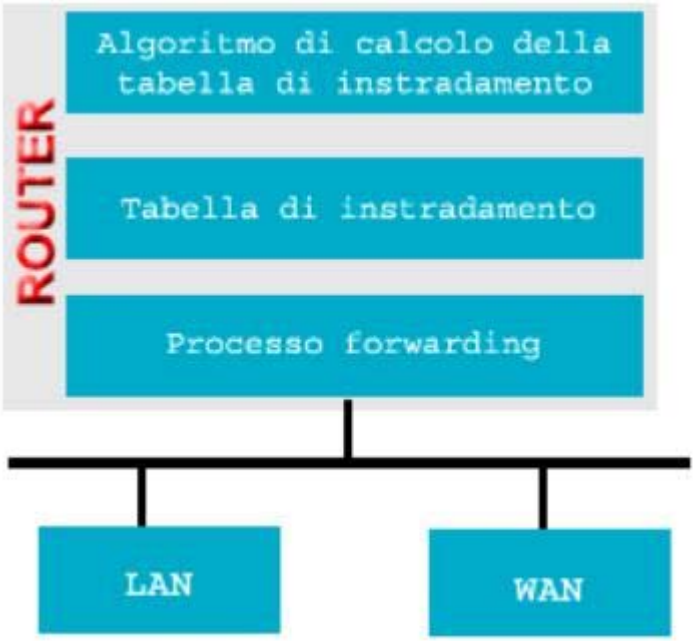
Il modo in cui IP gestisce gli indirizzi e decide i percorsi di *routing*, richiede che una macchina esamini solo la parte di indirizzo di rete dedicata all'indirizzo di destinazione, per determinare se l'*host* di destinazione è collegato direttamente o indirettamente alla rete dell'*host* di origine: in altri termini, la macchina verifica la corrispondenza della parte rete dell'indirizzo di destinazione e sceglie se effettuare un *forwarding* diretto o *forwarding* indiretto.

- *Forwarding* diretto: la trasmissione di un datagramma IP tra due *host* connessi su una singola rete logica IP (stesso *netid*): non coinvolge i *router*. Il trasmettitore incapsula il datagramma nel *frame* fisico e lo invia direttamente all'*host* destinatario.
- *Forwarding* indiretto: i datagrammi passano da un *router* all'altro finché non raggiungono un *router* che può trasmetterli direttamente. I *router* realizzano l'interconnessione tra le diverse reti.

Tabella di instradamento: ogni *router* contiene una tabella di instradamento, visto che se un pacchetto viene destinato al *router* questo dev'essere instradato. Ogni riga nella tabella deve contenere almeno i seguenti tre elementi:

- un indirizzo di destinazione: il *router* può avere più di un percorso per la stessa destinazione.
- L'interfaccia su cui inoltrare i pacchetti.
- Il costo per raggiungere la destinazione sul percorso, che inizia con l'interfaccia indicata nella riga.

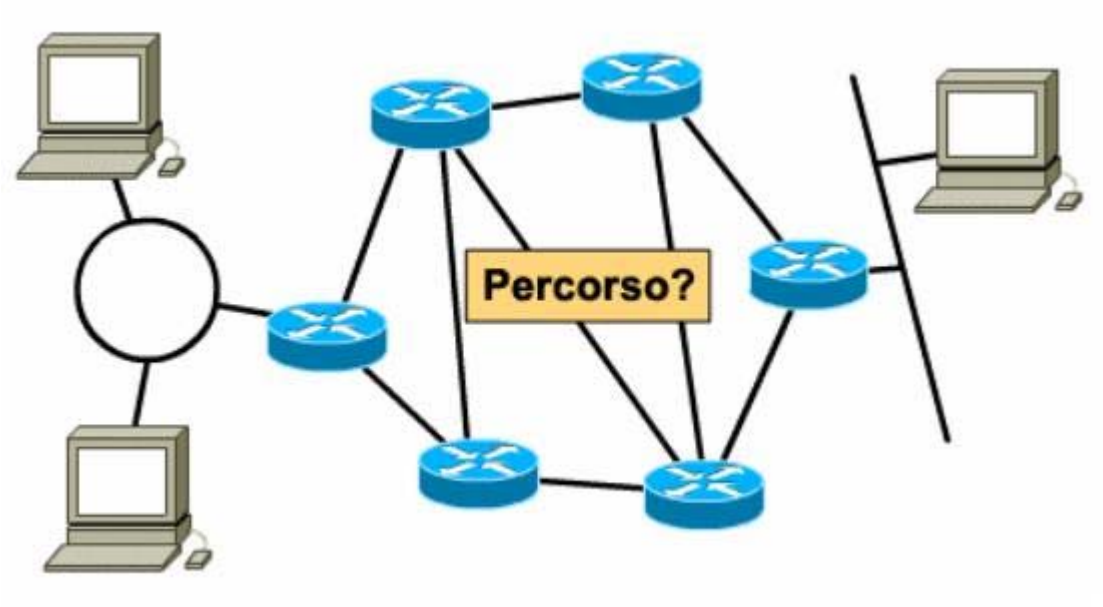
Il costo consentirà all'IS di scegliere tra eventuali percorsi alternativi; l'unità di misura di questo costo dipende dal protocollo utilizzato. Si indicano genericamente con il termine *route* le informazioni presettate su una riga della tabella di *routing*.



Quando il *router* deve inoltrare un pacchetto, scorre la tabella per individuare la riga corrispondente al destinatario del pacchetto stesso. Mediamente il tempo di ricerca (*table lookup*) è pari alla metà del numero di righe. Considerando che tale operazione viene eseguita ogni volta che si deve inoltrare un pacchetto, diventa molto critica la complessità della tabella ai fini delle prestazioni dell'apparato.

Routing by network address

Nel *routing by network address*, la ricerca non verrà basata sull'intero indirizzo del destinatario, ma su un prefisso, molto spesso di lunghezza variabile. La ricerca dovrà essere eseguita nei confronti di quella riga che specifica il *route* con più lungo prefisso comune all'indirizzo del destinatario (*longest prefix matching*).
Affinché i pacchetti arrivino a destinazione è indispensabile che le tabelle nei vari IS siano coerenti tra di loro, al fine di evitare l'invio di pacchetti in percorsi ciclici (*routing loop*). In tal caso i pacchetti girerebbero a vuoto, consumando inutilmente risorse computazionali e trasmissive dei vari *router*.



Dal percorso lungo il quale un pacchetto viene inoltrato, dipendono il ritardo che esso subirà, la probabilità che venga scartato a causa di eventuali congestioni del IS e il fatto che esso raggiunga o no la destinazione. Inoltre se la rete contiene maglie, una destinazione potrà essere raggiunta attraverso una o più percorsi alternativi; in presenza di guasti, la scelta di un percorso che eviti nodi o collegamenti non funzionanti consentirà alla rete di continuare a recapitare dati.
Dunque la scelta del percorso, in altre parole il *routing*, sarà un fattore chiave per il buon funzionamento della rete e per la sua robustezza (*fault-tolerance*).

Classificazione degli algoritmi di routing

Gli algoritmi di *routing* possono essere classificati per tipo:

- Statico: negli algoritmi statici, le tabelle di *routing* che vengono memorizzate sono compilate da una persona (amministratore di rete) e i valori di tali tabelle non cambiano per nessun motivo fino a quando l'amministratore di rete non li cambia, mentre negli algoritmi dinamici le tabelle vengono continuamente aggiornate e cambiate secondo i cambiamenti della rete (caduta di una rete, inserimento di una rete).
- Gerarchici: i *router* gerarchici hanno funzioni diverse da quelli che non lo sono, poiché vengono suddivisi più nodi in gruppi logici chiamati domini di *routing*, *autonomous system* o aree. Solo alcuni di questi *router* possono interagire con ulteriori *router* di altri domini di *routing*, mentre altri possono interagire con *router* appartenenti allo stesso dominio.
- *Link-State*: *link-state* (conosciuto anche come *shortest path first*) trasferisce tutte le informazioni di *routing* a tutti i nodi: ogni *router* invia solo la porzione di tabella che descrive lo stato dei suoi link. Gli algoritmi del tipo *distance-vector* inviano tutta o parte della tabella ai soli *router* vicini. Quindi *link-state* spedisce piccoli aggiornamenti a tutti, *distance-vector* spedisce grossi aggiornamenti ma solo ai *router* vicini: i *link-state* richiedono più risorse *hardware* (*CPU* e memoria) rispetto ai *distance-vector*, ma sono meno propensi ai *routing loop*.