Link Layer - introduzione

lunedì 28 agosto 2023 17:05

Ci sono due tipologie di canali:

1. la tipologia di **canali broadcast,** dove un gruppo di host si connette alle: LAN wireless, reti satellitari, reti di accesso HFC… Questo avviene tramite il protocollo *Medium Access Protocol* per coordinare la trasmissione dei frame gestendo l'accesso al mezzo;
2. **La tipologia di canale di comunicazione punto a punto**, implementato dal protocollo **PPP** *(point-to-point Protocol)* ha una più semplice gestione ed è il più utilizzato.

L'idea di base è sempre la stessa: ogni nodo (dispositivo di rete) che deve spedire un datagramma, lo incapsula in un frame a livello di collegamento e lo immette nel collegamento stesso. Facciamo un'analogia, Immaginiamo di avere un'agenzia di viaggi che deve programmare un viaggio per una persona (datagramma) da una città A ad una città D; per fare questo l'agenzia (protocollo d'instradamento) programma un viaggio in macchina da A a B, un viaggio in treno da B a C e infine un viaggio in aereo da C a D, il tratto di viaggio è il collegamento e il modo in cui il passeggero viaggia è il protocollo utilizzato dal livello di collegamento.

Servizi del livello di collegamento

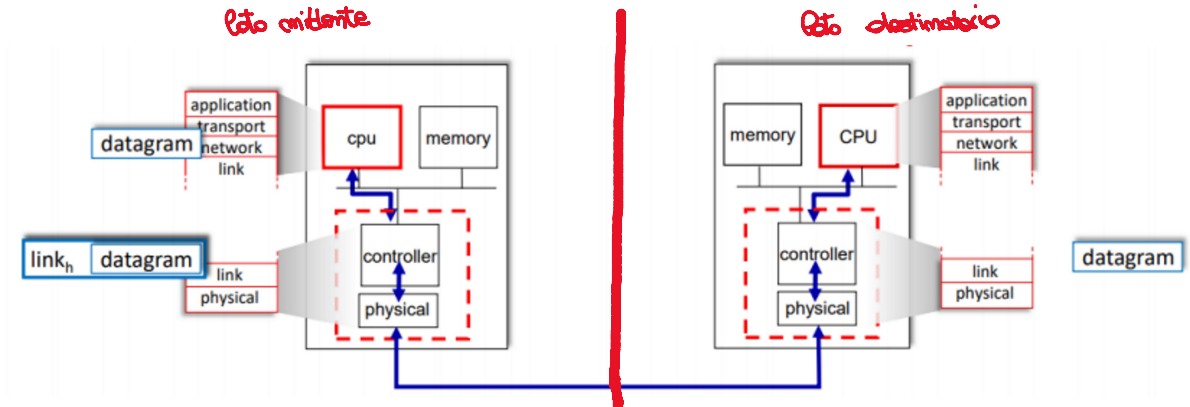
i servizi offerti dal livello di collegamento non vengono implementati tutti da tutti i protocolli, ogni protocollo ha un suo **modello di servizio**.

Vediamo generalmente tali servizi:

- **Framing**: quasi tutti i protocolli incapsulano i datagrammi in un frame prima di trasmetterlo; anch'esso è costituito da campi di intestazione e campo dati;
- **Accesso al collegamento**: le regole necessarie per immettere i frame nel mezzo trasmissivo vengono dettate dal protocollo che controlla l'accesso al mezzo in questione, il protocollo MAC (Medium Access Control);
- **Consegna affidabile**: i protocolli di collegamento gestiscono la consegna affidabile del datagramma, diverso dal controllo del livello di trasporto poiché fatto a livello software, a livello di collegamento si ha un controllo hardware, comunque la metodologia è simile, abbiamo ACK e ritrasmissioni, in alcuni casi ci sono anche metodi per correggere l'errore localmente;
- **Rilevazione e correzione di errori**: alcuni protocolli implementano un servizio di rilevazione e correzione degli errori, inserendo da parte del nodo che trasmette dei bit di controllo.

Per un dato collegamento, il protocollo del livello di collegamento, è realizzato da un adattatore di rete (network adapter), noto come **scheda di rete**, abbiamo già visto che questa risiede all'interno delle porte.

La scheda di rete ha un controller a livello di collegamento che implementa i servizi precedentemente elencati, la maggior parte delle funzionalità sono implementate in hardware.

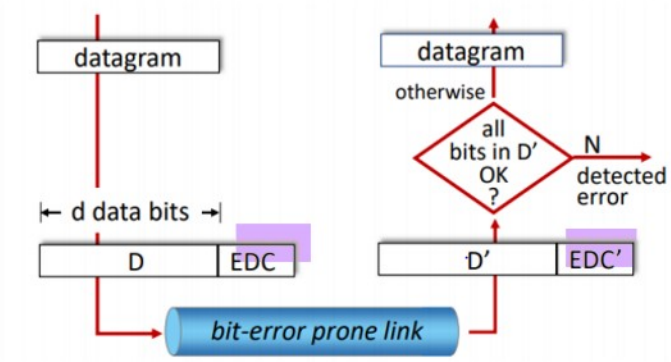


Viene gestito dal proprio controller, questo prende un datagramma passato dai livelli superiori e lo incapsula in un frame, ponendo i campi d'intestazione, a livello di collegamento; segue poi il protocollo di accesso al collegamento, infine il frame viene immesso dal controller sul collegamento stesso.

Fa l'opposto il controller riceve il frame da cui estrapola il datagramma e lo consegna al livello superiore

Notiamo come il livello di collegamento viene implementato sulla scheda di rete, ma tutta la parte di assemblaggio e di verifica viene seguita dalla CPU, quindi a livello software. Possiamo quindi dire che il livello di collegamento sia effettivamente il punto cruciale della pila dove si incrociano hardware e software.

Tecniche di rilevazione e correzione errori

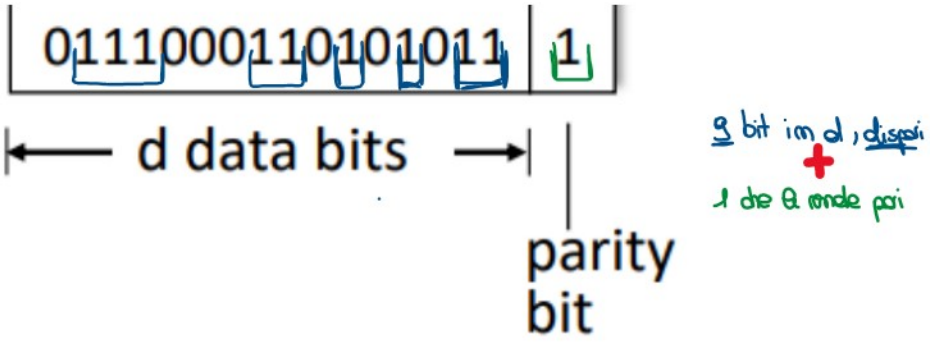


EDC è l'acronimo di *Error Detection and Correction*, che sono sostanzialmente dei bit che ci consentono di rilevare un errore; tuttavia si può verificare che gli errori non vengono comunque rilevati e quindi consegnare un datagramma alterato a livello di rete.
3 tecniche più famose per il controllo e correzione degli errori sono: **controllo di parità, check sum e controllo di ridondanza ciclica.**

Controllo di parità

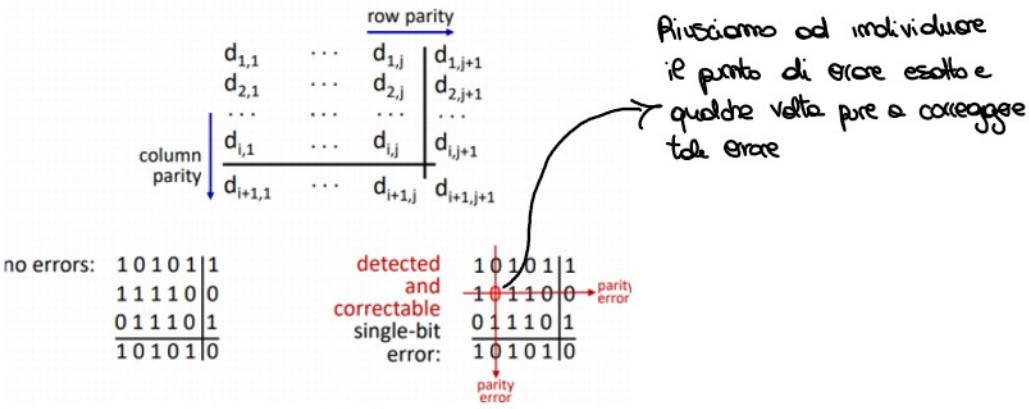
In questo tecnica utilizziamo un unico bit di parità il nostro e dc bit di rilevazione e correzione degli errori. Supponiamo di dover rinviare *d* bit, il mittente aggiunge un bit di parità a questi *d* bit, e qui possono accadere due situazioni:

1. schema di parità pari: il mittente sceglie il valore del bit di parità da inserire in modo tale da rendere pari il numero totale di bit ad 1 nei *d* + 1 bit trasmessi in totale…



2. **Schema di parità dispari**: il mittente sceglie il valore del bit di parità da inserire in modo tale che ci sia un numero pare di d bit ad 1, solo che questi sono già pari, il bit di parità sarà quindi "dispari", *il contrario dello scenario precedente*…Quindi abbiamo numero di 1 pari in *d* bit allora metto il bit di parità a zero, dato che *d* bit 1 sono già pari.

Il ricevente deve contare il numero di bit ad 1 tra quelli ricevuti, se trova un numero dispari di bit ad 1 sa che si è verificato almeno un errore in un bit, ma al momento non sappiamo ancora identificare dove.
Ora dividiamo i d bit in i righe e j colonne per ognuna delle quali è stato calcolato un valore di parità: i risultanti **i** *(bit di parità delle righe)* + **j** *(bit di parità delle colonne)* + **il bit di parità per la rilevazione dell'errore**, struttura uno schema di parità bidimensionale…



La capacità del ricevente sia di rilevare che di correggere gli errori è conosciuta come **Forward Error Correction: FEC**, *correzione degli errori in avanti*.

Checksum in Internet

Nelle tecniche che usano la checksum abbiamo *d* bit di dati trattati come una sequenza di numeri interi da *k* bit.
Per eseguire l'operazione di checksum, dobbiamo sommare questi interi da *k* bit, i bit risultanti dalla somma vengono

usati come bit per la rilevazione degli errori.
Il checksum di Internet si basa su questo approccio, infatti i dati sono trattati come interi a 16 bit e sommati.
Il complemento ad 1 dei bit risultanti dalla somma costituiscono il checksum di Internet che viene trasposto nell'intestazione dei segmenti.
Il ricevente controlla il checksum calcolando a sua volta il complemento ad 1 della somma dei dati ricevuti *(compreso il checksum stesso)* e verifica che tutti i bit del risultato siano 0, altrimenti viene segnalato l'errore.

Codici di controllo e ridondanza ciclica

Consideriamo sempre i nostri *d* bit costituenti i dati *D* da trasmettere, inoltre sorgente e destinatario si sono accordati su una stringa **generatore *G*** di *r+1* bit.
Dato il blocco di dati D, il mittente sceglierà *r* bit addizionali, costituente l'insieme ***R***, in modo tale da ottenere una stringa *d+r* che sia divisibile per il generatore *G* nell'aritmetica modulo 2.
A questo se la divisione ***d+r/G*** ha un resto diverso da 0 il ricevente sa che si è verificato un errore, altrimenti i dati sono corretti.
Abbiamo che:

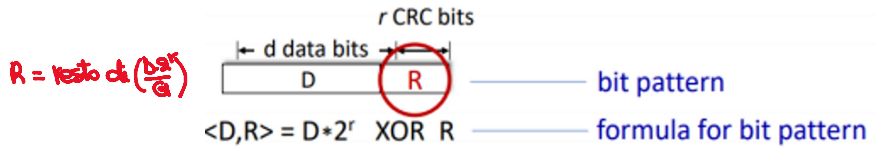
- Addizione e sottrazione equivalgono all'operazione XOR, quindi D + R == DXOR R;
- La moltiplicazione di una stringa 2^k corrisponde allo *shift* a sinistra della stringa di *k* posizioni.

Quindi dato D ed R, la quantità D * 2^r XOR R fornisce come risultato la stringa di lunghezza pari a d+r bits che il mittente invierà. Vediamo ora come R viene calcolato…

Nota: *tutti i calcoli CRC sono espressi in aritmetica modulo 2.*

Nota2: *è necessario che il bit più a sinistra del generatore sia ad 1.*

Nota3: *R è scelto in modo tale che D * 2^r XOR R sia divisibile per G*



Il ricevente poi dovrà fare la divisione con *G*, che conosce autonomamente, e vedere se vi sono errori, se il resto della divisione è diverso da 0 sono stati rilevati errori!
Sono stati definiti dei generatori standard da 8, 12, 16, 32 bit.