Checksum

Gruppo di bit di controllo associati al messaggio. Di solito, è posizionato alla fine del messaggio e calcolato come **complemento a uno del risultato della somma dei dati**. In questo modo gli errori possono essere rilevati sommando l'intera parola contenente sia i bit dati sia il checksum.

CRC

È una tecnica di error detection basata sull'**aritmetica polinomiale** in base 2. Una stringa di m bit viene interpreta come un polinomio di grado $m-1$ che ha come coefficienti i bit della stringa.

La sorgente e la destinazione si accordano su un **polinomio generatore** $G(x)$, che deve avere come primo ed ultimo bit 1. Il frame da controllare $M(x)$ deve essere di ordine maggiore di $G(x)$. L'idea è aggiungere una specie di checksum alla fine del frame in modo che il polinomio rappresentato dal frame, checksum compreso, sia divisibile per $G(x)$.

Il calcolo del checksum avviene in questo modo:

1. sia r il grado di $G(x)$. Si aggiungono r zeri alla fine del frame, in modo da ottenere una sequenza $x^r M(x)$ composta da $m+r$ bit;
2. si divide tale sequenza per $G(x)$ tramite la divisione in modulo 2;
3. si sottrae il resto ottenuto dalla divisione a $x^r M(x)$, sempre in modulo 2;
4. la sequenza risultante sarà il frame con il checksum pronto per la trasmissione.

La destinazione riceve il polinomio e prova a dividerlo per $G(x)$, ossia eseguo uno shift a sinistra di r bit. Se c'è **resto allora si è verificato un errore**. Dato che un errore di trasmissione può essere visto come sommare un polinomio E(x) al frame, allora è possibile fare error detection per tutti gli errori E(x) che non sono divisibili per G(x).

Esempio:

$$M(x) = 10011101$$

$$G(x) = x^4 + x + 1 = 10011$$

$$F(x) = x^4 M(x) + R(x) = 100111010000 + 1111 = 100111011111$$

2.3 Protocolli per il controllo di flusso

2.3.1 Stop-and-wait

È piuttosto elementare e si usa in canali simplex o half duplex. Il principio è il seguente: il mittente trasmette un frame e **aspetta** che chi è dall'altro capo

del canale invii una **conferma di ricezione**, detta **ACK**, *Acknowledgement*. Un chiaro svantaggio di questo protocollo è l'**elevato tempo di attesa** tra le trasmissioni dei frame: Stop-and-wait **non funziona bene in presenza di roundtrip delay elevato**.

Si possono verificare due errori:

Il frame non arriva a destinazione Il mittente si trova bloccato nell'**aspettare l'ACK, che non arriverà mai**, rendendo necessaria la presenza di un **timeout** dopo il quale il frame venga inviato nuovamente.

L'ACK non arriva al mittente Il destinatario riceve il frame ma al mittente non arriva la conferma: dopo il timeout descritto precedentemente i dati vengono ritrasmessi. Tuttavia, in questo modo **il destinatario riceve due volte lo stesso frame**. La soluzione consiste nel dotare i frame di un campo che lo etichetti con un numero; basta anche un solo bit, per indicare il precedente dal successivo.

2.3.2 I protocolli sliding window

Ogni partecipante alla conversazione deve tener sotto controllo **due finestre**: quella dei frame in **entrata** e quella dei frame in **uscita**. Ogni frame in uscita contiene un numero di sequenza e il destinatario deve tener traccia di questi per la ricezione, mentre il mittente per l'invio.

Quando il mittente **invia un frame, resta nella finestra** finché non viene ricevuto il corrispondente ACK. Dopodiché la finestra verrà aggiornata. Quando il destinatario **riceve il frame**, controlla che il numero sia uguale a quello che si aspettava e ne invia l'ACK. Se quest'ultimo contiene il numero che la sorgente si aspettava, prosegue nella trasmissione inviando un nuovo frame. Altrimenti, invia nuovamente quello segnato nel buffer.

Si possono **inviare più frame contemporaneamente** prima di entrare in attesa (pipelining). Il destinatario aggiorna la finestra non appena riceva il frame e invia l'ACK. In caso di pipelining, i protocolli **go back n** e **selective repeat** permettono di gestire i problemi dovuti a questo tipo di trasmissione.

Go back n

E' il protocollo a finestra scorrevole più semplice. La finestra di chi riceve ha ampiezza 1 mentre quella di chi trasmette ha ampiezza N. Vengono inviati fino a N pacchetti alla volta con un'etichetta che indica lo slot della finestra, anche se il ricevente li gestisce comunque uno alla volta.

La destinazione, una volta accortasi che un frame è corrotto, **scarta a prescindere tutti i frame successivi** (per numero di sequenza) già ricevuti. Per i frame scartati non invia ACK, ma aspetta che scadano i timeout nel mittente, il quale provvederà a trasmetterli nuovamente.

Go back n **funziona bene quando il roundtrip delay è elevato e il canale è affidabile**. Si noti che il mittente deve essere in grado di gestire N timer per rispedire i pacchetti e avere un buffer capiente in cui tenerne una copia da ritrasmettere.

Selective repeat

Evoluzione del Go back n, nella quale anche il ricevente ha un buffer per contenere più frame contemporaneamente. Il destinatario conserva tutti i frame validi e li salva in un buffer. La sorgente continua ritrasmettere i frame per i quali, prima della scadenza del proprio timeout, non ha ricevuto l'ACK. Se la destinazione riceve un frame corrotto, invia un **NACK** (*Not ACK*) per sollecitare la ritrasmissione prima della scadenza del timeout da parte del mittente.

Si può inoltre inviare un NAK per indicare al mittente che un frame potrebbe essere stato perso e diminuire ulteriormente lo spreco di tempo (Il timer del NAK deve essere minore del timer di rinvio). Questo protocollo **sfrutta il più possibile il canale**, però richiede maggiori risorse dato che è necessario gestire un timer e una parte di buffer per ogni slot aperto della finestra. Un **problema** di questo protocollo (delle sliding windows in generale) è che l'**apertura massima della finestra deve essere al più uguale alla metà delle etichette disponibili per evitare la perdita di sincronizzazione**.

Piggybacking

È una tecnica che consiste nello **sfruttare un messaggio del destinatario al mittente come *passaggio* per il messaggio di conferma** ACK. Il campo ACK è posto nell'header del frame. Chiaramente, il piggybacking dell'ACK si usa solo se vi è un messaggio del destinatario al mittente nell'immediata possibilità d'inviare la conferma, altrimenti si invia l'ACK separatamente. Di conseguenza questa tecnica funzione bene quando la comunicazione tra le due parti è bilanciata.

2.4 PPP

Il protocollo PPP, *Point to Point Protocol*, viene utilizzato per gestire la configurazione della rilevazione d'errore di una linea, supportare molteplici protocolli, permettere l'autenticazione e molto altro. Provvisto di numerose opzioni, il protocollo PPP ha le seguenti tre caratteristiche principali:

- un metodo di framing che permette di delimitare in modo non ambiguo la fine di un frame e l'inizio del successivo. Il formato del frame permette di gestire anche la rilevazione di errori;

- un protocollo per gestire la connessione, il test della linea, negoziare le opzioni di collegamento e gestire la disconnessione in modo pulito quando la linea non serve più. Questo protocollo è chiamato **LCP**, *Link Control Protocol*;

- una modalità per negoziare le opzioni relative al livello di rete, in modo indipendente dall'implementazione di tale livello che verrà usata per la comunicazione. Il metodo scelto avrà un diverso NCP (*Network Control Protocol*), un protocollo di controllo della rete, per ogni livello di rete supportato.

Questo protocollo viene largamente utilizzato nelle connessioni DSL (per il collegamento utente-provider) e GPRS.

Nome	Numero di bytes	Descrizione
Flag	1	indica l'inizio o la fine del frame
Address	1	indirizzo broadcast
Control	1	byte di controllo
Protocol	2	indica il protocollo del campo data
Data	variabile (da 0 a 1500)	campo di dati
FCS	2 (o 4)	somma di correzione

Figura 2.1: Frame PPP

Il campo più importante di un frame PPP è il campo **Protocol** che specifica il tipo di protocollo PPP in uso: se il primo bit è a 1 si sta usando LCP mentre se è a 0 NCP. Ci sono 11 tipi di frame LCP, 4 di configurazione 2 di terminazione, 2 di rifiuto, 2 di echo e 1 di test.

Configurazione: configure-request, configure-Ack, configure Nak, e configure-reject, vengono usati per stabilire e configurare la connessione, si può scegliere la lunghezza del campo dati, che livello di error-detection usare e se trasmettere o meno i campi Address e Control.

Terminazione: terminate-request e terminate-ack.

Rifiuto: code-reject (richiesta sconosciuta) e protocol-reject (protocollo richiesto non supportato).

Echo: echo-request e echo-reply, per il test della qualità della rete.

Test: discard-request

I frame PPP poi possono essere incapsulati in frame Ethernet (PPPoE) o ATM (PPPoA).

4.3.6 Principali differenze tra IPv4 e IPv6

- indirizzi più lunghi: 16 byte di IPv6 contro i 4 di IPv4. Questo si traduce nel **supporto a miliardi di host anche con un'allocazione di indirizzi altamente inefficiente**;
- **semplificazione** dell'intestazione. Ad esempio, si è deciso di togliere l'error detection per rendere la trasmissione più veloce. Dopotutto, era ridondante e rallentava l'instradamento dato che il decremento del campo TTL per ogni nodo attraversato rendeva necessario il ricalcolo del checksum;
- opzioni del protocollo: in IPv4 erano limitate a 40 byte mentre con IPv6, grazie al campo *Next Header*, sono potenzialmente illimitate. Lascia quindi **spazio ad evoluzioni future**;
- IPv6 fornisce un grado di sicurezza maggiore;
- IPv6 permette a un host di spostarsi senza cambiare il suo indirizzo;
- Il vecchio protocollo può coesistere con quello nuovo.

4.4 Altri protocolli

4.4.1 DHCP

DHCP, *Dynamic Host Configuration Protocol*, è un protocollo che permette l'assegnazione manuale o automatica dell'indirizzo IP. L'idea di base è semplice: esiste un server *speciale* che assegna gli indirizzi alle macchine che lo richiedono. Questo server può trovarsi sulla stessa LAN del richiedente.

Quando una macchina vuole ottenere un indirizzo IP, manda in broadcast un particolare pacchetto chiamato *DHCP DISCOVER*. Il gestore di rete centralizzato risponde alla macchina inviando un pacchetto contenente un indirizzo. Dato che nella stessa rete ci possono essere più gestori DHCP, l'host richiedente deve confermare al gestore che ha accettato l'IP proposto, il quale dovrà chiudere l'accordo con un ACK.

Essendo la rete dinamica, le informazioni dopo un po' di tempo devo essere rinnovate mediante un meccanismo di fading chiamato **leasing** secondo il quale sia il gestore sia la macchina interessata sanno quando scade l'informazione, in modo da poterla rinnovare prima che scada.

4.4.2 ARP

ARP, *Address Resolution Protocol*, è un protocollo di servizio appartenente alla suite del protocollo Internet IPv4, il cui compito è fornire la *mappatura* tra indirizzo IP e MAC di un terminale in una rete locale. Il suo analogo in IPv6 è NDP, *Neighbor Discovery Protocol*.

Per inviare un pacchetto IP, è necessario incapsularlo in un frame di livello data link, che dovrà avere come indirizzo di destinazione il MAC address dell'host a cui lo si vuole inviare. ARP viene utilizzato per ottenere questo indirizzo. Se il pacchetto deve essere inviato ad un'altra sottorete, l'indirizzo MAC da richiedere sarà quello del gateway o del router.

Ogni macchina connessa alla rete conserva in memoria una tabella, detta *ARP Cache*, con le corrispondenze IP-MAC già precedentemente richieste, in modo da evitare d'interrogare continuamente la rete per inviare ogni pacchetto. Le informazioni contenute in questa cache vengono cancellate dopo un certo periodo di tempo, tipicamente 5 minuti.

Quando un host deve inviare un pacchetto ad un IP non presente in tabella, manda in broadcast un messaggio detto *ARP REQUEST*, contenente il proprio indirizzo MAC e l'indirizzo IP della macchina di cui si vuole conoscere l'indirizzo IP. Tutti gli host della rete ricevono la richiesta e, in ciascuno di essi, il protocollo ARP verifica se viene richiesto il proprio indirizzo MAC confrontando il proprio IP con quello ricevuto. L'host che verifica positivamente questa corrispondenza provvede ad inviare in **unicast** una risposta, *ARP Reply*, contenente il proprio MAC.

Per ottimizzare il protocollo, una macchina, quando si collega alla rete, manda in broadcast la richiesta della propria associazione IP-MAC, così tutte le altre macchine la possono memorizzare. Inoltre, se qualcuno risponde significa che si è verificato un errore nell'assegnazione degli indirizzi IP.

Si noti che l'ARP Request ad un nodo aggiorna completamente la tabella ARP presente nella cache, senza rispetto per le voci già esistenti. Particolare molto importante perché causa di diversi attacchi chiamati *ARP Spoofing*, che verranno approfonditi nella sezione dedicata alla sicurezza delle reti.

4.4.3 NAT

Il NAT, *Network Address Translation*, costituisce una soluzione adottata per risolvere il problema dell'esaurimento degli indirizzi IP, **simulando una rete in un unico indirizzo**. Ad esempio, un'azienda può avere un solo indirizzo IP per il traffico Internet. Internamente, ogni host riceve un indirizzo univoco per instradare i dati nella rete aziendale. Quando il traffico esce da questa rete avviene una traduzione dell'indirizzo a quello di Internet, effettuato dal dispositivo NAT.

Il problema sta nel decidere a chi inviare internamente la risposta inviata dal Web, che come indirizzo ha quello generico aziendale. Siccome, la quasi totalità dei pacchetti IP trasporta carichi utili TCP e UDP, si è pensato di sfruttare queste informazioni per implementare l'inoltro. TCP e UDP, come si vedrà in seguito, utilizzano e includono nei propri header una porta sorgente e una di destinazione del pacchetto.

Ogni volta che un pacchetto diretto verso l'esterno raggiunge l'apparto NAT, oltre a sostituire l'indirizzo, viene sostituito anche il campo *Source*

port, con un indice che punta alla tabella di traduzione NAT. Al contrario, quando viene ricevuta la risposta, si usa il campo *Source port* ricevuto viene usato per ottenere l'indirizzo e la porta del mittente originale all'interno della rete NAT.

Sebbene questo schema in un certo senso risolva il problema dell'esaurimento degli indirizzi IP, molti della comunità lo considerano un vero e proprio abominio.

Viola il modello gerarchico di IP Non è più vero che ogni indirizzo IP identifica a livello mondiale una singola macchina.

Rompe il modello di connettività end-to-end Poiché l'associazione viene effettuata dai pacchetti in uscita, i pacchetti di entrata non possono essere accettati se non dopo quelli in uscita. Esistono comunque alcune tecniche per arginare questo problema.

Trasforma Internet in una rete orientata alla connessione Questo perché l'apparato NAT deve conservare le informazioni relative a ogni connessione che lo attraversa. **Se l'apparato NAT si blocca, tutte le sue connessioni TCP vanno perse**. Questo non succede in assenza di NAT.

Viola la regola principale della stratificazione dei protocolli Di norma, il livello k non deve essere costretto a formulare alcuna ipotesi su ciò che il livello $k + 1$ ha inserito nel payload. NAT distrugge questa indipendenza perché si basa sui livelli superiori TCP e UDP. Se in futuro TCP venisse aggiornato a TCPv2, con uno schema d'intestazione diverso, NAT non funzionerebbe più.

I processi su Internet non sono obbligati a usare TCP e UDP Se due utenti si accordassero per usare un nuovo protocollo di trasporto, il meccanismo di NAT non funzionerebbe più.

Alcune applicazioni usano più connessioni TCP o porte UDP NAT non sa gestire queste situazioni, a meno che non vengano prese speciali precauzioni.

Nonostante ciò, NAT è molto usato nella pratica, specialmente per reti domestiche o in piccole aziende, in quanto è l'unica tecnica che riesca a gestire il problema della mancanza di indirizzi IP. Per questa ragione è inverosimile che sparisca, anche qualora IPv6 fosse ampiamente adottato.