1 Lezione del 29-10-25

Riprendiamo l'argomento dei datagrammi IP.

1.0.1 Overhead di TCP/IP

Ricordiamo che il datagramma IP ha 20 byte di overhead per l'header. Visto che spesso trasporta payload rappresentato da segmenti TCP, che hanno anch'essi 20 byte di overhead, possiamo dire che l'overhead complessivo è $20 + 20 = 40$ byte di informazioni non direttamente collegate al payload che vogliamo trasportare.

Abbiamo poi che alcuni campi dell'header IP sono legati al protocollo di trasporto sovrastante. Avevamo già notato infatti il campo *upper layer*, contenente informazioni sul protocollo (TCP o UDP) usato.

1.0.2 Frammentazione IP

Il protocollo IP supporta un meccanismo di **frammentazione** e **riassemblaggio**, usato quando si inviano in rete datagrammi (header + segmento nel payload) più grandi della cosiddetta **MTU** (*Max Transmission Unit*, cioè la dimensione massima dei frame di livello datalink).

I campi **identifier**, **flags** and **fragment offset** dell'header riguardano esattamente questo: l'*identifier* identifica un datagramma, e il *fragment offset* la posizione all'interno del datagramma specificato del frammento corrente. In sostanza, un datagramma è composto da frammenti, che sono anch'essi inseriti datagrammi per la trasmissione in rete.

Il campo *flags*, su 3 bit, determina la modalità della frammentazione. In particolare abbiamo i bit:

```
bit 0: riservato
bit 1: DF (don't fragment)
bit 2: MF (more fragments)
```

- Il campo **DF** (*Don't Fragment*) significa che questo datagramma non dovrà essere frammentato, e che se è più grande della MTU dovrà essere scartato;
- Il campo **MF** (*More Fragments*) indica invece che questo datagramma è seguito da altri frammenti di un datagramma più grande.

  Se vale 0 può significare equivalentemente che è l'ultimo frammento, o che come datagramma non era stato mai frammentato. Il destinatario gestisce datagrammi frammento arrivati fuori ordine controllando se i datagrammi ricevuti finora, coi loro offset, riempiono completamente un dato buffer. Visto che un datagramma frammento arrivato in anticipo lascia necessariamente un "buco" (i datagrammi sono inviati per offset incrementali), questo funziona.

Notiamo a questo punto che il *fragment offset*, che rappresenta effettivamente l'offset del frammento all'interno del datagramma completo, è su 13 bit (mentre la dimensione del datagramma è su 16 bit). Questo significa che misuriamo gli offset non in byte, ma in blocchi da 64 bit (8 byte, in quanto $16 - 13 = 3 \to 2^3 = 8$).

Vediamo quindi come devono comportarsi i router:

- Il **trasmettitore** di livello network avrà quindi il (semplice) compito di, nota l'MTU, suddividere (*frammentare*) i datagrammi troppo grandi in più datagrammi *"di frammento"*, che poi inoltrera singolarmente sulla rete.

  Questo, chiaramente, nel caso il capmo DF non sia abilitato: in caso contrario semplicemente scarterà il datagramma;

- Il **ricevitore** dovrà occuparsi invece di controllare l'header dei datagrammi IP, capire quando è stato inviato un frammento di un datagramma più grande (attraverso il campo *flags*), e aspettare successivamente tutti i frammenti necessari a *riassemblare* il datagramma.

  Nel caso non tutti i frammenti arrivino in un quanto di tempo utile, si scarta il datagramma e si va avanti. Questo non ci turba, in quanto abbiamo detto IP è *best effort*.

1.1 Indirizzamento IP

Iniziamo quindi a vedere nel dettaglio l'**indirizzo IP**.

Avevamo già introdotto l'*indirizzo MAC*, o *fisico*, associato alla NIC e assegnato su base globale (quindi solitamente unico per ogni dispositivo, ecc...). Gli indirizzi IP, come vederemo, sono invece assegnati su base **dinamica**.

Un indirizzo IP è un identificatore su 32 bit per **interfacce** di *host* e *router*. Un'*interfaccia* è la connessione fra host/router e il livello fisico. Solitamente gli host hanno un'interfaccia e i router più di una.

1.1.1 Subnet

Una **subnet**, o *sottorete*, è un insieme di interfacce che possono comunicare fisicamente l'una con l'altra senza l'intervento di un router di livello 3. Sono sottoreti le reti LAN che abbiamo studiato in 14.1.

Gli indirizzi IP sono fatti per gestire le sottoreti:

- La prima parte dell'indirizzo identifica la sottorete, e quindi più dispositivi sulla stessa sottorete hanno gli stessi bit più significativi (un numero variabile di questi, specificato dalla **subnet mask** o *maschera di sottorete*);

- L'ultima parte dell'indirizzo identifica quindi i singoli dispositivi.

Riguardo alla maschera di sottorete, abbiamo che questa è notata come `<addr>/<mask>`, dove `<mask>` è un numero da 0 a 32 che identifica quanti dei primi bit dell'indirizzo sono dedicati alla sottorete.

Possiamo distinguere i seguenti tipi di indirizzo:

- Indirizzi di **tipo A**: `8 sottorete + 24 dispositivo`;

- Indirizzi di **tipo B**: `16 sottorete + 16 dispositivo`;

- Indirizzi di **tipo C**: `24 sottorete + 8 dispositivo`;

1.1.2 CIDR

CIDR sta per *Classless InterDomain Routing*, e rappresenta il formato standard degli indirizzi IP appena descritto:

```
a.b.c.d/x
```

dove `x` è il numero di bit nella porzione di sottorete dell'indirizzo. Notiamo che, a scapito di quanto abbiamo detto sul tipo degli indirizzi nella scorsa sezione, `x` non deve necessariamente essere multiplo di 8 bit, ma può essere su un numero arbitrario di bit fra 0 e 32.

Potremmo interrogarci su come vengono ottenuti gli indirizzi IP.

- Per gli host, la risoluzione dell'IP all'interno della sua sottorete è fatta dall'amministratore di sistema, o dal protocollo **DHCP** *Dynamic Host Configuration Protocol*;
- Per le sottoreti, gli amministratori di rete si rivologono agli ISP, che a loro volta (attraverso una struttura gerarchica) si rivolgono all'**ICANN** (*Internet Corporation for Assigned Names and Numbers*).

1.2 DHCP

Il **DHCP** (*Dynamic Host Resolution Protocol*) è un protocollo di livello application che si occupa di fornire ai client di un server (il server DHCP, che si assume gestisca una certa sottorete) i loro indirizzi IP.

Questo permette agli host di accedere ai seguenti servizi:

- Richiedere ed ottenere un indirizzo IP quando si unisce ad una rete;
- Riutilizzare tale indirizzo finché si mantiene in contatto con la rete.

Si assume che il server DHCP sia collegato al router che server la sottorete, in modo che possa servire tutte le sottoreti su tale router. Solitamente, DHCP sta sulla porta 68.

Quando un host si collega alla rete:

1. Avrà bisogno di un indirizzo IP, e quindi interrogherà la rete (con indirizzo broadcast, `255.255.255.255` (tutti i bit a 1)) per un server DHCP. Questa fase si dice di **DHCP discover**;
2. Il server DHCP, se presente sulla rete, risponderà al DHCP discover con un **DHCP offer**, cioè offrendo un indirizzo IP al client. Notiamo che anche la risposta del server è in broadcast: si disambigua fra host sfruttando un *transaction ID* univoco generato dal client;
3. Il client può quindi rispondere con una **DHCP request**, dove richiede di poter sfruttarel'indirizzo IP usato. Anche questo messaggio è inviato in broadcast: potrebbe essere inviato all'IP del sever (ormai noto dalla fase di offer), ma il protocollo viene definito in questo modo;
4. Il server invia allora l'ultimo messaggio, il **DHCP acknowledge**, inviato sempre in broadcast, che dà la conferma al client di poter iniziare ad usare l'indirizzo IP ottenuto.

Vediamo che le fasi del DHCP rispettano il cosiddetto acronimo **DORA**: *Discover*, *Offer*, *Request* and *Acknowledge*.

Notiamo inoltre che queste richieste sono associate ad un certo *lifetime*, che rappresenta il tempo di vita per cui la richiesta viene presa in considerazione dal server (oltre il lifetime, l'offerta dell'indirizzo IP viene scartata e quell'indirizzo può essere offerto ad altri host).

Il DHCP può fornire più informazioni, non solo l'indirizzo IP:

- Il **default gateway** della rete, cioè il router da contattare per uscire dalla sottorete;
- I **server DNS** predefinito e secondario sulla rete;
- La lunghezza della maschera di sottorete sulla rete locale (che chiaramente all'host è ignota).