Appunti di Reti di Calcolatori

Simone Ianniciello

A.A. 2020/2021

Contents

Chapter 1

Introduzione alle Reti

1.1 Introduzione

Rete Con rete si intende un'interconnessione di dispositivi in grado di scambiarsi informazioni. Le reti sono commposte da elementi quali:

- Sistemi terminali *(Host)*

  Macchine degli utenti finali

  Server *Fornitori di servizi*

- Switch: Dispositivi adibiti all'interconnessione locale di Host
- Router: Dispositivi di interconnessione di reti diverse
- Collegamenti: I mezzi tramite i quali vengono trasferite le informazioni

  Cavi in rame

  Fibra ottica

  Onde radio *(WiFi)*

1.2 Tipi di rete

LAN Con *LocalAreaNetwork* o *Rete Locale* si intende un insieme di Host appartenenti allo stesso ente *(Organizzazione, Casa, Scuola)* in grado di comunicare.
Le LAN possono essere:

- a cavo condiviso: Tutti gli host condividono lo stesso cavo per comunicare. Questo sistema non e' piu' usato perche' poco efficente.
- con switch: Tutti gli host sono collegati a uno switch che instrada le informazioni nella direzione desiderata. Questo sistema e' molto piu' efficiente perche' le macchine non hanno bisogno di monopolizzare la rete.

WAN Per *WideAreaNetwork* o *Rete Geografica* si intende una rete formata da piu' LAN e/o singoli host separati da grandi distanze. Essa viene gestita da da un operatore che fornisce il servizio di interconnessione ai clienti.
Le WAN si distinguono in:

- WAN punto-punto
- WAN a commutazione

Una applicazione tipica sono reti locali appartenenti ad un'azienda interconnesse tramite WAN p-p

1.3 Tecniche di commutazione

I due sistemi principali per determinare il percorso tra due host e dedicargli le risorse sono:

- Circuit-switched network *(Commutazione di circuto)*
- Packet-switched network *(Commutazione di pacchetto)*

Commutazione di circuito Per la commutazione di circuito si procede instaurando un cammino dedicato tra i due host: vengono assegnate le risorse necessarie alla comunicazione e sono garantite per l'intera durata della connessione. Cio' significa che una volta instaurata la connessione, essa non verra' disturbata in alcun modo.
I principali problemi di questa tecnica sono pero' il tempo di instaurazione della connessione *(risorse non disponibili)*, e il non sfruttamento delle risiorse disponibili durante i *silenzi* nella comunicazione.

Commutazione di pacchetto Nelle connessioni a commutazione di pacchetto il flusso di dati viene diviso in pacchetti ed essi vengono *spediti sulla rete* sul percorso prescelto. Le risorse vengono quindi utilizzate solo se necessarie e possono essere condivise da pacchetti provenienti da host differenti.

Ogni nodo della rete si occupa di ricevere e riservire i pacchetti che gli arrivano. Per fare cio' il commutatore, dopo aver ricevuto un pacchetto, lo mette in una coda di tipo FIFO; quando e' pronto a ritrasmettere preleva il primo pacchetto dalla coda. Cio' porta a dei ritardi (Il commutatore deve ricevere l'intero pacchetto per reinviarlo, i pacchetti potrebbero dover *aspettare* in coda) e a delle perdite di pacchetti (coda piena).
Questo metodo si chiama **Store and Forward**.

1.4 Internet

Con **i**nternet si intende un sistema formato da due o piu' reti comunicanti. L'**I**nternet e' l'insieme di reti piu' comune. Ogni rete che intende aggiungersi ad essa deve seguire L'Internet Protocol (IP) e rispettare certe convenzioni.

L'infrastruttura di Internet fornisce servizi di comunicazione alle applicazioni

- Senza connesione **(UDP)**
- Orientati alla connessione **(TCP)**

Sono stati definiti dei **protocolli** di comunicazione per le applicazioni piu' comuni di Internet *(TCP, IP, HTTP, FTP...)*

Ci sono delle organizzazioni adibite alla definizione degli standard di internet:

- **IETF** Internet Engineering Task Force
  - Studia e sviluppa i protocolli in uso su internet.
  - Pubblica i documenti ufficiali che li descrivono sotto forma di RFC/STD *(Request For Comments, STanDards)*
- **ICANN** Internet Corporation for Assigned Names and Numbers
  - Coordina i DNS
  - Assegna i gruppi di indirizzi di rete
  - Ha funzioni di controllo semplice dello sviluppo di Internet
- **W3C** World Wide Web Consortium
  - Sviluppa di standard aperti *(HTML, XML...)*

1.4.1 Strati della rete

Le reti degli host si collegano a Internet tramite gli ISPs *Internet Service Provider*. I livelli della rete sono:

Livello 3 ISP di accesso: Sono quelli a cui si connettono comunemente le reti locali.

Livello 2 ISP regionali: Sono dei collegamenti intermedi che uniscono tutti gli ISP di livello 2 in una zona geografica

Livello 1 Dorsali: Esse sono la parte piu' *alta* di Internet, tutti gli altri ISP si connettono ad esse. *(ne esistono circa 11)*

1.4.2 Peering point

Sono accordi tra due ISP che gli permettono di ricevere e riinoltrare il traffico da uno all'altro: per fare cio' esistono gli IXP *(Internet eXchange Point)* ovvero sistemi, anche gestiti da aziende di terzi, che effettuano il peering.

1.4.3 Reti di acesso

Il collegamento tra l'utente e Internet e' detto **rete di accesso**.

- Accesso via rete telefonica
  - dial-up
  - Digital Subscriber Line **(DSL)**
  - Fibra ottica
- Accesso tramite reti wireless
  - 3G, 4G, 5G
- Collegamento diretto
  - Collegamenti WAN dedicati per aziende, universita'...

1.5 Metriche di riferimento

Larghezza di banda (Bandwidth) Larghezza in Hertz dell'intervallo di frequenze utilizzato per la trasmissione.

Velocita' di trasmissione (bitrate) Quantita' di dati trasmissibili nell'unita' di tempo (bps).

Troughput Quantita' di dati trasmissibili da un nodo A ad un nodo B in una unita' di tempo. Tiene di conto anche di perdite sulla rete, protocolli, ecc...

Latenza Tempo che intercorre tra l'invio e la ricezione del primo bit di un messaggio.

$$\text{Latenza} = \text{elaborazione} + \text{accodamento} + \text{trasmissione} + \text{propagazione}$$

1.5.1 Ritardi

Ritardo di elaborazione Causato dal sistema di controllo degli errori e dal sistema che determina il canale di uscita.

Ritardo di accodamento Tempo tra l'inserimento nella coda di trasmissione e la ritrasmissione del pacchetto.

Ritardo di trasmissione Tempo impiegato per trasmettere un pacchetto sul mezzo di trasmissione.

Misurato in PacketLenght / BitRate

Ritardo di propagazione Tempo che inpiega un bit ad essere propagato da un nodo all'altro.

Misurato in LinkLenght / PropSpeed $(3 - 2 * 10^{-}8)$

Ritardo end-to-end Ritardo cumulato tra tutti i nodi in una trasmissione. E' pari alla sommatoria dei ritardi tra i vari nodi del collegamento.

1.5.2 Prodotto rate-ritardo

Numero massimo di bit che possono *essere contenuti* nel mezzo trasmissivo.

1.6 Modelli stratificati

Il modello statificato permette di scomporre un sistema complesso in piu' sistemi piu' facili da implementare e comprendere. La modularizzazione dei livelli permette di dividere l'interfaccia dall'implementazione di un servizio. Percio' dall'esterno ogni modulo e' visto come un'interfaccia che: accetta determinati parametri in ingresso, esegue le sue mansioni, e ritorna una risposta. Quindi l'implementazione puo' essere cambiata senza che il resto del sistema *ne venga a conoscenza*

1.7 OSI RM **(Open System Interconnection Reference Model)**

Nel 1976 sono iniziati i lavori per definire uno standard aperto per i protocolli di Internet. La ISO ha da prima pubblicato questi standard sotto forma del OSI-RM, che poi e' diventato uno standar internazionale nel 1983 (ISO 7498).

Il modello ISO/OSI prevede la statificazione del protocollo di telecomunicazione.

Gli strati OSI sono:

7 Applicazione - elaborazione dati

6 Presentazione - unificazione dati

5 Sessione - controllo del dialogo

4 Trasporto - trasferimento dati tra hosts

3 Rete - instradamento del traffico

2 Datalink - consegne trame sul link

1 Fisico - trasmette un flusso di bit

Le informazioni si propagano dal livello piu' alto (applicazione) fino al piu' basso per poi passare sul mezzo trasmissivo e risalire i livelli fino alla destinazione.

Ogni livello aggiunge all'informazione del livello superione una propria sezione informativa (header / trailer) Questo processo di incapsulamento delle informazioni e' reversibile percio' ogni livello e' in grado di estrarre i dati degli strati superiori.

1.8 Stack Protocollare TCP-IP

E' la famiglia di protocolli attualmente utilizzata in Internet. E' definita attualmente da cinque livelli:

Applicazione Applicazioni di rete, collegamento logico end-to-end, scambio di messaggi tra processi. *ftp, smtp, http*

Trasporto Trasferimento dati end-to-end. *tcp, udp*

rete Instradamento dei datagrammi. *IP, ICMP*

Link Trasferimento dei dati in frame tra elementi vicini. *ppp, ethernet, ...*

Fisico Trasferimento di bit di un frame sul mezzo trasmissivo.

Chapter 2

Lo Strato Applicativo: URI-HTTP

2.1 Applicazioni di rete

Sono applicazioni formate da processi distibuiti comunicanti; Cio' significa che i processi possono essere eseguiti su host diversi sulla rete. Sullo stesso host piu' processi possono comunicare anche attraverso la comunicazione inter-processo definita dal sistema operativo (quindi esterna alla rete). Due processi su macchine diverse comunicano tramite la rete con dei **messaggi**. Grazie alla statificazione dei livelli, i livelli applicazione comunicano tra loro come se esistesse un collegamento diretto tra i due.

2.2 Protocollo dello Strato di Applicazione

Esso definisce:

- i tipi di messaggi scambiati
- la sintassi dei messaggi *(i campi)*
- la semantica dei campi *(il significato)*
- le regole per l'invio dei messaggi *(quando e come)*

2.3 Paradigmi

Client-Server I server *(pochi ma potenti)* offrono un servizio e sono sempre in attesa di richieste.

Peer-tp-Peer (p2p) I peer offrono e richiedono servizi contemporaneamente.

Misto Programmi che utilizzano entrambi i paradigmi per utilizzi diversi *(Skype, . . .)*

2.4 Application Programming Interface (API)

E' un insieme di regole da rispettare per utilizzare le risorse di un servizio. Per esempio, se un processo vuole inviare un messaggio sulla rete, utilizza delle chiamate al sistema operativo che comunica con gli strati inferiori dello stack TCP-IP attraverso l'Interfaccia Socket.

L'Interfaccia Socket E' un'API che collega gli strati applicazione e trasporto. Grazie ad essa un programmatore che intende sviluppare un'applicazione con accesso a Internet non deve preoccuparsi di tutte le *formalita'* per la connessione. Per identificare un processo sulla rete si utilizza una coppia:

<Indirizzo IP (32b) + numero di porta (16b)>

2.5 Uso dei servizi di trasporto

Nel livello di trasporto sono previsti dule protocolli principali:

TCP connection-oriented: client e server devono *salutarsi* prima di iniziare a comunicare; c'e' controllo degli errori sui messaggi.

UDP connection-less: **CHAOS!** Non c'e' nessun controllo, il mittente tira i messaggi sulla rete e suppone che il destinatario lo riceva.

UDP e' utile se si ha bisogno di un alto troughput e/o un basso ritardo, ma solo se non ci interessa un trasferimento affidabile al 100% dei dati *(streaming, videogiochi, . . .).*

2.6 Web e HTTP

2.6.1 Il Web

Una pagina Web consiste di:

- Un file HTML
- Diversi oggetti referenziati indirizzabili tramite una URL (Uniform Reference Locator)

2.6.2 Uniform Resource Identifier

E' una forma generale per identificare una risorsa sulla rete. Ci sono due tipi di URI:

Uniform Resource Locator (URL) Risorse identificate tramite il meccanismo di accesso *(HTTP, FTP, ...)*

Uniform Resource Name (URN) Identificatore globalmente unico, anche se la risorsa diventa non disponibile.

2.6.3 Sintassi URL

scheme://host:port/path

- scheme: protocollo di accesso
- host: nome di dominio o indirizzo IP
- port: numero di porta del servizio richiesto
- path: identifica la risorsa nel contesto sopra-descritto

Nelle URL HTTP si aggiunge anche un parametro *?query* dopo *path*. Questa stringa viene interpretata dal server e serve a passare informazioni aggiuntive nella richiesta.

2.6.4 URI Assolute e Relative

URI assoluta identifica una risorsa indipendendentemente dal contesto.

URI relativa identifica una risorsa in relazione ad un'altra URL e viene interpretata dal client prima di mandare la richiesta.

2.7 HyperText Transfer Protocol (HTTP)

Protocollo stateless pubblicato nel 1990 e utilizzato fino ad ora nel World Wide Web. Una caratteristica fondamentale dell'HTTP e' la tipizzazione e negoziazione della rappresentazione dei dati percio' non e' limitato all'ipertesto.

Il client stabilisce una connessione e invia richieste (request) tramite HTTP al server, il quale invia messaggi di risposta (response) Usa TCP dato che il trasferimento deve essere senza perdite e che i ritardi non sono *un problema*

Dalla versione 1.1 la connessione client-server, a meno che diversamente espresso dal client, e' persistente; percio' il client puo' assumere che il server manterra' la connessione aperta. Cio' porta a minor utilizzo di CPU e ritardi minori dato che non c'e' bisogno di riaprire una nuova connessione per ogni richiesta. Inoltre si ha un mignor controllo di congestione.

La connessione viene chiusa dal server solo quando esplicitamente richiesto dal client nell'header del messaggio, o se non riceve piu' richieste (time out)

2.7.1 Pipelining e HeadOfLineBlocking

Per migliorare le le prestazioni il client puo' inviare contemporaneamente piu' richieste al server, il quale **DEVE** rispondere nello stesso ordine in cui sono state ricevute.

Se pero' una richiesta richiede piu' tempo al server per essere processata blocca tutte le risposte successive e porta all'HOLB. In questi casi non si ha un miglioramento delle performance.

2.7.2 HTTP Request

La richesta e' formata da:

Request-Line
*(general-header
| request-header
| entity-header)
CRLF
[message-body]

2.7.3 HTTP Response

La risposta e' formata da:

Status-Line
*(general-header
| response-header
| entity-header)
CRLF
[message-body]

Request-Line

Method SP
Request-URI SP
HTTP-Version CRLF

Status-Line

HTTP-Version SP
Status-Code SP
Reason-Phrase CRLF

General-header Relativi alla connessione.

Entity-header Relativi all'entita' trasmessa.

Request-header Relativi alla richiesta.

Response-header Nel messaggio di risposta.

2.7.4 Content Negotiation

Le risorse possono essere disponibili in piu' rappresentazioni; Il client puo' richiederne una in particolare tramite richieste su Request e Entity headers.

2.7.5 Request methods

OPTIONS Il server ritorna solo le opzioni di comunicazione associate ad una URL (capacita', metodi esposti, ...).

GET Richiede il trasferimento della risorsa indicata. Sono possibili i **conditional-get** (If-Mofidied-Since, If-Match, ...).

HEAD Simile alla GET ma il server non trasferisce il message-body. Utile per controllare lo stato dei documenti (validita', modifiche, ...).

POST Serve per inviare al server informazioni inserite nel body del messaggio.

PUT Chiede al server di creare/modificare una risorsa (Recuperabile poi tramite GET).

DELETE Il client chiede di eliminare una risorsa identidficata dalla Request-URI.

2.7.6 Caching

E' possibile memorizzare copie temporanee di risorse Web e ri-servirle al client senza contattare il server. Si possono avere:

User-Agent Cache Risorsa mantenuta dal browser.

Proxy Cache Il proxy mantiene una copia delle risorse richieste. Quando un browser richiede una risorsa cached essa gli viene riservita dal proxy senza interrogare nuovamente il server.

2.7.7 Cookies

Essendo stateless, non e' mantenere una memora su un'applicazione web (come ad esempio l'identita' di un client). Una soluzione e' utilizzare i cookies, cioe' id unici che il client presenta al server ogni volta che esso effettua una richiesta.

Alla prima connessione, il server invia la normale risposta HTTP + una line **Set-cookie: id**. Il client si salva il cookie e lo associa al server. Ogni successiva richiesta a quel server conterra' quel cookie.

Chapter 3

Lo Strato Applicativo: Telnet, EMAIL-SMTP

3.1 Telnet

TErminaL NETwork e' un protocollo che permette l'uso di shell su macchine remote. Permette al client di effettuare una sessione di login al server, dopo la quale esso puo' accedere a comandi e programmi disponibili sulla macchina remota.

Dopo il login, vengono passate le battute dei tasti al server come standard-input e l'output viene riportato direttamente al client.

Il protocollo TELNET [RFC854] detta che:

- Il client stabilisce una connessione di tipo TCP (tipicamente sulla porta 23) con il server. la connessione persiste per tutta la durata della sessione di login.
- Vengono *"collegate"* le STDIN e STDOUT dei tue terminali.

Il server Telnet utilizza uno Pseudo Terminal Driver per eseguire i comandi.

3.1.1 Network Virtual Terminal: NVT

Telnet deve poter operare con il numero massimo di sistemi quindi deve poter lavorare con client su sistemi operativi diversi.

Per fare questo il client e il server passano attraverso un terminale virtuale in modo da avere una sola codifica ed entrambi effettuano una conversione dalla propria a quella del NVT. Sulla rete vengono quindi trasferiti i comandi con l'**NVT Character Set**.

3.1.2 NVT Character Set

I dati vengono trasferiti tramite 7-bit US-ASCII.

- Ogni carattere e' inviato come un byte con il primo bit settato a 0.
- Per le sequenze di comandi si imposta il bit piu' significativo a 1.
- I comandi come EOL iniziano con 0xFF (Interpret As Command **IET**)

3.2 E-Mail

Permette il trasferimento di messaggi tra un mittente e un destinatario. Dato che il destinatario potrebbe non avere il client mail aperto in quel momento, il servizio di posta elettronica deve basarsi su componenti intermediari per mantenere il messaggio finché il destinatario non e' in grado di riceverlo.

3.2.1 Principio di funzionamento

- L'utente invia un messaggio.
- Il sistema ne mantiene una copia nell'area di accodamento *(spool)*, insieme a id mittente, id destinatario, id destinazione e tempo di deposito.
- Il client avvia il trasferimento alla macchina remota stabilendo una connessione TCP.
- Se la connessione viene aperta inizia il trasferimento del messaggio e, a trasferimento completato, cancella la copia locale.
- Altrimenti il processo viene ripetuto periodicamente scandendo i messaggi nella spool.
- Oltre un certo intervallo di tempo, se il messaggio non e' ancora stato consegnato, viene inviata una notifica al mittente.

3.2.2 Indirizzi email

Formato da

local-part @ domain-name

local-part Specifica la cassetta a cui consegnare il messaggio.

domain-name Specifica il mail server.

3.2.3 Simple Mail Transfer Protocol (SMTP)

L'obbiettivo di SMTP e' il trasferimento affidabile ed efficiente di mail. Il protocollo e' indipendente dal sistema di trasmissione usato. Una caratteristica e' la capacita' di trasportare mail attraverso piu' reti; Un messaggio puo' quindi passare da server intermedi prima di arrivare al destinatario.

Modello: Un client apre un canale bidirezionale con il server SMTP. Inizia poi il trasferimento del messaggio (se non va a buon fine, comunica l'insuccesso). Per trovare il server SMTP, il client risolve il dominio attraverso un DNS.

Protocollo: Solitamente SMTP si basa sulla porta 25 TCP. Le fasi del trasferimento sono:

- handshaking
- trasferimento del messaggio
- chiusura della connessione

L'interazione e' di tipo **comando/risposta**

Handshaking Il client stabilisce la connessione e aspetta la risposta 220 READY FOR MAIL. Il client risponde con HELO ed il server risponde identificandosi.

Comandi SMTP:

- HELO *client-id*
- MAIL FROM: *reverse-path* [SP *mail-parameters*] <CRLF>
- RCPT TO: *forward-path* [SP *rcpt-parameters*] <CRLF>
- DATA
- QUIT

<CRLF>.<CRLF> determina la fine di un messaggio.

Formato dei messaggi

- Linee di intestazione (header)
  - To:
  - From:
  - Subject:
  - ...
- body
  - Soli caratteri ASCII a 7 bit.

Per ovviare al body solo testuale e' stato creato lo standard MIME (Multipurpose Internet Mail Extension). Questo standard *vive sopra* alla RFC 822 originale ma da nuove regole di interpretazione del body. Cio' ha permesso di mantenere i mail-server esistenti e cambiare solo gli user-agents.

Linee di intestazione aggiuntive per il MIME:

- MIME-Version:
- Content-Transfer-Encoding:
- Content-Type:

Content-Type: type/subtype;

- text
  - plain
  - html
- image
  - jpeg
  - gif
- audio
  - basics
  - 32kadpcm
- video
  - mpeg
- application
  - msword
  - ocet-stream

Esiste anche il tipo multipart: esso permette di inviare messaggi con piu' tipi separati da un boundary=

3.2.4 Sistemi pull per le mail POP3, IMAP e HTML

Post Office Protocol Lo user-agent apre una connessione TCP (porta 110) verso il server di posta. Comandi permessi:

- Autorizzazione
  - user:
  - pass:
- Transazione
  - list:
  - retr:
  - dele:
  - quit:
- Aggiornamento
  - Dopo `quit` il server cancella i messaggi marcati per la rimozione.

Internet Mail Access Protocol E' piu' complesso di POP3 ma permette la manipolazione dei messaggi memorizzati e da la possibilita' di estrarre solo alcuni componenti dei messaggi.

HTTP Hotmail, Gmail, ...

Chapter 4

Lo Strato Applicativo: DNS

4.1 Introduzione

Ogni rete e' identificata sulla rete tramite un indirizzo IP. Il problema di essi e' che non sono facili da ricordare *e sono brutti*. E' quindi nato il bisogno di associare gli IP a dei nomi logici (e spesso mnemonici). Per associare gli IP ai nomi e' stato creato il DNS.

Il DNS adotta il paradigma client-server. Si affida al protocollo di trasporto sottostante per trasferire i messaggi. E' costituito da:

- Uno schema di assegnazione dei nomi
- Un database distribuito contenente le associazioni

$$nomedominio \implies IP$$

- Un protocollo per la distribuzione delle informazioni sui nomi tra i name server
  - Utilizza UDP (porta 53) [oppure TCP]

4.2 Servizi DNS

Risoluzione Traduzione $hostname \implies indirizzoIP$

Host aliasing Traduzione $nomi \implies nomecanonico/IP$

Mail server aliasing Stessa cosa degli host, permette tra le altre cose di usare nomi identici per mail e web server.

Distribuzione di carico Ad un hostname possono corrispondere piu' indirizzi IP; il DNS restituisce la lista di IP variandone l'ordinamento ogni volta.

4.3 Spazio dei nomi

Struttura "flat" Sequenza di caratteri senza alcuna ulteriore struttura (deprecato)

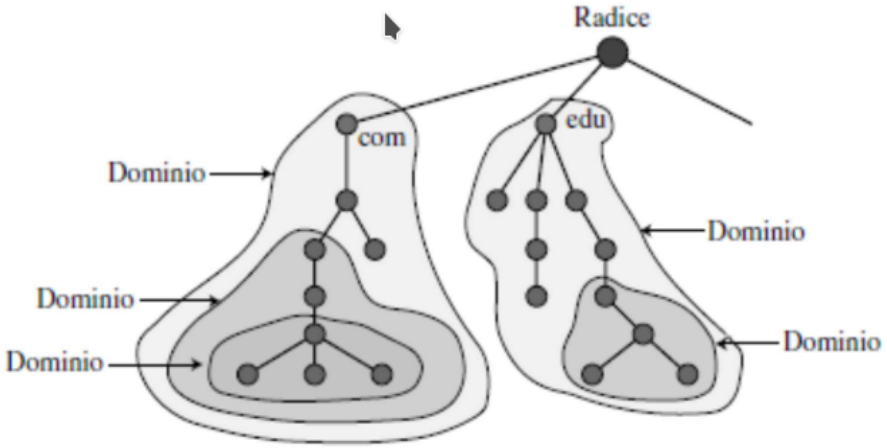
Struttura gerarchica Il nome e' costituito da diverse parti

(p.e. studenti.di.unipi.it)

L'assegnazione dei nomi e' delegabile al sistema decentralizzato.

4.3.1 Nomi di dominio

I nomi hanno una struttura ad albero. Ogni nodo e' individuato da un'etichetta.(la radice ha etichetta vuota)



La struttura gerarchica permette autonomia nella scelta dei nomi all'interno di un dominio.

4.3.2 Esempi di Top Level Domain (TLD)

com Organizzazioni commerciali

edu Istituti di istruzione

mil Gruppi militari

gov Istituzioni governative

net Centri di supporto alla rete

org Organizzazioni diverse dalle precedenti

it, uk, fr, us Schema geografico per nazioni

Mantenuti da IANA (Internet Assigned Numbers Authority)

4.4 Gerarchia dei Name Server

DNS Database distribuito implementato come gerarchia di piu' Name Server.

Name Server Gestisce le richieste. Ogni server immagazzina le informazioni relative alla propria zona inclusi i riferimenti ai server dei domini di livello inferiore.

4.4.1 Root Name Server

Restituisce le informazioni sui name server dei TLD. Ce ne sono centinaia in tutto il mondo.

4.4.2 TLD Server

Mantiene le informazioni dei nomi di dominio che appartengono a un TLD; Restituisce le informazioni sui Name Server di competenza dei sotto domini.

4.4.3 Authoritative Name Server

E' l'autorità per una certa zona. Puo' effettuare traduzioni nome $\Longrightarrow$ indirizzo o inoltrare la richiesta a altri Name Servers. Sono di due tipi:

Server primari Mantengono il file di zona.

Server secondari Ricevono il file di zona e offrono il servizio di traduzione.

4.4.4 Local Name Server

E' un Name Server mantenuto dagli ISP. Le query DNS vengono prima rivolte al server locale il quale, in caso non riesca a risolverle, le inoltra alla gerarchia DNS.

4.4.5 Query DNS

La query puo' essere:

Ricorsiva La richiesta fa il *giro completo* attraversando ogni name server fino a trovare il dominio che sta' cercando.

Iterativa Le risposte sono restituite direttamente al client insieme al riferimento al server da contattare.

4.4.6 DNS Caching

I name server mantengono una memoria delle associazioni gia' richieste, le quali vengono cancellate dopo un tempo di timeout.

4.5 Resource Records

RR format: **(name, value, type, ttl)**

TTL Tempo di vita della risorsa della cache.

Type A

Name hostname
Value IP

Type CNAME

Name hostname
Value nome canonico

Type NS

Name Nome di dominio
Value hostname dell'authoritative name server per quel dominio

Type MX

Name Nome di dominio
Value Nome canonico del server di posta elettronica associato

Chapter 5

Lo Strato Trasporto: Introduzione

5.1 Obbiettivo

Realizza una comunicazione logica tra processi su terminali diversi. Cio' significa che essi comunicano come se ci fosse un collegamento diretto tra i due. Lo strato applicazione trasmette dati allo strato di trasporto, il quale poi comunica con gli strati inferiori per trasmetterli all'host di destinazione.

5.2 Servizi offerti

Lo strato di trasporto si occupa principalmente di multiplexing/demultiplexing e del controllo degli errori.

5.2.1 Protocolli

In base al protocollo utilizzato per la connessione, lo strato di trasporto si occupa anche di altri fattori. I due protocolli (principali?) sono:

TCP (RFC 793) Si occupa della gestione della connessione, dei controlli di flusso e congestione, e assicura una consegna affidabile.

UDP (RFC 768) Protocollo connection-less, non affidabile ma piu' veloce.

5.2.2 Multiplexing/Demultiplexing

Multiplexing Lo strato di trasporto si occupa di *accorpare* i flussi di dati e spedirli verso la loro destinazione insieme ad una *busta* di trasporto.

Demultiplexing Lo strato di trasporto si occupa di ricevere i dati dalla rete e instradarli verso i processi desiderati.

Entrambi si basano sul socket addres (IP + porta) per identificare i processi. In base al protocollo usato, lo strato di trasporto si occupa anche del

Demultiplexing connectionless - UDP La *busta* contiene solamente la socket address del destinatario. I datagrammi provenienti da host differenti vengono consegnati tutti alla stessa socket di destinazione.

Demultiplexing orientato alla connessione - TCP La *busta* contiene le socket address di mittente e destinatario. Un host puo' supportare piu' socket contemporaneamente (p.e. server Web). Lo strato di trasporto puo' quindi inviare i dati alla socket appropriata in base al mittente.

5.2.3 Le porte

Ogni processo che intende comunicare con la rete (punto di demultiplexing) e' identificato tramite un socket address. La porta e' un valore u-int16(0-65535). I range standard sono:

System ports (0-1023) Assegnate da IANA, identificano i processi server.

User ports (1024-49151) Assegnate da IANA, non e' possibile la duplicazione.

Dynamic ports (49152-65535), Non assegnate da IANA, muoiono al momento della chiusura della connessione.

5.2.4 Well known ports

20/tcp	ftp-data
21/tcp	ftp
22/tcp	ssh
23/tcp	telnet
25/tcp	smtp
53/udp	dns
53/tcp	dns
69/udp	tftp
80/tcp	www-http
110/tcp	pop3
119/tcp	nntp
161/udp	sntp
220/tcp	imap3
510/udp	RIP
3306/tcp	mysql

Chapter 6

Lo Strato di Trasporto: TCP

6.1 Proprietà del servizio

I processi effettuano un handshake COVID!1!. Lo strato della connessione risiede sui puni terminali e non sui nodi della rete.

Permette di trasferire un flusso continuo di dati in formato bidirezionale (full-duplex). Inoltre consente di assegnare una connessione diretta processo - processo. Offre anche controllo di connessione tramite meccanismi di inizio e fine trasmissione. Tramite alcuni bit di controllo, permette di correggere alcuni tipi di errore. Con il controllo di flusso si evita di spedire piu' dati di quanti il destinatario sia in grado di trattare. Nel caso di sovraccarico della rete, tramite il controllo di congestione, si hanno sistemi di recupero della connessione.

Trasferimento bufferizzato Il protocollo TCP puo' suddividere il flusso di byte in segmenti. Per farlo e' necessario un buffer dove immagazzinare i dati finché non ce n'e' un numero sufficiente da essere spediti. Cio' consente un minor numero di messaggi scambiati per trasferire una sequenza di byte.

6.2 Formato dei segmenti

6.2.1 Numeri di sequenza e di riscontro

Numero di sequenza e' il numero del primo byte di un segmento. Generalmente si parte da un Initial Sequence Number (ISN) casuale.

Numero di riscontro e' il +1 del numero dell'ultimo byte ricevuto correttamente.

$ACK{=}x$ = significa: ho ricevuto tutti i byte fino a *x-1*, aspetto *x*.

6.2.2 Forma dei segmenti

L'header aggiunto al messaggio contiene:

Porta sorgente 16bit

Porta destinazione 16bit

n-SEQ Numero di sequenza, 32 bit

n-ACK Numero di riscontro, 32 bit

HLEN Lunghezza dell'header espressa in parole da 4Byte, 4bit

Reserved 6bit

URG Il campo puntatore contiene i dati da trasferire in via prioritaria, 1bit

ACK Considerare n-ACK, 1bit

PSH Trasferimento immediato da trasporto a applicazione

RST Reset della connessione

SYN Sincronizza n-SEQ

FIN Chiusura della connessione

Dimensione finestra Numero di byte a partire da n-ACK, elaborabili, 16bit

Checksum Considera l'intero pacchetto; serve a rilevare gli errori, 16bit

Puntatore urgente Punta al primo Byte non URG partendo da n-SEQ, 16bit

Opzioni e riempimento Negoziazione di vari parametri opzionali, 0-40Byte

6.3 Gestione della connessione

6.3.1 Handshake a tre vie

Richiesta di connessione Il client invia una richiesta al server con:

SYN 1
n-SEQ #rand $_{= x}$

Il server inizializza due buffer e le variabili di connessione per il controllo di flusso e congestione.

Autorizzazzione connessione (SYNACK) Il server risponde con:

SYN 1
ACK 1
n-SEQ #rand $_{= y}$
n-ACK x+1

Il client inizializza gli stessi buffer e variabili.

ACK Riscontro positivo dell'avvenuta connessione, contiene:

SYN 0
ACK 1
n-SEQ x+1
n-ACK y+1

Connessione stabilita.

6.3.2 Chiusura della connessione

C $\Longrightarrow$ S Chiusura da parte del client:

FIN 1
n-SEQ x

Il client non puo' piu' inviare dati.

S $\Longrightarrow$ C ACK di chiusura:

ACK 1
n-ACK x+1

Il server puo' ancora inviare dati, il client no.

S $\Longrightarrow$ C Chiusura da parte del server:

FIN 1
n-SEQ y

Il server aspetta la ricevuta di chiusura.

C $\Longrightarrow$ S ACK di chiusura.

ACK 1
n-ACK y+1

Connessione terminata.

Quando il client riceve FIN dal server entra nello stato TIME_WAIT e ci resta per un tempo dettato da MSL (diverso su macchine differenti) prima di poter chiudere totalmente la connessione. Questo serve nel caso l'ultimo ACK venga perso perché a un certo punto uno dei due host richiederà la chiusura passiva e l'altro dovrà rispondere con un ACK. Allo stesso modo dell'handshake di apertura, si puo' avere un three-way handshake di chiusura.

6.4 Stati TCP

6.4.1 Stati client

SYN-SENT Dopo aver inviato una richiesta di connessione, si aspetta la conferma.

ESTABLISHED La connessione e' pienamente stabilita, e' possibile trasferire dati.

FIN-WAIT-1 Il client aspetta una richiesta di chiusura o l'ack di una sua richiesta di terminazione.

FIN-WAIT-2 Aspetta la richiesta di terminazione di connessione da un host remoto.

TIME-WAIT Vedi sopra.

CLOSED Connessione chiusa. Non e' piu' possibile comunicare.

6.4.2 Stati server

LISTEN In attesa di una connessione TCP qualunque.

SYN-RECEIVED Si entra in questo stato appena si riceve una richiesta di connessione.

ESTABLISHED Vedi sopra.

CLOSE-WAIT Si aspetta la richiesta di chiusura dal client.

LAST-ACK Si aspetta l'ack di chiusura.

CLOSED Vedi sopra.

6.5 Trasferimento di dati affidabile

6.5.1 Esempio Telnet su TCP

C $\Longrightarrow$ **S** L'utente digita C

n-SEQ 42
n-ACK 79
data "C"

S $\Longrightarrow$ **C** ACK per ricevuta di C

n-SEQ 79
n-ACK 43
data "C"

S $\Longrightarrow$ **C** ACK dell'ACK

n-SEQ 43
n-ACK 80

Il mittente puo' anche inviare piu' segmenti senza attendere il riscontro (pipelining).

6.5.2 Eventi lato mittente

Lo strato di trasporto riceve i dati dall'applicazione, li incapsula, gia assegna un numero di sequenza, ed avvia un timer di ritrasmissione (RTO).

La ritrasmissione avviene in caso di:

Timeout Non si riceve un ACK prima della scadenza del RTO.

ACK duplicato Il mittente riceve tre ACK uguali significa che il segmento successivo a quello e' andato perso. Si ha quindi la fast retransmission.

6.5.3 Segmenti fuori sequenza

Se i dati arrivano fuori sequenza (ordinamento sparso), l'entità TCP destinataria puo' memorizzarli temporaneamente ma il protocollo non specifica che utilizzo farne.

Nelle versioni piu' recenti si implementa SACK, che manda l'ACK dei pacchetti fuori sequenza in OPTIONS.

6.5.4 Eventi lato destinatario

Tutti i segmenti inviati per trasmettere dati includono ACK.

Delayed ACK Se il destinatario riceve un segmento atteso, puo' ritardare l'invio di ACK di 500ms a meno che non riceva un altro segmento.

Fast retrasmission request Se il destinatario riceve un segmento fuori sequenza, un duplicato, o un capisce che c'e' un segmento mancante, risponde subito con un ACK indicando il segmento da cui ripartire.

6.5.5 Calcolo del timeout

Il RTO deve essere maggiore del Round Trip Time (RTT) (tempo tra l'invio di un messaggio e la ricevuta dell'ACK).

$$eRTT = (1 - \alpha) * eRTT + \alpha * sRTT$$

eRTT RTT stimato, cumulativo su piu' misure.

sRTT RTT di un segmento trasmesso con successo.

alpha 1/8 (RFC2988)

$$dRTT = (1 - \beta) * dRTT + \beta * (sRTT - eRTT)$$

dRTT Deviazione di eRTT da sRTT

beta 1/4 (RFC2988)

$$\boldsymbol{RTO} = eRTT + 4 * dRTT$$

In molte implementazioni dopo un errore si raddoppia RTO

6.5.6 Finestra di trasmissione

Il buffer di invio, si dividono i byte in finestre. La finestra di trasmissione e' la sottosezione del buffer di invio che deve essere spedita con il prossimo messaggio. Ha dimensione variabile (modificata in base alla condizione della rete) per permettere i controlli di flusso e congestione. Viene fatta avanzare non appena si riceve l'ACK dell'ultima trasmissione.

6.6 Controllo di flusso

Il destinatario di un messaggio mantiene un buffer di ricezione. il processo sullo strato applicazione legge i dati da una finestra di ricezione (non necessariamente quando arrivano). Si intende con controllo di flusso la capacità del mittente di evitare di saturare il buffer del destinatario. Il mittente mantiene una variabile detta finestra di ricezione (rwnd) che detta lo spazio disponibile nel buffer ricevente; essa e' passata dal destinatario al mittente tramite il campo window nell'header TCP.

$$rwnd = RcvBuffer - (LastByteReceived - LastByteRead)$$

Il mittente si assicura che:

$$LastByteSent - LastByteAcked < rwnd$$

Se $rwnd = 0$ il mittente lo *richiede* con un segmento sonda da un byte

6.7 Controllo di congestione

Se si provano a spedire troppi dati su una rete che non puo' supportarli avviene il fenomeno della congestione; cio' puo' provocare:

- Lunghi ritardi (accodamenti nei buffer)
- Perdita di pacchetti (overflow dei buffer)

TCP permette il controllo di congestione: se si percepisce uno scarso traffico, viene aumentata la frequenza di invio; altrimenti diminuisce.

6.7.1 Algoritmo di controllo

L'algoritmo utilizzato per regolare la frequenza di invio in funzione della congestione e' costituito da tre fasi:

- Slow start
- AIMD
- Fast recovery

6.7.2 Finestra di congestione (Congestion Window)

Misurata in Max Segment Size (MSS); E' il numero di segmenti che si possono inviare insieme, senza bisogno di aspettarne l'ACK.

6.7.3 Slow start

All'inizio la congestion window (CW) e' posta a 1MSS. Per ogni ACK, CW viene aumentata di 1MSS. Cio' significa che essa raddoppia per ogni RTT.

6.7.4 Additive Increase Multiplicative Decrease

La CW viene aumentata in modo da avere un incremento pari a 1MSS per ogni RTT; Cio' significa che per ogni ACK, cwnd = cwnd+1/cwnd.

Ad ogni evento di perdita di ACK la CWND viene dimezzata.

6.7.5 TCP RENO

Viene definita una variabile di threshold alla quale viene assegnato un valore alto; Finché $cwnd < threshold$, siamo in slow start (cwnd aumenta esponenzialmente).
Non appena $cwnd > theshold$, si entra in AI.