Sicurezza dei sistemi e delle reti[1]

Mattia Monga

Dip. di Informatica
Università degli Studi di Milano, Italia
mattia.monga@unimi.it

a.a. 2014/15

[1]

Lezione IV: Dal livello link a quello di trasporto

ARP

In una rete locale, il numero IP è *superfluo:* è sufficiente (e necessario) il numero MAC.

- ARP (Address Resolution Protocol): numero MAC da un numero IP

Come funziona

- Ogni nodo mantiene una tabella (ARP cache) in cui ci sono le associazioni già note
- altrimenti si chiede a tutti i nodi della rete locale chi ha un certo numero IP

ARP cache poisoning

L'assunzione di trust nella LAN...

1. Chi ha il numero IP 192.168.0.2?
2. Sono io: 00:23:a2:d6:f2:15
3. Le comunicazioni dirette a 192.168.0.2 vanno a chi riceve i frame destinati a 00:23:a2:d6:f2:15

In realtà funziona con arp reply (o anche request!) anche non sollecitate.

Che fare?

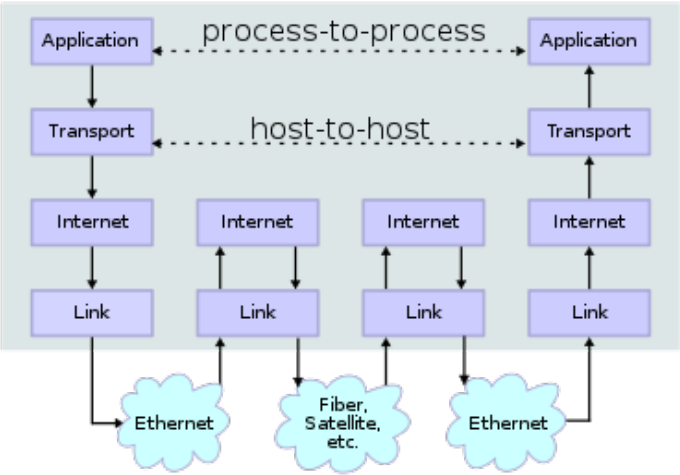
Una possibile difesa è l'uso di tabelle ARP statiche.

Attenzione: l'ARP poisoning ha anche usi perfettamente legittimi: p.es. per ridondanza o per fare convergere il primo collegamento verso un server di autenticazione.

Riassumendo

- Le reti locali assumono che i nodi collegati condividano una relazione di fiducia
- ARP poisoning: permette di *impersonare* uno o piú nodi della LAN

Il livello di trasporto



Poiché a livello applicativo la comunicazione avviene fra processi, a livello trasposto occorre identificare nodi e processi.

Porte

Un segmento di scambio fra due processi necessita di 4 numeri (*socket pair*)

$$\langle ip_1, n_1 : ip_2, n_2 \rangle$$

Port

Port

n_1, n_2 (0–65536) si dicono porte: quelle lato server devono essere note al client e rappresentano quindi il punto *d'accoglienza*.

Nota: il client è il nodo che inizia la connessione con il server.

Porte ben note

da http://www.iana.org/assignments/port-numbers

```
discard 9/tcp sink null
discard 9/udp sink null
ftp-data 20/tcp
ftp 21/tcp
ssh 22/tcp # SSH Remote Login Protocol
ssh 22/udp
telnet 23/tcp
smtp 25/tcp mail
domain 53/tcp # name-domain server
domain 53/udp
finger 79/tcp
www 80/tcp http # WorldWideWeb HTTP
pop3 110/tcp pop-3 # POP version 3
nntp 119/tcp readnews untp # USENET News Transfer Protocol
ntp 123/udp # Network Time Protocol
irc 194/tcp # Internet Relay Chat
https 443/tcp # http protocol over TLS/SSL
printer 515/tcp spooler # line printer spooler
# ...
```

Porte ben note

```
# Non fissate da IANA
socks 1080/tcp # socks proxy server
openvpn 1194/tcp
openvpn 1194/udp
rmiregistry 1099/tcp # Java RMI Registry
# ...
```

Porte convenzionali

Ricordare sempre che le porte sono numeri convenzionali (concordate con IANA per i numeri ≤ 1024)

- in generale non identificano un servizio, ma la possibilità di stabilire una connessione.

Uso delle porte a scopi di sicurezza

- vietare l'uso della porta destinazione 22 non significa vietare SSH, ma impedire che client e server possano accordarsi sull'uso di tale porta.
- il divieto può funzionare solo se l'amministratore controlla il server: se gestisce solo la rete il divieto è aggirabile.

Riassumendo

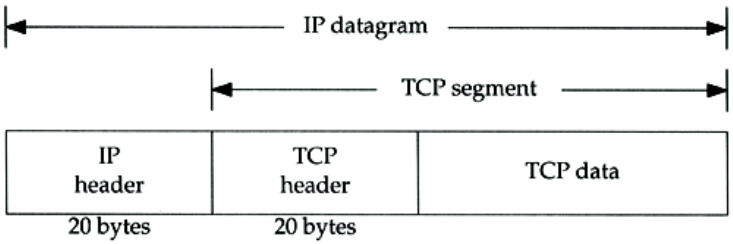
- Una connessione è identificata da 4 numeri $\langle ip_1, n_1 : ip_2, n_2 \rangle$
- Le porte sono semplicemente una convenzione stabilita fra client e server.

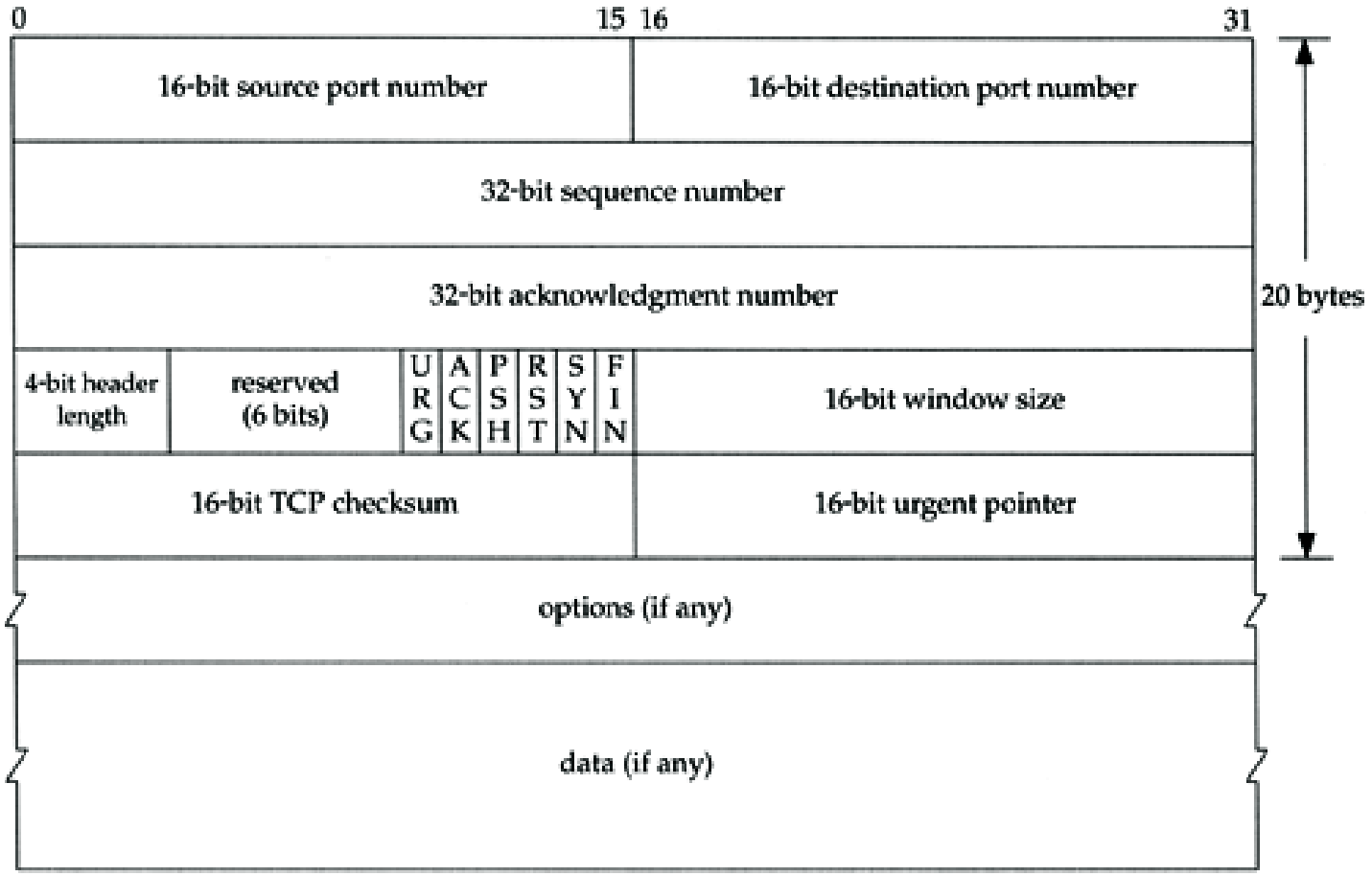
TCP

Transmission Control Protocol

- connection-oriented: è necessario uno handshake preliminare
- full-duplex
- lo "stato" è conservato interamente nei nodi (+ timer)

TCP segment





Flag

SYN richiesta di connessione, sempre il primo pacchetto di una comunicazione

FIN indica l'intenzione del mittente di terminare la sessione in maniera concordata

ACK conferma del pacchetto precedente, sia esso dati, SYN o FIN

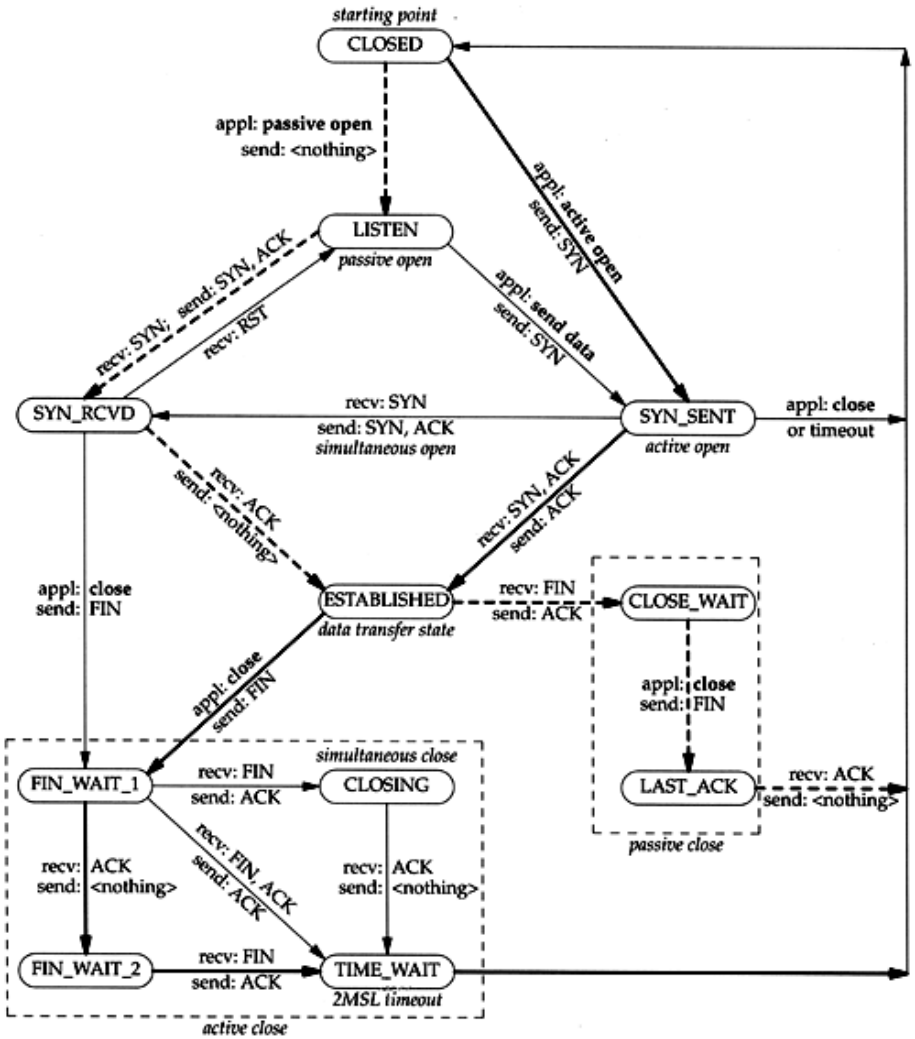
Flag

RST reset della sessione

PSH operazione di push, i dati che vengono inviati al destinatario non dovrebbero essere bufferizzati

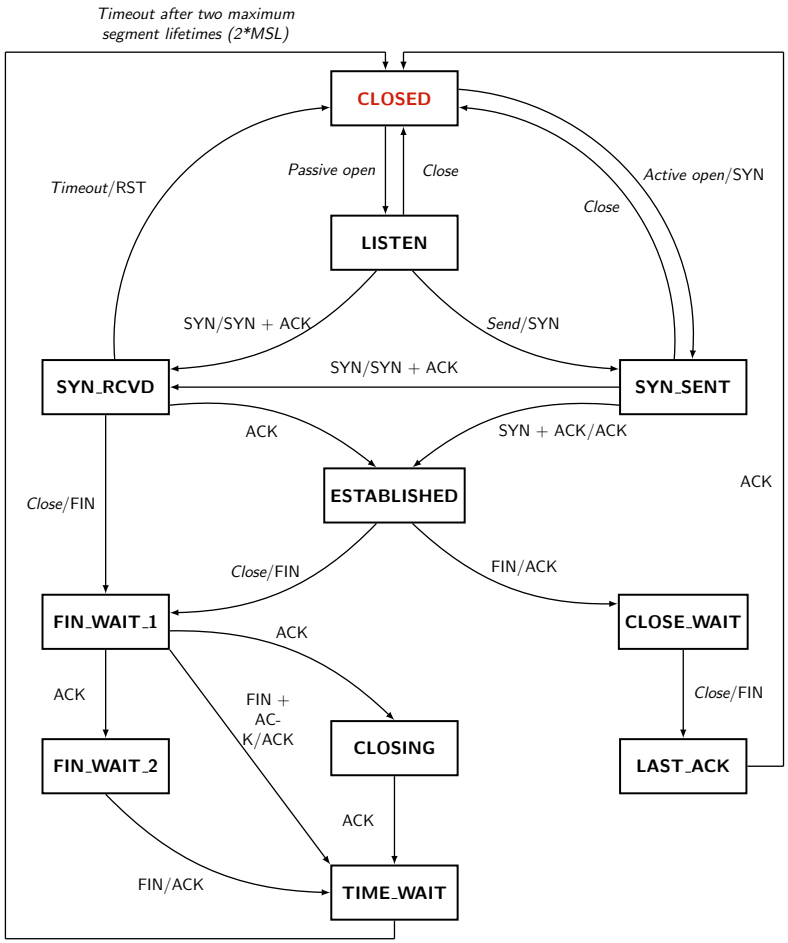
URG dati urgenti (es. CTRL+C) vengono inviati con precedenza sugli altri

State diagram

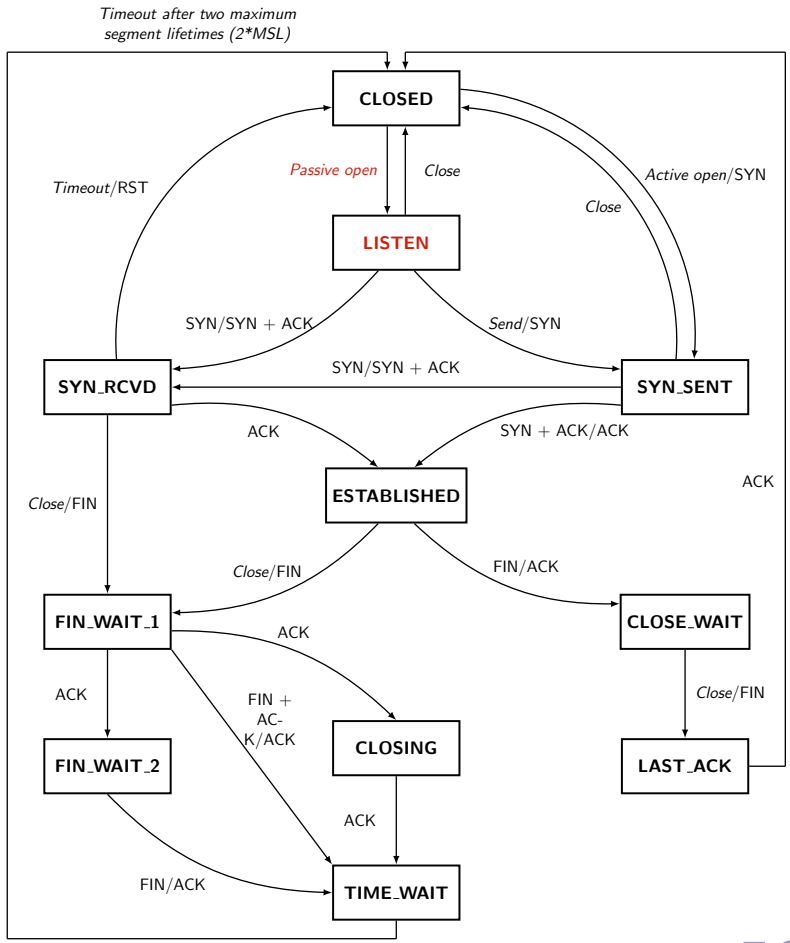


TCP state diagram

Client

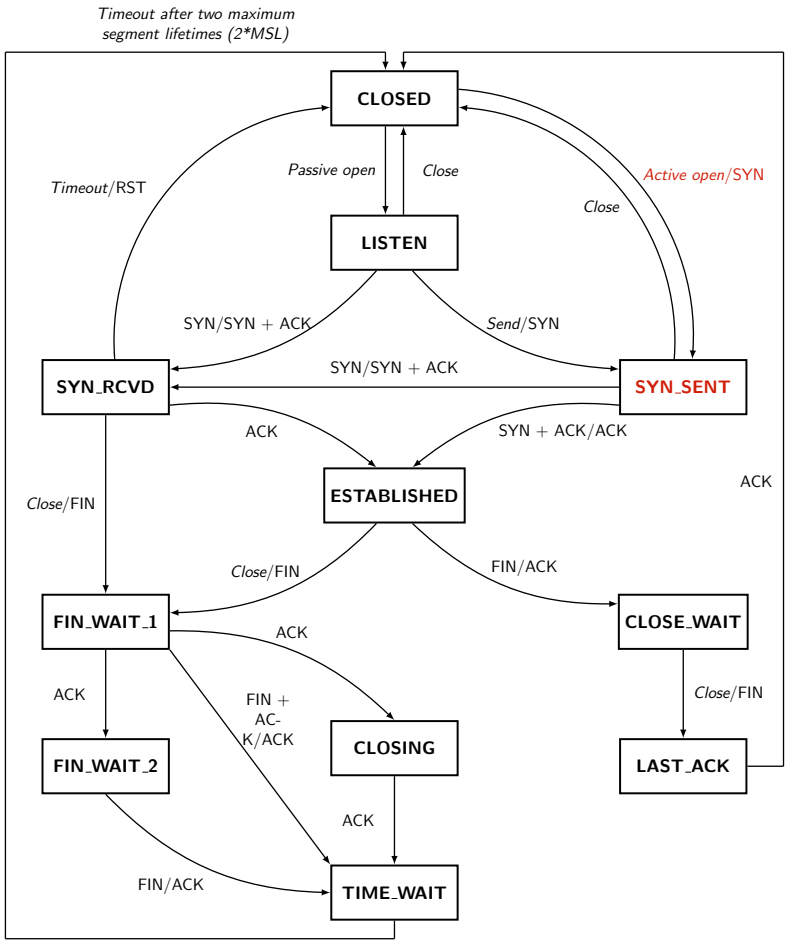


Server

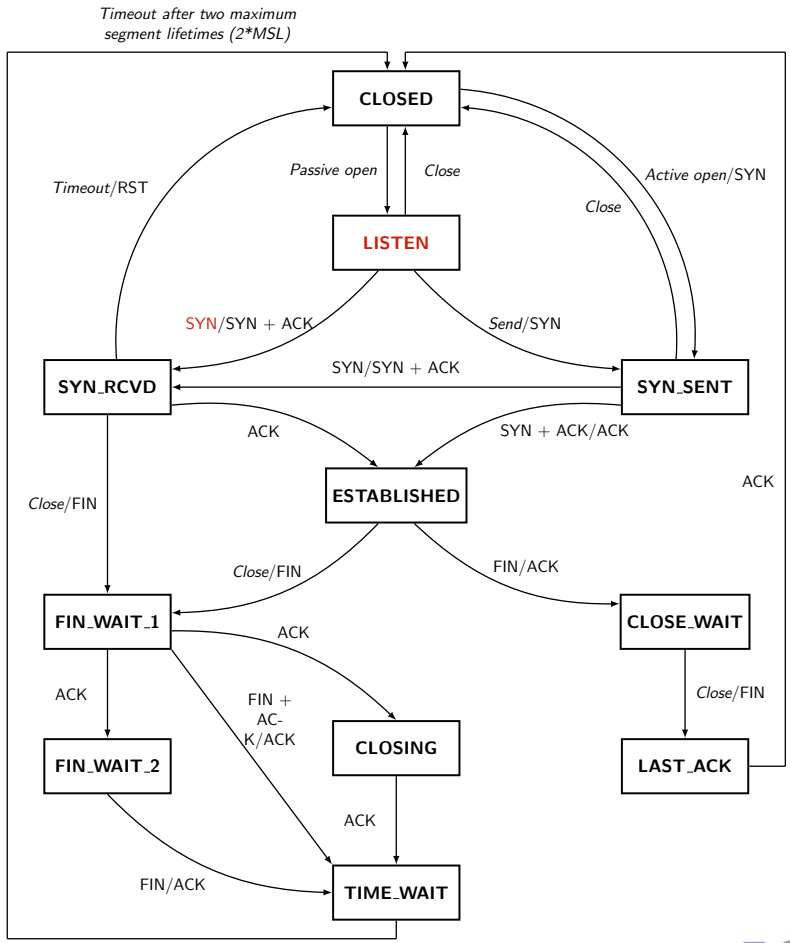


TCP state diagram

Client

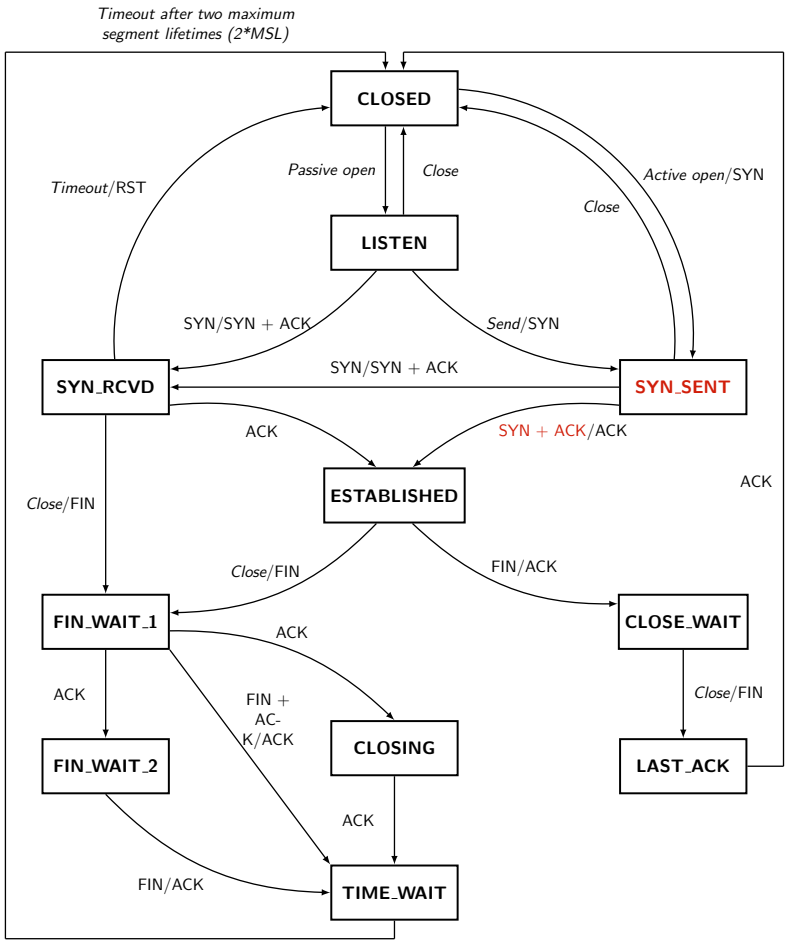


Server

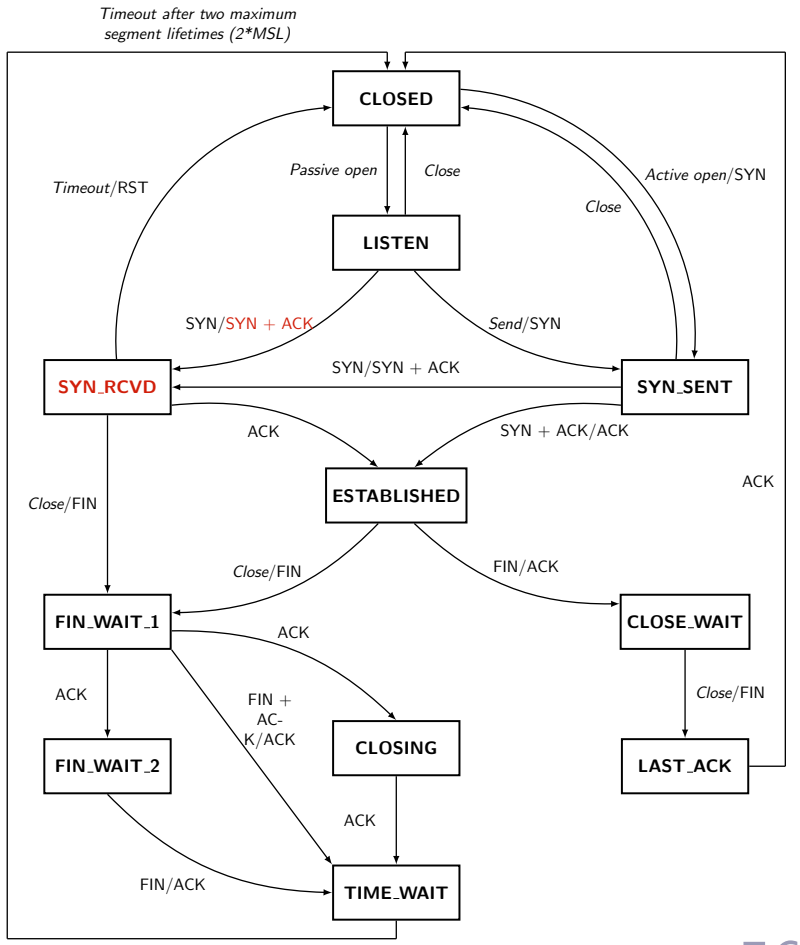


TCP state diagram

Client

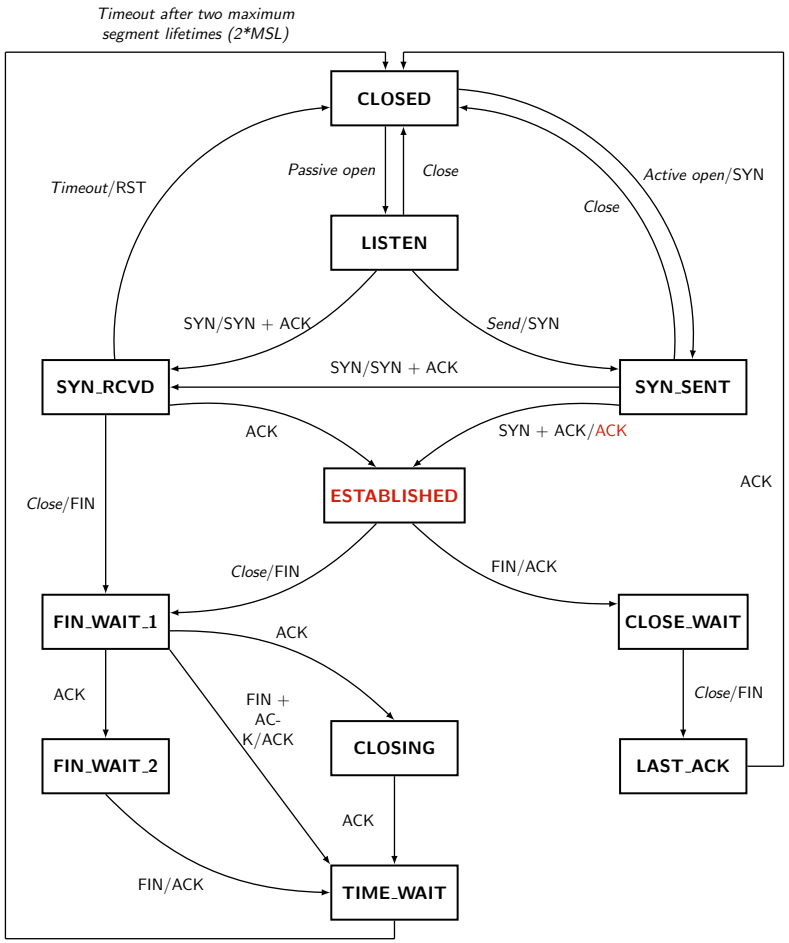


Server

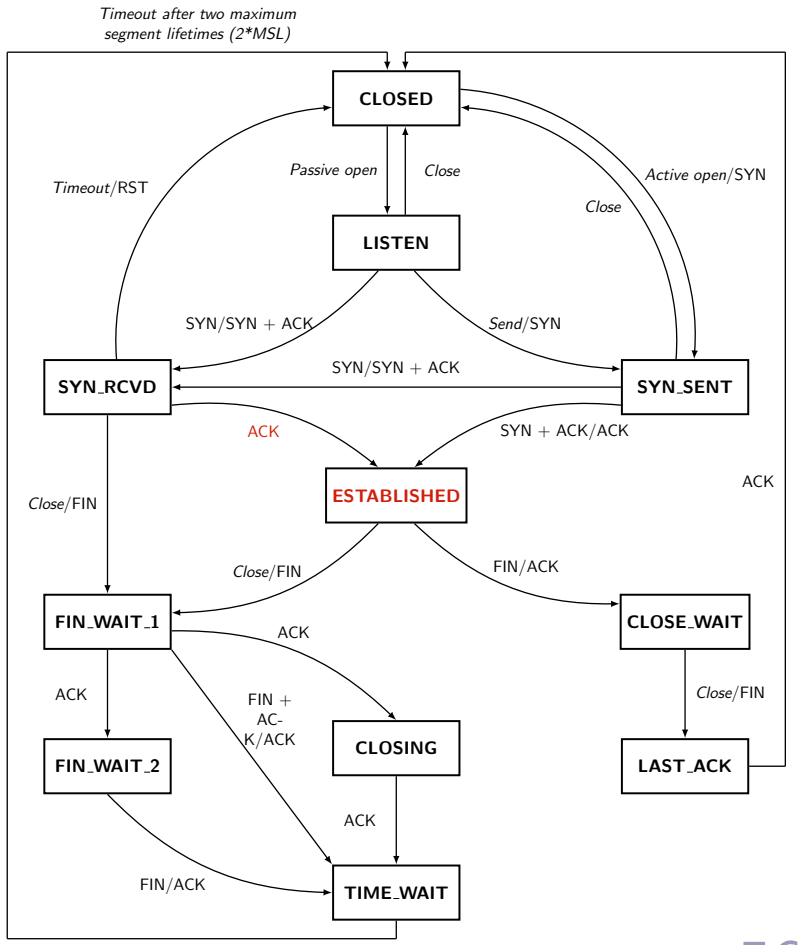


TCP state diagram

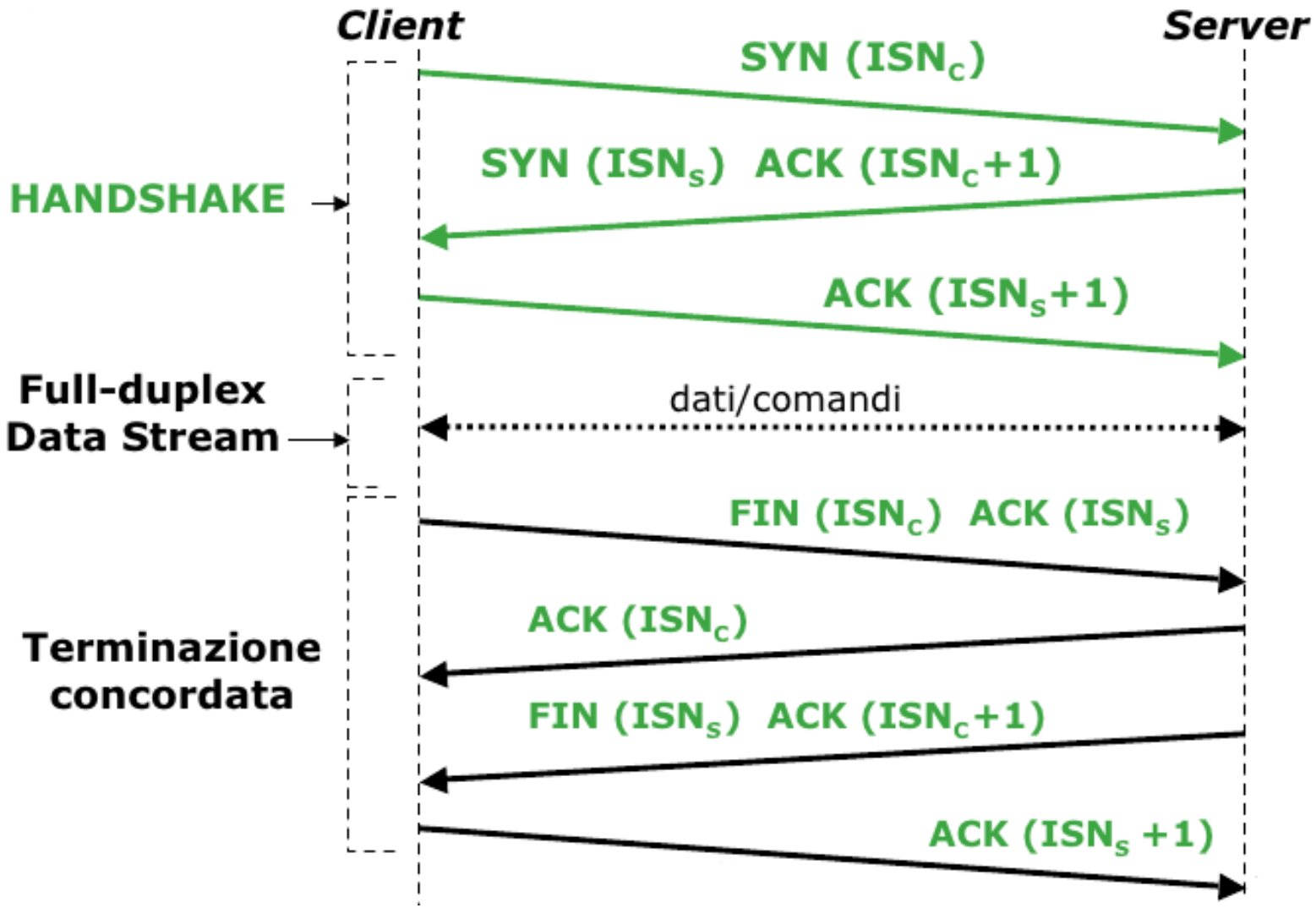
Client



Server



Sequence diagram

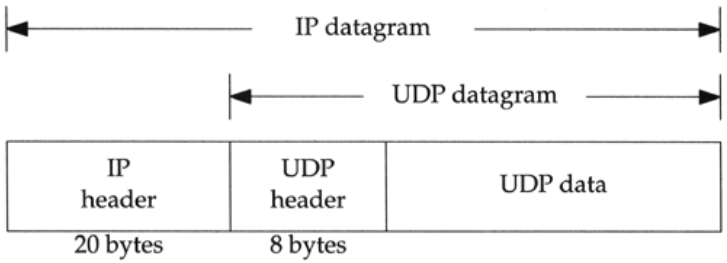


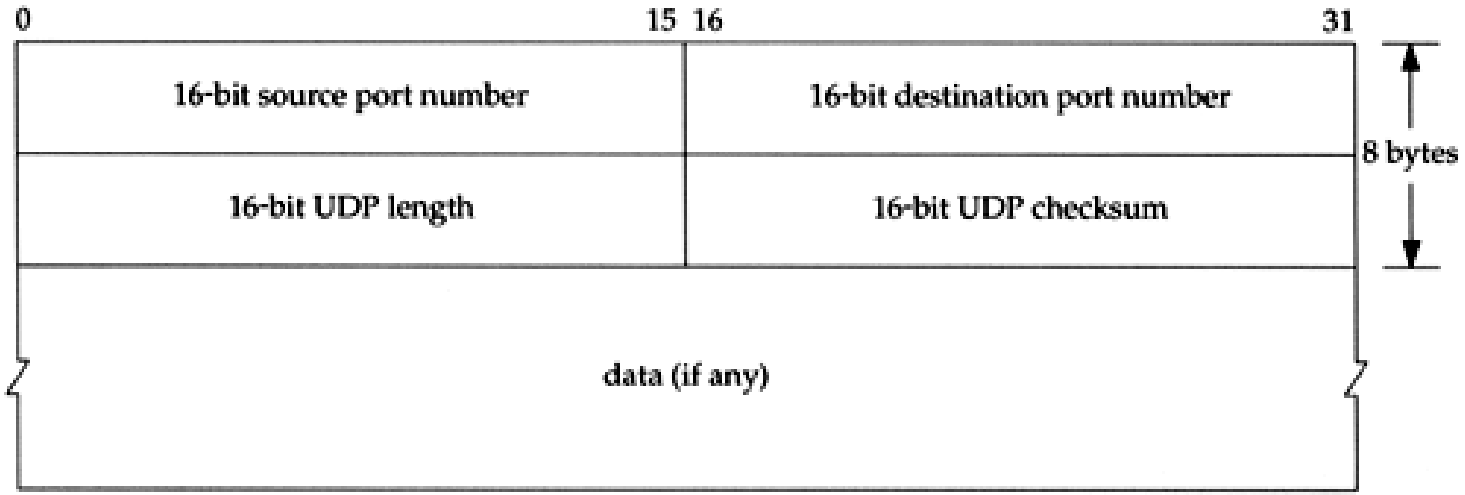
UDP

User Datagram Protocol

- Protocollo di trasporto “minimo”, senza connessione, senza stato
- minimo overhead (TCP: +20B, UDP: +8B)

UDP segment


IP datagram
UDP datagram
IP header
UDP header
UDP data
20 bytes
8 bytes


0
15 16
31
16-bit source port number
16-bit destination port number
16-bit UDP length
16-bit UDP checksum
8 bytes
data (if any)

Checksum

Sia TCP che UDP portano nei segmenti un checksum.
Attenzione: ha lo scopo di proteggere solo dagli errori di trasmissione, non dalle alterazioni maligne!

Riassumendo

- TCP: connessione tramite 3-way handshake, stato mantenuto dai nodi
- UDP: minimo overhead rispetto a IP, nessuno stato
- Protocolli senza particolari caratteristiche di sicurezza (confidenzialità o integrità)

Problemi intrinseci in TCP/IP

- Non c'è autenticazione fra le parti
- I controlli d'integrità sono banali
- Si difende la disponibilità della rete dalla congestione, ma non la possibilità di connettersi ad un determinato nodo

IP spoofing

Il campo `SRC` dello header IP è falsificabile senza particolari difficoltà.

- Le autenticazioni basate su indirizzi IP sono insicure, soprattutto all'interno di una rete locale.
- Fra l'altro la presenza di numeri IP duplicati può causare *denial of service*

Dal punto di vista dell'attaccante

Se l'IP sorgente è falso

- le risposte andranno al vero nodo titolare
- "spoofare" l'IP non è sufficiente per inserirsi in una connessione TCP

Spoofing in connessioni TCP

Per una connessione serve lo handshaking

1. $C \rightarrow S : SYN, ISN_C$
2. $S \rightarrow C : SYN, ISN_S, ACK(ISN_C)$
3. $C \rightarrow S : ACK(ISN_S)$

Se ISN_S è imprevedibile è difficile (2^{-32}) per X farsi passare per C (e se C è *up*, manderà un RST).

Initial sequence number

RFC793: ISN va incrementato circa 1 volta ogni 4 microsecondi per evitare confusioni con connessioni duplicate. Alcune implementazioni ancora piú prevedibili (famose le kick-off war con IRC e stack vulnerabili come quelli di alcune versioni di Windows)

ISN sicuri

Non può essere completamente casuale. RFC1948 (ora RFC6528) propone:

$$ISN = M + F_S(localhost, localport, remotehost, remoteport)$$

con F_S funzione hash crittografica, non prevedibile da un attaccante e M un contatore incrementato ogni 4 microsecondi.

Riassumendo

- I campi dei pacchetti sono facilmente falsificabili
- Il numero di sequenza è un parametro particolarmente delicato

Frammentazione

I segmenti TCP sono spesso frammentati e riassemblati dal destinatario.
Un *man-in-the-middle* può alterare i frammenti: in questo caso non serve indovinare i sequence number. I checksum sono facili da *aggiustare* perché semplici controlli d'errore di trasmissione.

SYN flooding

Quando un host S riceve una richiesta SYN, tiene traccia per un certo tempo (spesso 75s) della connessione in una coda.

- La coda ha lunghezza finita: talvolta addirittura 5
- SYN cui non segue un ACK possono portare a DoS

I SYN-cookie (D. J. Bernstein) usano gli ISN per evitare il flooding.

Fingerprinting

Dall'esame (non intrusivo) dei pacchetti di rete è possibile identificare molti dettagli utili negli attacchi...

- p.es. `p0f` è in grado di riconoscere molte implementazioni di stack TCP/IP

Topologia della rete

È possibile studiare la topologia della rete esaminando il TTL

- p.es. Windows TTL=128, Linux TTL=64
- TTL==80 $\Rightarrow$ Windows, e il nodo è distante 48 hop

Letture obbligatorie

- Steven M. Bellovin. *A Look Back at "Security Problems in the TCP/IP Protocol Suite".* In Proceedings of the 20th Annual Computer Security Applications Conference (ACSAC '04). 229-249.
- Steven M. Bellovin, *Defending Against Sequence Number Attacks*, February 2012, RFC6528

Riassumendo

- Il controllo d'integrità è lasco
- Il DoS può essere ottenuto abbastanza facilmente
- Gli header dei pacchetti rivelano molte informazioni ai potenziali attaccanti