

Nivelul Transport

Lenuta Alboaie
adria@info.uaic.ro

Cuprins

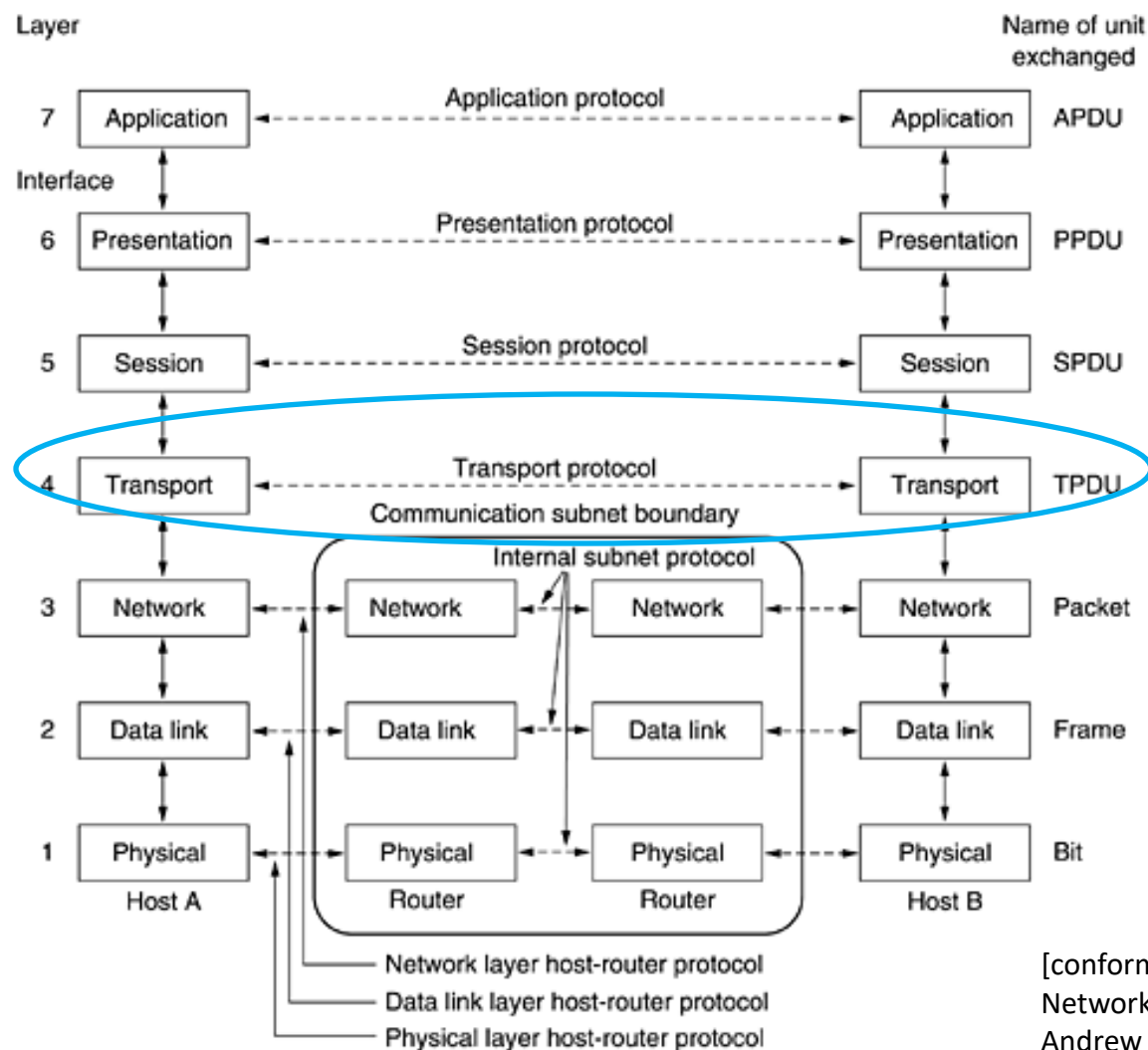
Nivelul Transport

- Preliminarii
- UDP (User Datagram Protocol)
- TCP (Transmission Control Protocol)
- TCP versus UDP

Nivelul transport | Preliminarii

Utilizatoare de
servicii de
Transport

Furnizare de
servicii de
transport

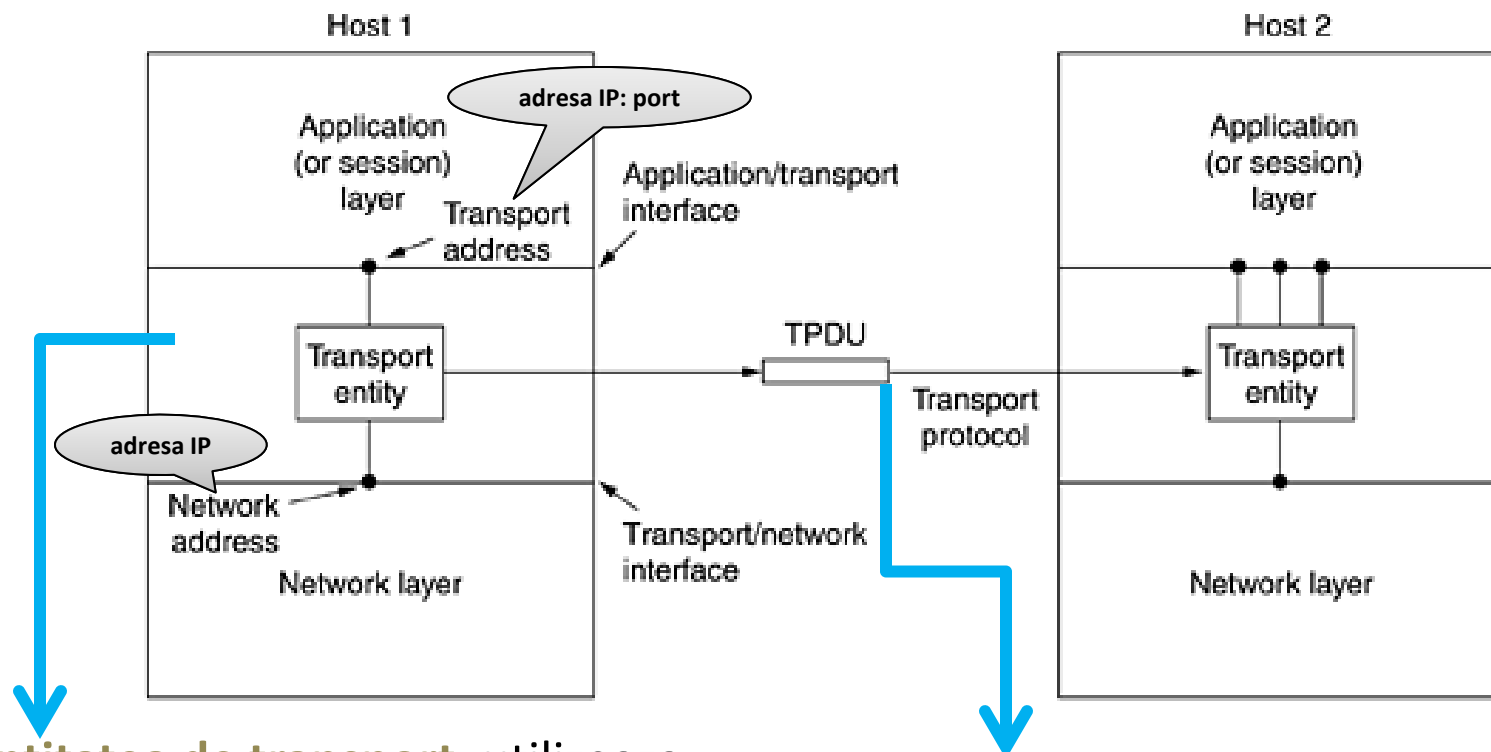


[conform Computer Networks, 2010 – Andrew S. Tanenbaum, et.al.]

Figura. Modelul OSI

Nivelul Transport

- Relatia (logica) intre nivele: retea-**transport**-aplicatie



Entitatea de transport: utilizeaza serviciile nivelului retea si ofera servicii nivelului superior

TPDU (Transport Protocol Data Unit) – unitatea de date pentru transport

[conform Computer Networks, 2010
– Andrew S. Tanenbaum, et.al.]

Nivelul Transport

- Oferă servicii mult mai fiabile decât cele de la nivelul rețea (e.g. pachetele pierdute/incorecte de la nivelul rețea pot fi detectate/corectate la nivelul transport)
- **Calitatea serviciilor – QoS** (*Quality of Service*)
 - Intarzierea la stabilirea conexiunii
 - Productivitatea sau rata de transfer
 - Intirzierea la transfer
 - Rata reziduala a erorilor
 - Protectia
 - Prioritatea
 - Rezilienta

Nivelul Transport

- Asigura comunicarea logica dintre **procese** rulinde pe *host-uri* diferite (comunicare *end-to-end*)
 - (Curs anterior!) Nivelul retea asigura comunicarea logica intre **host-uri**
- **PORT**
 - Adauga o noua dimensiune adreselor IP de la nivelul retea
 - Se asociaza unei aplicatii (serviciu) si nu unei gazde
 - Un proces poate oferi mai multe servicii => poate utiliza mai multe porturi
 - Un serviciu poate corespunde mai multor procese

Nivelul Transport

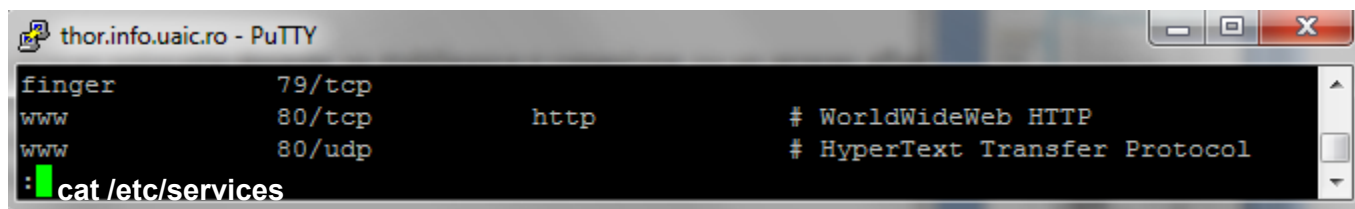
- **PORT**

- Este un numar pe 16 biti si poate avea valori intre 0 – 65535
- 1-1024 – valori rezervate (*well-known*), 1-512 servicii de sistem (IANA – Internet Assigned Number Authority: RFC 1700)

Exemple:

/etc/services : sunt precizate porturile asociate serviciilor sistem

HTTP – 80; SMTP – 25; telnet – 23; SSH - 22



A screenshot of a PuTTY terminal window titled 'thor.info.uaic.ro - PuTTY'. The terminal displays the output of the command 'cat /etc/services'. The output shows several entries: 'finger 79/tcp', 'www 80/tcp http # WorldWideWeb HTTP', and 'www 80/udp # HyperText Transfer Protocol'. The prompt is a green cursor followed by ': cat /etc/services'.

```
thor.info.uaic.ro - PuTTY
finger      79/tcp
www         80/tcp      http          # WorldWideWeb HTTP
www         80/udp      # HyperText Transfer Protocol
: cat /etc/services
```

Nivelul Transport

Primitive generale

- Permit utilizatorilor nivelului transport (e.g. programe de aplicatie) sa acceseze serviciile

Primitiva	Unitatea de date trimisa (TPDU)	Explicatie
LISTEN	(nimic)	Se blocheaza pana cand un proces incearca sa se conecteze
CONNECT	CONNECTION REQUEST	Incearca sa stabileasca conexiunea
SEND	DATA	Transmite informatie
RECEIVE	(nimic)	Se blocheaza pana cand se primeste date trimise
DISCONNECT	DISCONNECTION REQ.	Trimisa de partea care vrea sa se deconecteze

Nivelul Transport

Cele mai importante protocoale la nivelul transport:

- **TCP (Transmission Control Protocol)** – protocol de transport orientat conexiune; RFC 793 (1122), 1323
- **UDP (User Datagram Protocol)** – protocol de transport neorientat conexiune; RFC 768

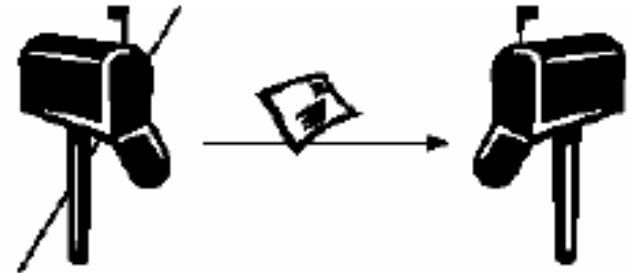
UDP

- Protocol de transport neorientat conexiune, nesigur, minimal
- Nu ofera nici o calitate suplimentara serviciilor
- Nu recurge la negocieri sau la confirmari ale primirii datelor

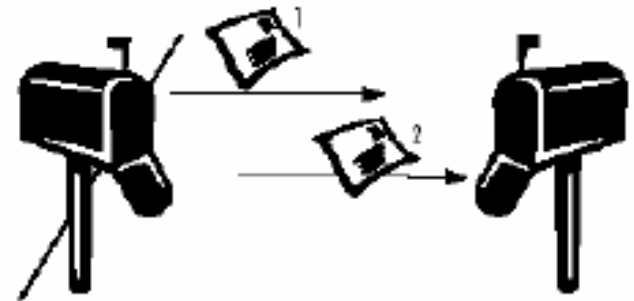
UDP

- **Analogie:** UDP – similar **postei terestre**

➤ Trimiterea unei scrisori



➤ Nu se garanteaza ordinea receptionarii



➤ Mesajul se poate pierde



[conform Retele de calculatoare – curs
2007-2008, Sabin Buraga]

UDP

- Utilizeaza IP
- Pentru a oferi servicii de comunicare intre procese foloseste **porturi**
- UDP transmite pachete: antet (8 bytes) + continut

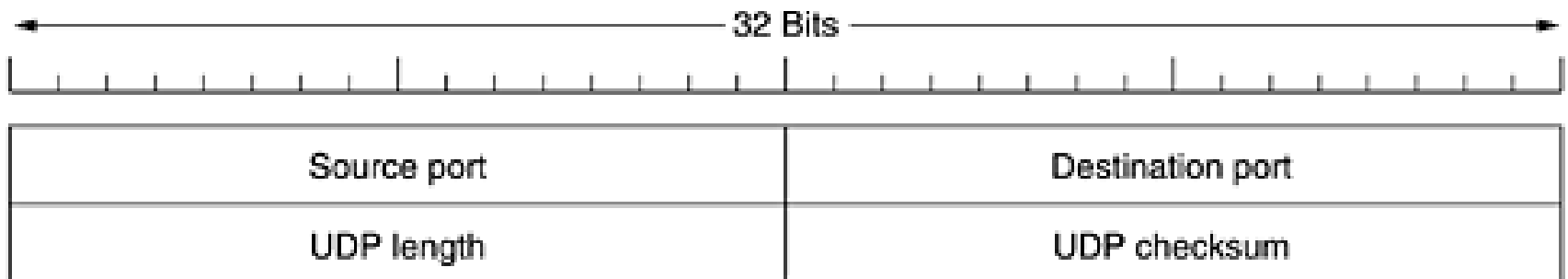


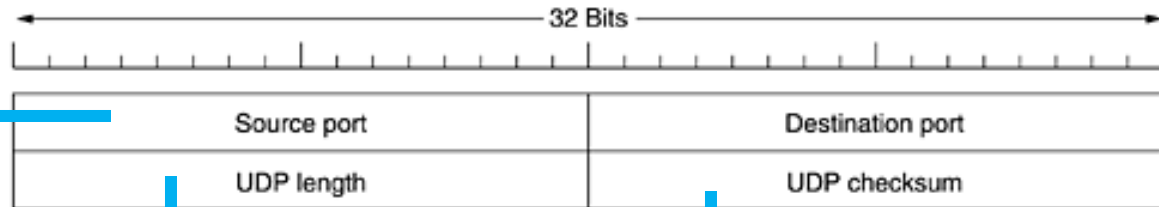
Figura: Antet UDP

[conform Computer Networks, 2010
– Andrew S. Tanenbaum, et.al.]

UDP

Figura: Antet UDP

- *Source port* si *Destination port* identifica “end-point”-urile de pe masinile sursa si destinatie



- *UDP length* = 8 bytes + dimensiunea continutului

- *UDP checksum* nu este obligatoriu

UDP

- Exemple de utilizari:

- **DNS** (Domain Name System)

Use-case: A are nevoie de IP-ul *host*-ului cu numele `www.info.uaic.ro`

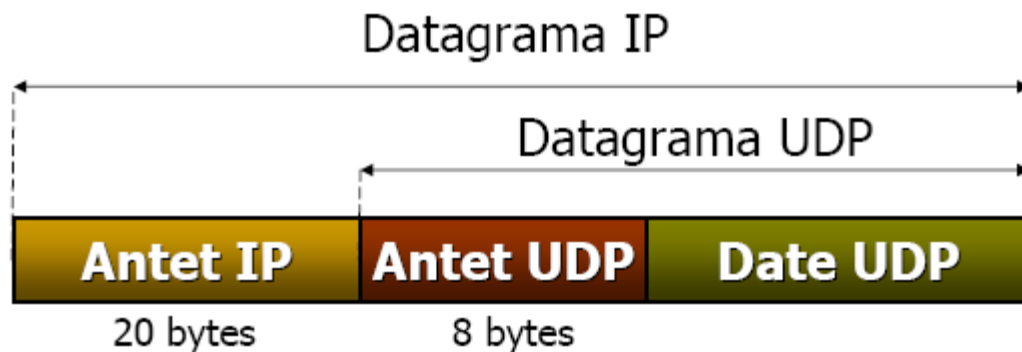
Pas 1. A trimite un pachet UDP continind numele *host*-ului : `www.info.uaic.ro`

Pas 2. Serverul de DNS trimite un pachet UDP de raspuns continind adresa IP a *host*-ului : `85.122.23.146`

- **RPC** (Remote Procedure Call) – mecanism de apel al procedurilor la distanta

UDP

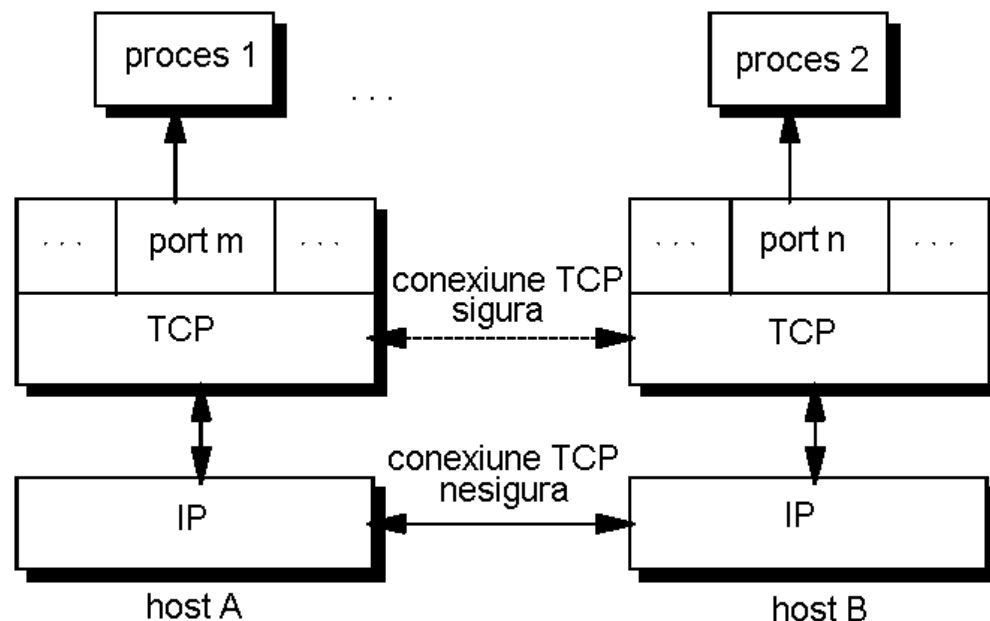
- Ce nu ofera?
 - Controlul fluxului
 - Controlul erorilor
 - Retransmisia la receptionarea unui pachet eronat
- Ce ofera?
 - Folosind **port-uri** extinde protocolul IP pentru comunicarea între procese aflate pe host-uri diferite si nu numai între host-uri



[conform Retele de calculatoare – curs
2007-2008, Sabin Buraga]

TCP - Transmission Control Protocol

- Protocol de transport orientat conexiune, fara pierdere de informatii
- Vizeaza oferirea calitatii maxime a serviciilor
- Integreaza mecanisme de stabilire si de eliberare a conexiunii
- Controleaza fluxul de date (*stream-oriented*)
- Utilizat de multe protocoale de aplicatii: HTTP, SMTP, ...

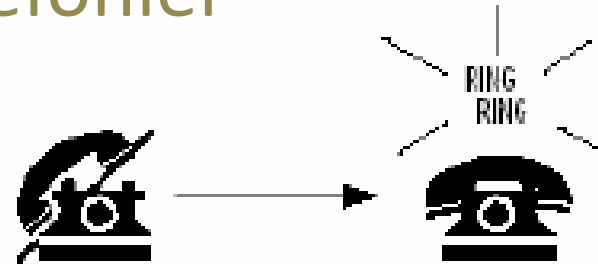


[conform IBM – TCP/IP Tutorial and Technical Overview, 2006]

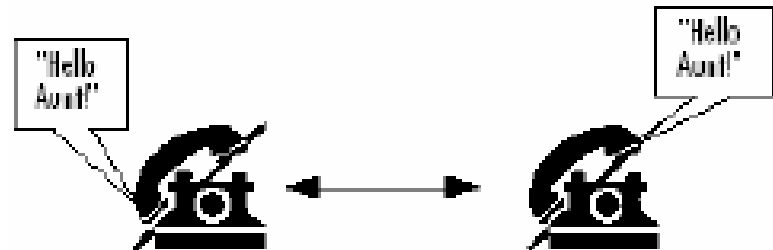
TCP

- Analogie: TCP – similar telefoniei

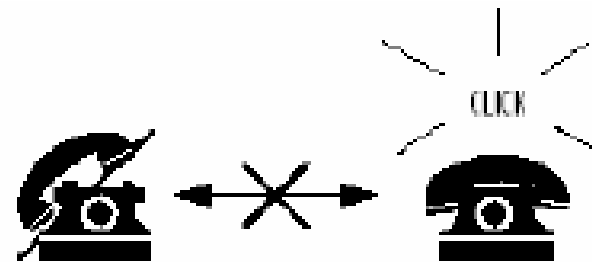
➤ Inițierea unei convorbiri



➤ Dialogul dintre parti



➤ Terminarea convorbirii



[conform Rețele de calculatoare – curs
2007-2008, Sabin Buraga]

TCP

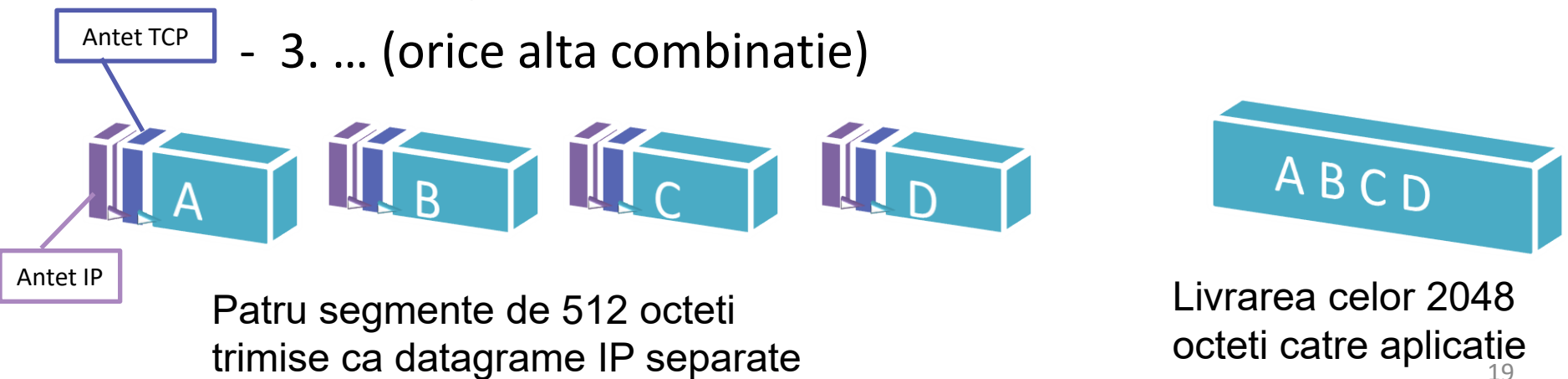
- Utilizeaza **conexiuni**, nu porturi ca abstractiuni fundamentale
- Ambele parti (expeditorul, destinatarul) trebuie sa participe la realizarea conexiunii
- Conexiunile se identifica prin perechi reprezentate de **adresa IP:PORT** (*soclu - socket*)
 - Exemplu:
(85.122.23.146: 12345, 85.122.23.146: 22)
- Una din parti ofera o **deschidere pasiva** - asteapta aparitia unei cereri de conectare a partenerului de comunicare care realizeaza o **deschidere activa**
- Un *soclu* poate fi partajat de conexiuni multiple de pe aceeasi masina

TCP

- Conexiunile TCP sunt full-duplex
- O conexiune TCP este un flux de octeti si nu un flux de mesaje

Exemplu:

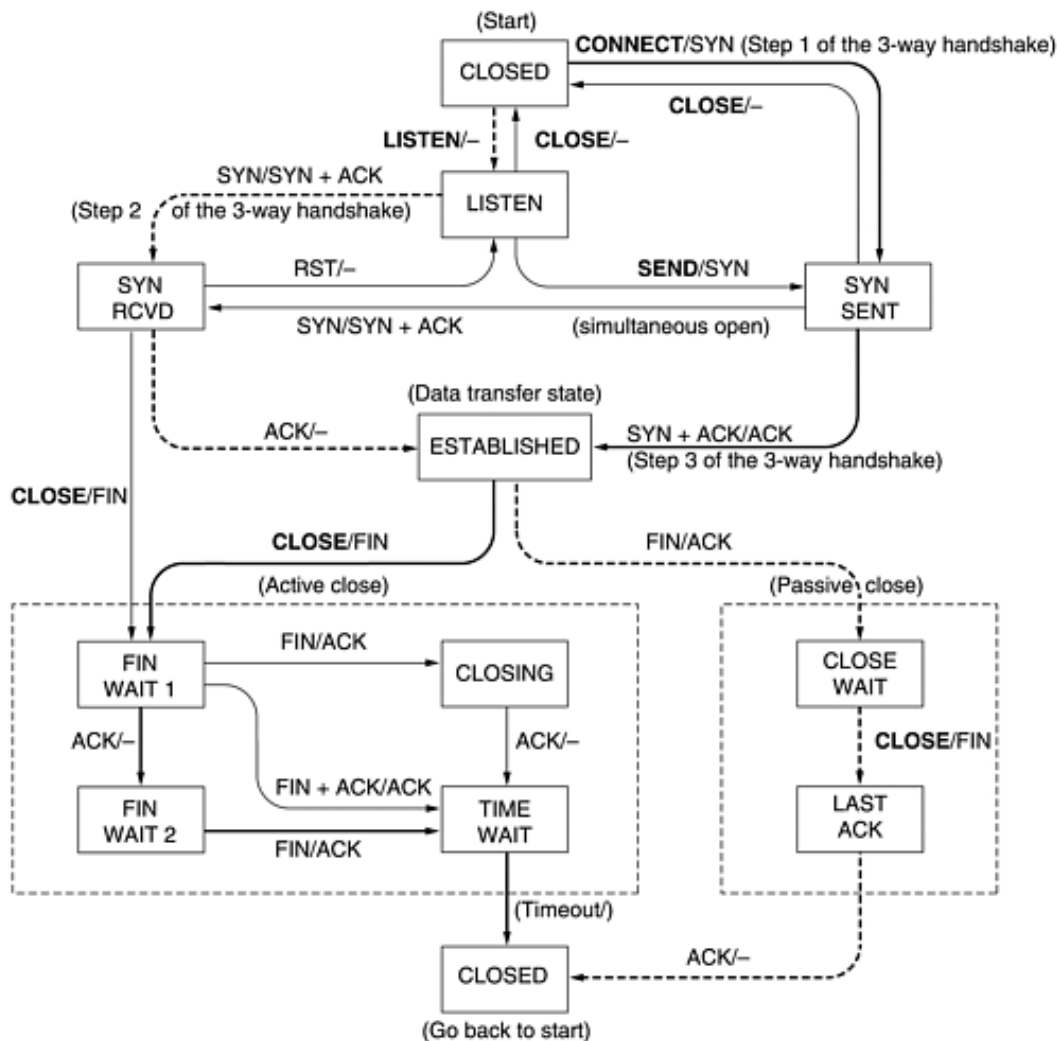
- Emitatorul trimite 4 fragmente de 512 octeti
- Receptorul poate primi:
 - 1. doua fragmente de 1024 de octeti
 - 2. un fragment de 1 octet si unul de 2047 de octeti
 - 3. ... (orice alta combinatie)



TCP

Automatul finit TCP

- Modelează comportamentul protocolului
- Stările sunt utilizate la managementul conexiunii



Legenda:

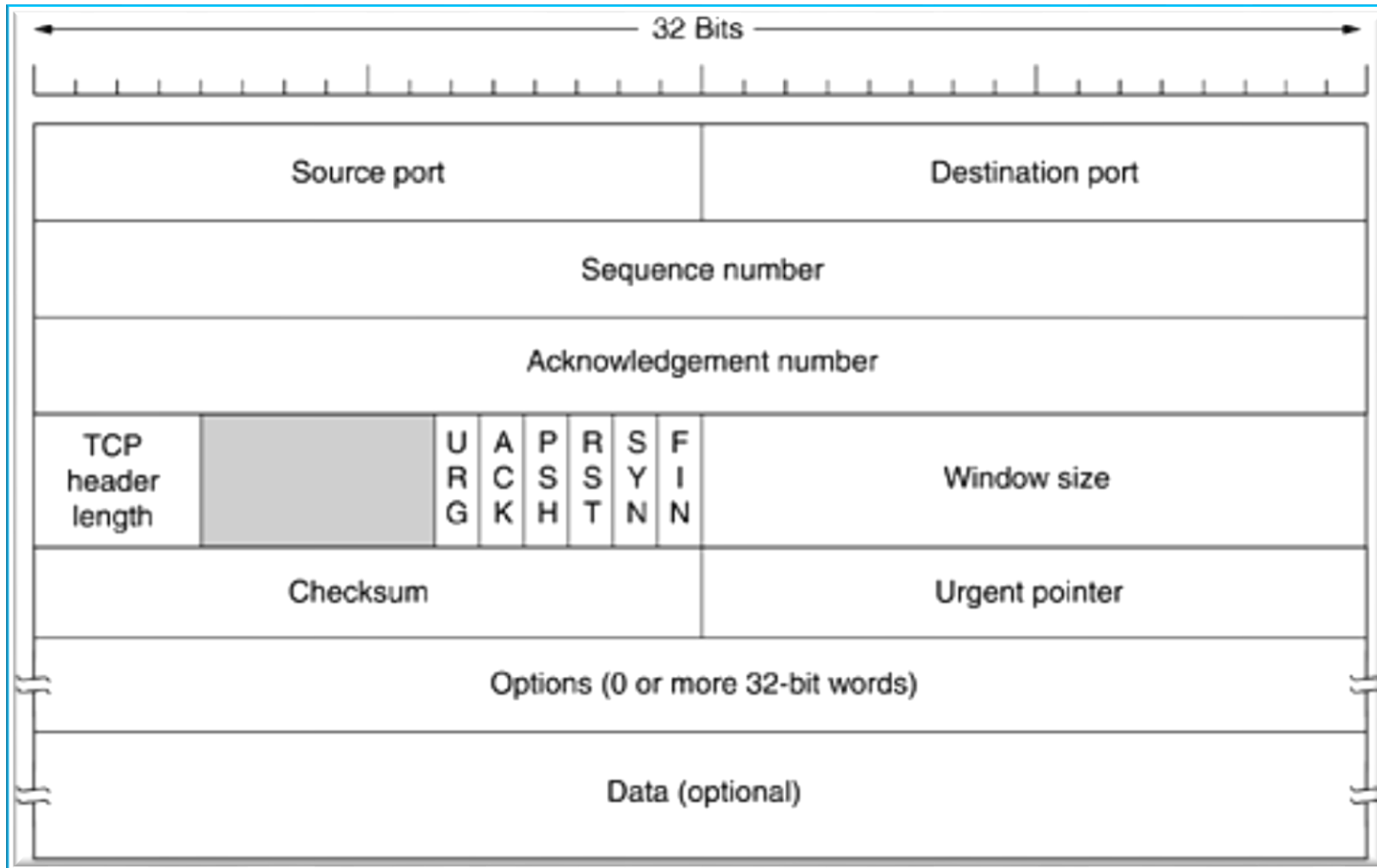
_____ Client
 ----- Server

-Fiecare linie este etichetată cu o pereche *eveniment/acțiune*
 Exemplu: ACK/-

[conform Computer Networks, 2010 – Andrew S. Tanenbaum, et.al.]

TCP

Antetul TCP



[conform
Computer
Networks, 2010
– Andrew S.
Tanenbaum,
et.al.]

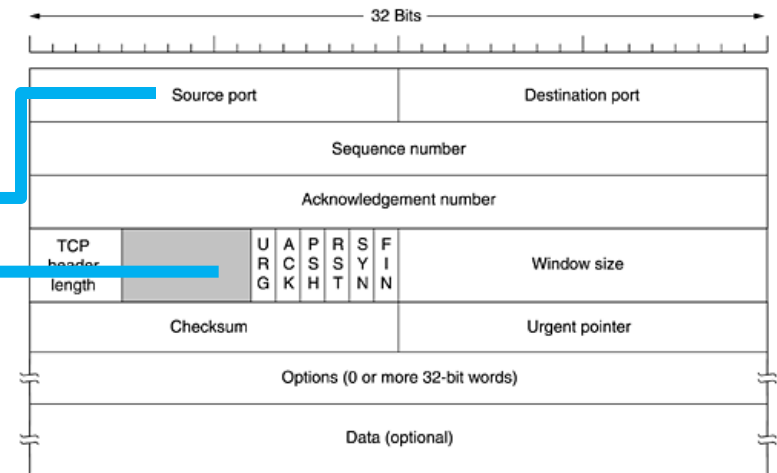
TCP

Antetul TCP

Cimpurile *Source Port* si *Destination Port* identifica punctele finale ale conexiunii

Flaguri de control

- **URG** (URGence) (1 bit) – daca este setat, campul *Urgent Pointer* este folosit
- **ACK** (ACKnowledgment)
 - daca este setat campul *Acknowledge Number* este folosit;
 - este setat in toate pachetele de confirmare



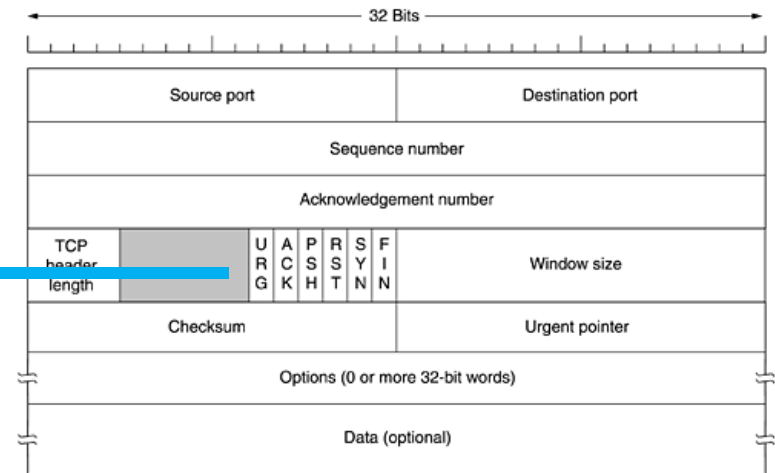
TCP

Antetul TCP

- **Flaguri de control**

- **PSH** (**PuSH**) – datele vor fi transmise imediat aplicatiei destinatare

- **RST** (**ReSeT**) – semnalizeaza erori in conexiune (e.g. deschiderea unei conexiuni este refuzata)



TCP

Antetul TCP

- **Flaguri de control**

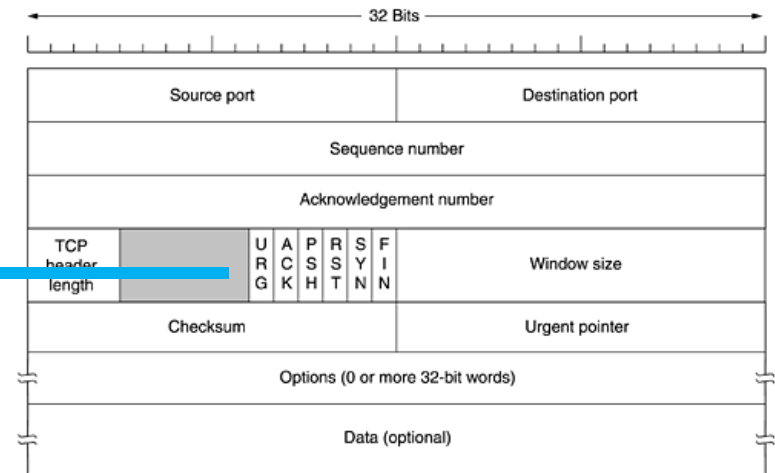
- **SYN** (SYNchronize)

- folosit pentru stabilirea conexiunii

(Cererea de conexiune: SYN=1,
ACK=0; Raspunsul la cerere:
SYN=1, ACK=1;)

- indica cererea de conexiune si conexiune acceptata

- **FIN** (FINish)- indica inchiderea conexiunii

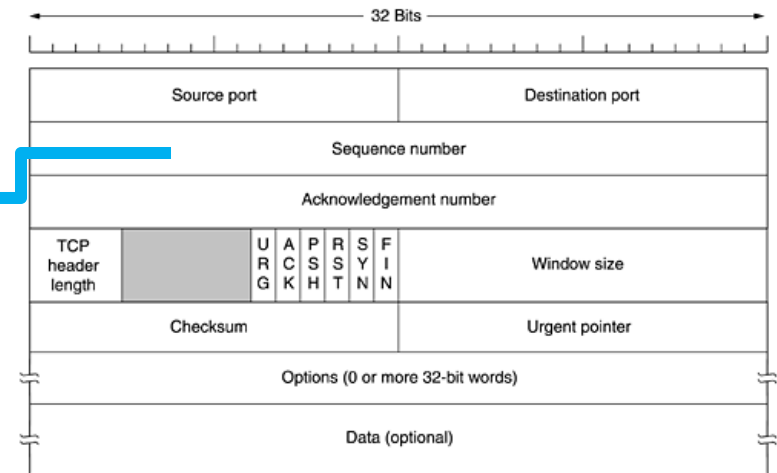


TCP

Antetul TCP

Campul *Sequence Number* (SEQ)

- Daca flagul SYN este setat => SEQ are valoarea initiala a numarului de ordine;
- Daca flagul SYN este nesetat => SEQ are ca valoare numarul de ordine a primului octet trimis in acest pachet



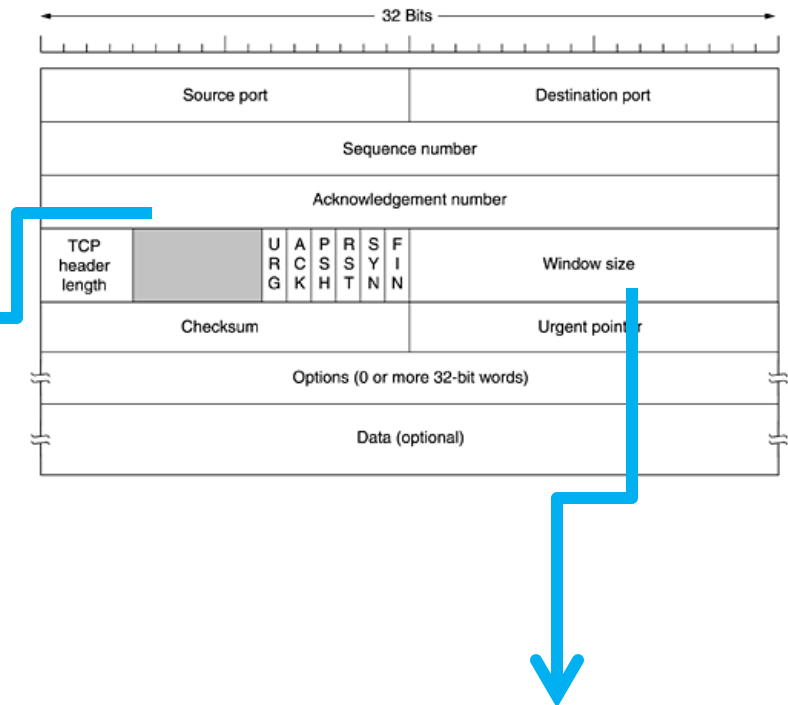
TCP

Antetul TCP

Campul *Acknowledge Number*

- Daca flagul ACK este setat
=> are valoarea numarului
de ordine al urmatorului
octet care se asteapta a fi
receptionat;

Campul *Window size* – numarul de octeti pe care
receptorul doreste sa ii receptioneze (controlul
fluxului)

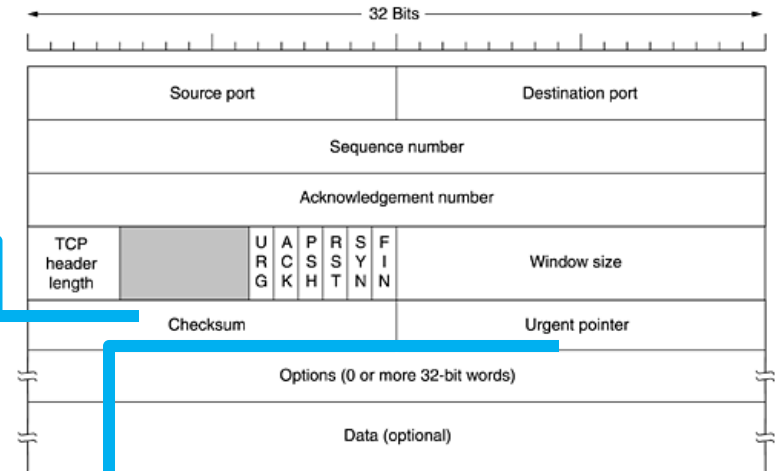


TCP

Antetul TCP

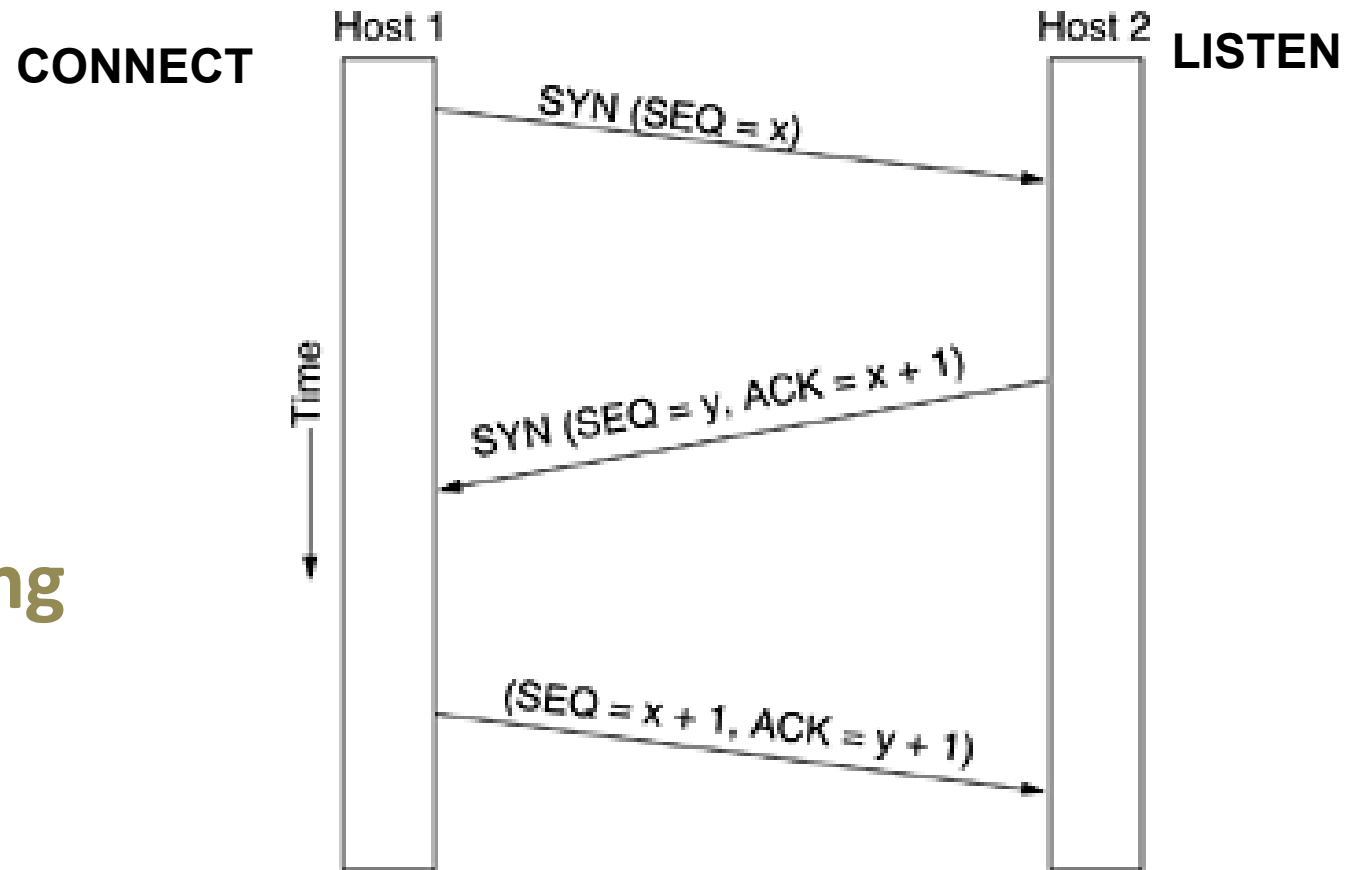
Campul **Checksum** – folosit pentru verificarea sumei de control al pachetului TCP (antet si data)

Campul **Urgent Pointer** – daca flagul URG este setat, indica deplasamentul fata de numarul curent de ordine la care se gaseste informatia urgenta



Administrarea conexiunii TCP

Stabilirea conexiunii Three-way handshaking



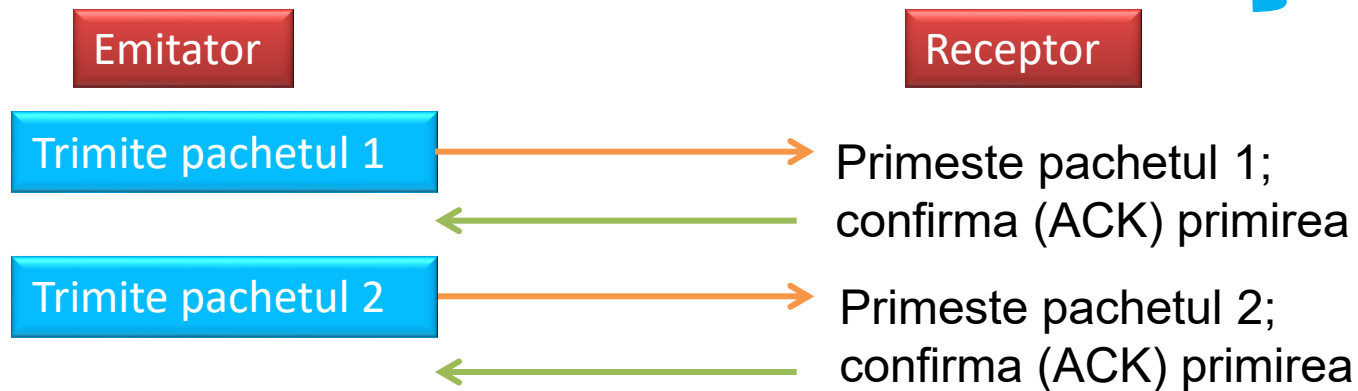
[conform Computer
Networks, 2010 – Andrew
S. Tanenbaum, et.al.]

Controlul fluxului

Mecanism general:

- Protocol de comunicare simplu: se trimite un pachet si apoi se asteapta confirmarea de primire de la destinatar inainte de a trimite pachetul urmator.
- Daca confirmarea (ACK) nu este primita intr-un interval de timp prestabilit, pachetul este trimis din nou

**Fiabilitatea
Comunicarii**
(*reliability*)



Problema: mecanismul asigura siguranta comunicarii dar determina folosirea doar a unei parti din banda disponibila a retelei

Administrarea conexiunii TCP

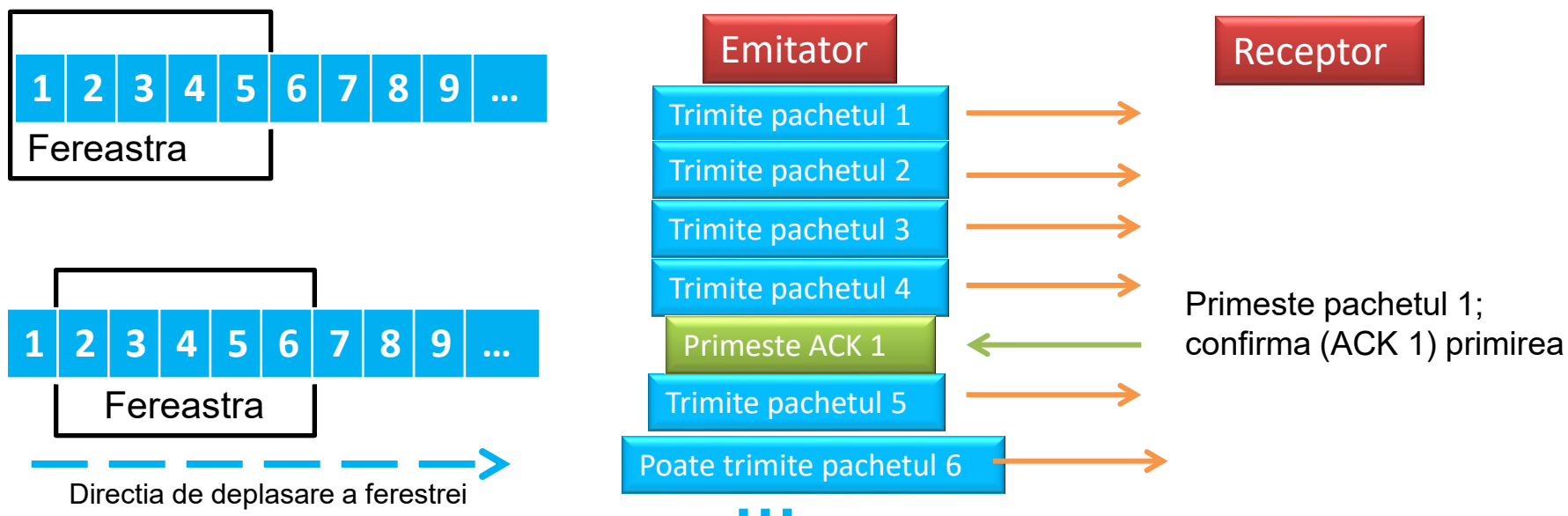
Fereastra glisanta – context:

- TCP asigura un flux de octeti => numerele de secventa sunt atribuite fiecarui octet din stream
- TCP imparte fluxul de octeti in segmente TCP; segmentele trimise si confirmarile vor transporta numere de secventa (vezi slide 19)
- Dimensiunea ferestrei este exprimata in octeti si nu in numar de pachete
- Dimensiunea ferestrei este determinata de catre receptor cind conexiunea este stabilita si este variabila in timpul transferului de date; Fiecare mesaj ACK va include dimensiunea ferestrei pe care receptorul este apt sa opereze in momentul respectiv al comunicarii

Administrarea conexiunii TCP

Controlul fluxului

- Protocol de comunicare cu **fereastra glisanta** :
 - emitatorul grupeaza pachetele intr-un *buffer*
 - emitatorul poate transmite pachetele din fereastra fara sa receptioneze o confirmare (ACK), dar porneste cate un cronometru (care va semnaliza depasirea unui interval de timp prestabilit) pentru fiecare dintre pachete
 - receptorul trebuie sa confirme fiecare pachet primit, indicand numarul de secventa al ultimului octet **receptionat corect**
 - in urma confirmarilor primite emitatorul deplaseaza fereastra (o **gliseaza**)



Administrarea conexiunii TCP

Controlul fluxului folosind fereastra glisanta – situatii

- **Pachetul 2 s-a pierdut**
 - emitatorul nu receptioneaza ACK 2, deci fereastra ramine pe pozitia 1
 - receptorul nu primeste pachetul 2, si nu va confirma inca pachetele 3,4,5
 - cronometrul emitatorului va semnaliza timpul de asteptare a confirmarii si pachetul este retransmis
 - receptorul primeste pachetul 2 si va emite confirmarea ACK 5 (secventa de pachete 2-5 au fost receptionate cu succes) => fereastra glisanta se deplaseaza cu patru pozitii dupa receptionarea ACK 5
- **Pachetul 2 a ajuns la destinatie dar confirmarea s-a pierdut**
 - Emitatorul nu primeste ACK 2, dar primeste ACK 3 (toate pachetele pina la 3 au ajuns cu succes) => emitentul deplaseaza fereastra glisanta la pachetul 4

Administrarea conexiunii TCP

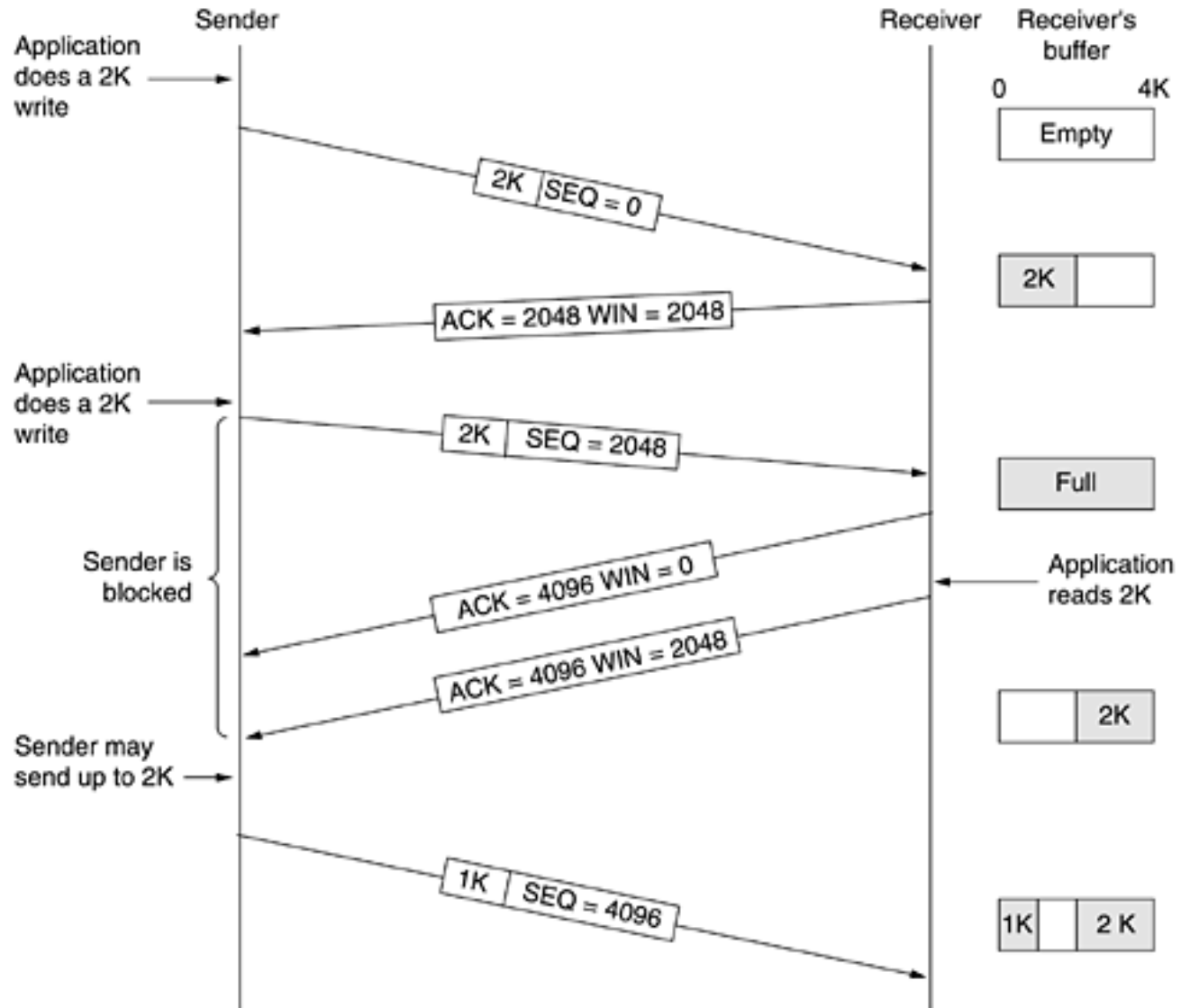
Ferastra glisanta asigura:

- Transmisie sigura
- Folosirea mai buna a latimii de banda => o mai mare viteza de transmisie
- Controlul transmisiei (receptorul poate intarzia confirmarea datelor, cunoscind memoria libera de care dispune si dimensiunea ferestrei de comunicare)

Administrarea conexiunii TCP

Exemplu:

Politica TCP de
transmisie a
datelor si
managmentul
ferestrei
TCP



[conform Computer
Networks, 2010 –
Andrew S. Tanenbaum,
et.al.]

Administrarea conexiunii TCP

- Detectia erorilor & retransmiterea datelor
 - Fiecare segment trimis contine un numar de secventa (*Sequence Number*) indicind pozitia octetilor transmisi in cadrul fluxului de date
 - Gazda destinatar verifica numarul de secventa pentru fiecare segment (se testeaza daca anumite segmente se pierd, sunt duplicate sau nu sunt in ordine) si trimite inapoi pentru fiecare segment un numar de confirmare (*Acknowledgment Number*), specificind numarul de secventa al urmatorului octet care se astepta a fi receptionat
 - Segmentele pierdute sunt detectate folosindu-se un timer de retransmisie a datelor
 - Pentru detectarea erorilor se utilizeaza si *checksum*-uri

Administrarea conexiunii TCP

- Resetarea conexiunii

- Uneori conditii anormale forteaza aplicatiile sau software-ul de retea sa distruga conexiunea
- Pentru resetarea conexiunii, o parte a comunicarii initiaza terminarea, trimittind un segment cu bitul **RST** setat
- Cealalta parte abandoneaza conexiunea, fara a se mai transmite eventuale date ramase netransmise
- Transferul in ambele directii este oprit, *buffer*-ele sunt golite

Administrarea conexiunii TCP

- Fortarea transmiterii datelor
 - TCP poate divide fluxul de date in segmente de dimensiuni diferite de pachetele vehiculate de aplicatii => eficienta transmisiei
 - Uneori intervine situatia de a transmite datele fara a se mai astepta umplerea buffer-elor (e.g., aplicatii interactive)
 - Fortarea transmiterii se realizeaza prin **push**: se seteaza bitul **PSH** si se forteaza transmiterea segmentelor, indiferent de starea de umplere a *buffer*-elor

Administrarea conexiunii TCP

- Includerea conexiunii

- Conexiunile TCP fiind full-duplex, cand o aplicatie semnaleaza ca nu mai exista date de trimis, TCP va inchide conexiunea doar intr-o directie
 - Partenerul de comunicatie poate expedia un segment TCP cu bitul **FIN** setat; confirmarea semnifica ca sensul respectiv de comunicare este efectiv oprit
- Conexiunea este desfiintata cand ambele directii au fost oprite

TCP

- **Automatul finit TCP**

- Modeleaza comportamentul protocolului
- Starile sunt utilizate la managementul conexiunii

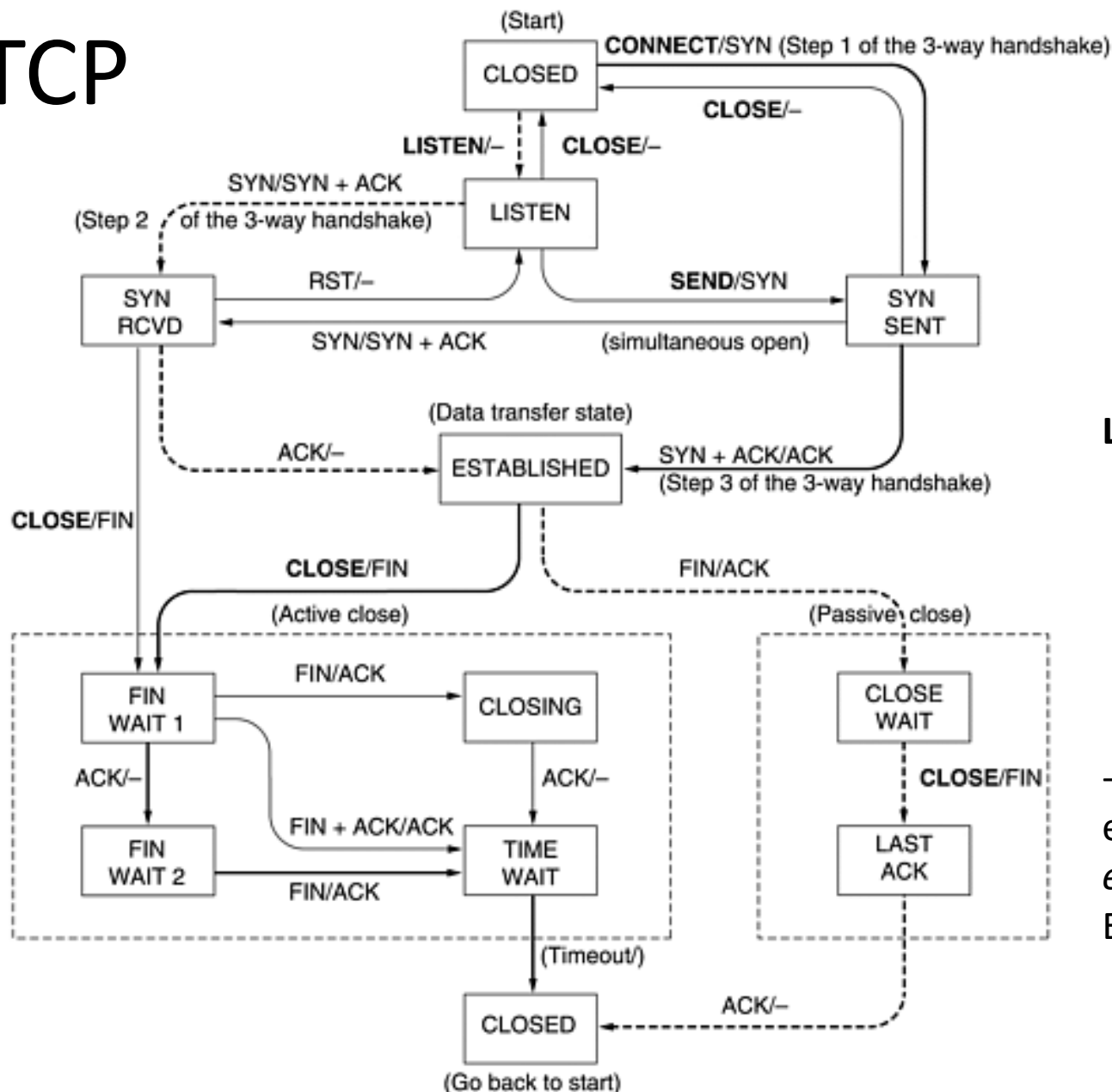
State	Description
CLOSED	No connection is active or pending
LISTEN	The server is waiting for an incoming call
SYN RCVD	A connection request has arrived; wait for ACK
SYN SENT	The application has started to open a connection
ESTABLISHED	The normal data transfer state
FIN WAIT 1	The application has said it is finished
FIN WAIT 2	The other side has agreed to release
TIMED WAIT	Wait for all packets to die off
CLOSING	Both sides have tried to close simultaneously
CLOSE WAIT	The other side has initiated a release
LAST ACK	Wait for all packets to die off

Figura. Starile utilizate in automatul cu stari finite pentru controlul conexiunii TCP

[conform Computer Networks, 2010 – Andrew S. Tanenbaum, et.al.]

TCP

Automatul finit TCP



Legenda:

- _____ Client
- Server
- _____ Evenimente mai puțin obisnuite

-Fiecare linie este etichetata cu o pereche *eveniment/actiune*
Exemplu: ACK/-

[conform Computer Networks, 2010 – Andrew S. Tanenbaum, et.al.]

TCP

- Automatul finit TCP

- Stabilirea conexiunii:
 - **CLOSED** – din aceasta stare se poate cere o deschidere activa (se trece in **SYN_SENT**) sau pasiva (**SYN_RCVD**)
 - **LISTEN** – se poate trimite o cerere de conexiune activa (se trece in **SYN_SENT**) ori pasiva (**SYN_RCVD**)
- Conexiune stabilita:
 - **ESTABLISHED** – poate incepe transmisia de date (din aceasta stare se poate trece in **CLOSE_WAIT** sau **FIN_WAIT_1**)
- Deconectarea initiata de procesul partener
 - **CLOSE_WAIT, LAST_ACK, CLOSE**
- Stari ce intervin in procesul de deconectare
 - **FIN_WAIT_1, FIN_WAIT_2, CLOSING, TIME_WAIT**

TCP

Exemplu:
netstat -t

```
adria@thor:~/html/teach/courses/net$ netstat -t
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp      0      0 thor.info.uaic.ro:smtp 186.122.246.108:33374   SYN_RECV
tcp      0      0 localhost:60000         localhost:45740         ESTABLISHED
tcp      0      0 thor.info.uaic.ro:imaps mail.traveltech.c:55156 ESTABLISHED
tcp      0      0 localhost:45740         localhost:60000         ESTABLISHED
tcp      0      0 thor.info.uaic.ro:imaps 81.253.57.112:51462     ESTABLISHED
tcp      0      0 thor.info.uaic.ro:smtp fenrir.info.uaic.:59806 TIME_WAIT
tcp      0      0 thor.info.uaic.ro:imaps ia.info.uaic.ro:51058   ESTABLISHED
tcp      0      0 thor.info.uaic.ro:imaps 77-58-250-39.dcli:56424 ESTABLISHED
tcp      0      0 thor.info.uaic.ro:59605 fenrir.info.uaic.ro:ssh ESTABLISHED
tcp      0      0 localhost:57572         localhost:spamd         TIME_WAIT
tcp      0      0 thor.info.uaic.ro:pop3s 79-112-42-154.iasi:2019 ESTABLISHED
tcp      0      0 thor.info.uaic.ro:pop3s info-c-117.info.ua:1302 ESTABLISHED
tcp      0      0 thor.info.uaic.ro:ssh   mail.traveltech.c:52266 ESTABLISHED
tcp      0      0 thor.info.uaic.ro:pop3s info-c-59.info.ua:50149 ESTABLISHED
tcp      0      0 thor.info.uaic.ro:imaps mail.traveltech.c:55152 ESTABLISHED
tcp      0      0 localhost:58053         localhost:10025         TIME_WAIT
tcp      0      0 thor.info.uaic.ro:imaps info-c-70.info.uai:1266 ESTABLISHED
tcp      0      0 thor.info.uaic.ro:imaps info-c-91.info.uai:4285 ESTABLISHED
tcp      0      0 192.168.103.2:39107    pdc:65022              ESTABLISHED
tcp      0      0 thor.info.uaic.ro:pop3s info-c-59.info.ua:55896 ESTABLISHED
tcp      0      0 thor.info.uaic.ro:imaps 77-58-250-39.dcli:56476 ESTABLISHED
tcp      0      0 192.168.103.2:39082    pdc:65022              ESTABLISHED
tcp      0      0 thor.info.uaic.ro:smtp  main.dntis.ro:52854     ESTABLISHED
tcp      0      0 thor.info.uaic.ro:imaps ws70-228.unine.ch:35907 ESTABLISHED
tcp      0      0 thor.info.uaic.ro:imaps info-c-70.info.uai:1364 ESTABLISHED
tcp      0      0 thor.info.uaic.ro:ssh   ia.info.uaic.ro:40542   ESTABLISHED
tcp      0    396  thor.info.uaic.ro:ssh   79-112-21-031.ias:54540 ESTABLISHED
tcp      0      0 thor.info.uaic.ro:imaps ia.info.uaic.ro:39325   ESTABLISHED
tcp      0      0 thor.info.uaic.ro:pop3s 79-112-42-179.iasi:1146 ESTABLISHED
tcp      0      0 thor.info.uaic.ro:imaps 81.253.57.112:51461     ESTABLISHED
```

TCP

- Exemple de utilizari ale TCP:
 - Majoritatea protocoalelor de aplicatii:
 - HTTP
 - TELNET
 - SMTP
 - SSH
 - ...

TCP versus UDP

- Ambele se bazeaza pe IP, utilizeaza porturi
- Unitatea de transmisie
 - TCP -> Segment TCP
 - UDP -> Pachet UDP

TCP versus UDP

- UDP ofera servicii minimale de transport (efort minim de transmisie)

TCP ofera servicii orientate-conexiune, full-duplex, sigure – pentru transportul fluxurilor de octeti (-> mecanism complex de transmisie)

- Utilizarea TCP sau UDP depinde de aplicatie:
e-mail, transfer de fisiere, operare in timp-real, transmisii multimedia in timp real, *chat*, ...

Rezumat

Nivelul Transport

- Preliminarii
- UDP (User Datagram Protocol)
- TCP (Transmission Control Protocol)
- TCP versus UDP

Bibliografie

- Andrew S. Tanenbaum, David J. Wetherall, Computer Networks (5th Edition), ISBN-10: 0132126958 , Publication Date: October 7, 2010
- A. Tanenbaum, Computer Networks. 4th Edition. Prentice Hall. 2003
- James F. Kurose, Keith W. Ross; Computer Networking: A Top-Down Approach (5th Edition), ISBN-10: 0136079679
- Lydia Parziale, David T. Britt, Chuck Davis, Jason Forrester, Wei Liu, Carolyn Matthews, Nicolas Rosselot , IBM – TCP/IP Tutorial and Technical Overview, 2006
- Tamara Dean, Network +Guide to Networks (Editia 5), ISBN-10: 1-423-90245-9, 2009



Intrebari?