Capa de Transport

Maria dels Àngels Cerveró Abelló

10 de novembre de 2015

1 Introducció

La Capa de Transport s'encarrega de rebre les dades des de la Capa d'Aplicació, les segmenta, les encapsula i les envia a la Capa d'Internet, on seran preparades per a ser enrutades. La seva PDU (*Protocol Data Unit*) s'anomena **Segment**. En el sentit contrari, la Capa de Transport també és qui s'engarrega de rebre els Segments des de la Capa d'Internet, desencapsular-ne les dades i entregar-les a l'aplicació correcta per tal que puguin ser tractades per la Capa d'Aplicació.

També podem dir que la Capa de Transport s'encarrega de la transferència punt a punt de les dades de l'aplicació.

Funcions bàsiques

1. Permetre que múltiples aplicacions d'un mateis dispositiu es puguin comunicar amb aplicacions d'un altre dispositiu alhora ⇒ Segmentació.
2. Si és necessari, assegurar que les dades arriben de manera confiable i ordenada al destí.
3. Gestionar possibles errors.

2 Funcionalitats

Grosso modo, la Capa de Transport és l'encarregada de segmentar les dades en el dispositiu d'origen i de tornar-les a unir correctament un cop hagin arribat al destí. No obstant això, les seves funcions, de manera més detallada, són:

1. Seguiment de les Converses Individuals: la Capa de Transport és la que s'engarrega de mantenir el fluxe de comunicació entre les aplicacions d'origen i les de destí. Més endavant es veurà que, relacionat amb aquesta funció, la Capa de Transport pot enviar missatges de *missatge rebut*, *connexió preparada*, etc.
2. Segmentació de les Dades: la capa de Transport és l'encarregada de segmentar les dades de la Capa d'Aplicació en trossos que siguin fàcils d'enviar a través de la xarxa. A més a més, encapsula cada segment de dades conjuntament amb una capçalera que conté les dades necessàries per a identificar les aplicacions que s'estan comunicant i l'ordre del segment en particular.

3. Reacoblament dels Segments: tal i com hem dit abans, la Capa de Transport utilitza les dades de la seva capçalera per

   (a) trobar l'aplicació a la que ha d'entregar les dades;

   (b) tornar a unificar els segments per tal de reconstruir les dades originals emeses per l'aplicació d'origen.

4. identificació de les Aplicacions: la Capa de Transport assigna a cada aplicació un identificador anomenat port. A cada aplicació o procés d'un dispositiu que necessiti comunicarse a través de la xarxa se li assigna un número de port únic. La capçalera de la Capa de Transport contindrà els ports de les aplicacions d'origen i de destí per tal d'indicar a quins processos s'ha de redirigir un segment de dades particular.

   Per tot plegat, la Capa de Transport garanteix una mena d'abstracció entre la Capa d'Aplicació i les capes inferiors del model TCP/IP:

   (a) Cada aplicació (Capa d'Aplicació) genera un fluxe de dades per a ser enviat però no es preocupa de com es fa aquest enviament ni de com és la xarxa.

   (b) Les Capes d'Internet i d'Accés a la Xarxa s'engarreguen d'enrutar i transmetre els Segments a través de la xarxa sense tenir consciència de què existeixen múltiples aplicacions. Per aquestes capes tot són dades a enviar i només es preocupen de trobar el dispositiu de destí.

5. Satisfer els Requeriments de l'Entrega de Dades: aquesta funcionalitat està relacionada amb la QoS (*Quality of Service*). Actualment, les alicacions i dispositius que es comuniquen per la xarxa tenen diferents requeriments:

   (a) Hi ha aplicacions que necessiten que les dades arribin completes i ordenades.

   (b) Hi ha aplicacions que necessiten molta eficiència en l'entrega.

   (c) Hi ha prioritats segons els dispositius o les aplicacions.

   (d) Algunes aplicacions poden tolerar que hi hagi una petita pèrdua de dades.

   Tots aquests requeriments són atesos i solucionats per la Capa de Transport.

   Cal tenir en compte que com més elevat sigui el requeriment de confiabilitat necessari, més trànsit es crearà, ja que s'hauran d'enviar missatges confirmant que les dades han arribat bé o demanant el reenviament dels trossos perduts.

6. Separació de les Comunicacions Múltiples: aquesta funcionalitat està directament relacionada amb les funcionalitats 4 i 5. La Capa de Transport s'encarrega de gestionar el trànsit en les xarxes convergents. És a dir, permet que els segments de diverses aplicacions i, fins i tot, de diversos sistemes de comunicació (ex: text, telèfon, veu, vídeo, etc) viatgin intercalats i arribin a l'aplicació destí de manera correcta.

El bon funcionament de la Capa de Transport recau en els processos de **Segmentació** i d'**Identificació de Ports**, els quals permeten:

1. Entrellaçar i multiplexar les comunicacions, és a dir, permetre que múltiples aplicacions s'estiguin comunicant al mateix temps ⇒ Ús compartit de la xarxa

2. Gestionar múltiples comunicacions en un mateix dispositiu

3 Control de les Converses

Control bàsic

1. Amb la segmentació i la multiplexació es poden gestionar múltiples converses al mateix temps.
2. Els ports permeten identificar les aplicacions que emeten i reben els missatges

Control addicional

3. Converses orientades a connexió (establiment de sessió): la Capa de Transport permet comunicacions orientades a connexió quan es necessari. Per tal d'aconseguir-ho, s'engarrega de crear una sessió entre les aplicacions abans d'enviar les dades. Utilitza missatges de control.
4. Entrega confiable: en cas que sigui necessari, la Capa de Transport permet assegurar que totes les dades arriben a destí. Això ho aconsegueix mitjançant un connjunt de missatges que li permeten indicar que la comunicació ha estat un èxit o, per contra, que hi ha hagut pèrdua de dades. En cas que hi hagi hagut pèrdua, pot tornar a demanar els segments perduts (només els perduts) al dispositiu d'origen. Utilitza missatges de control.
5. Entrega en el mateix ordre: aquesta funcionalitat està relacionada amb la segmentació i el reacoblament de les dades.
6. Control de fluxe: en cas que els recursos (memòria) del dispositiu de destí quedin col·lapsats, la Capa de Transport és capaç d'enviar missatges al dispositiu d'origen per demanar-li que vagi més a poc a poc enviant dades. Això permet disminuir la pèrdua i retransmissió de segments. Utilitza missatges de control.

4 Comunicació confiable

La funcionalitat bàsica de la Capa de Transport és la gestió de la comunicació entre aplicacions. No obstant això, cada aplicació té uns requeriments especials i això implica l'existència de diversos protocols de transport.

Un dels requeriments més importants és la confiabilitat, és a dir, que es garanteixi que totes les dades originades per l'aplicació origen arriben correctament al seu destí.

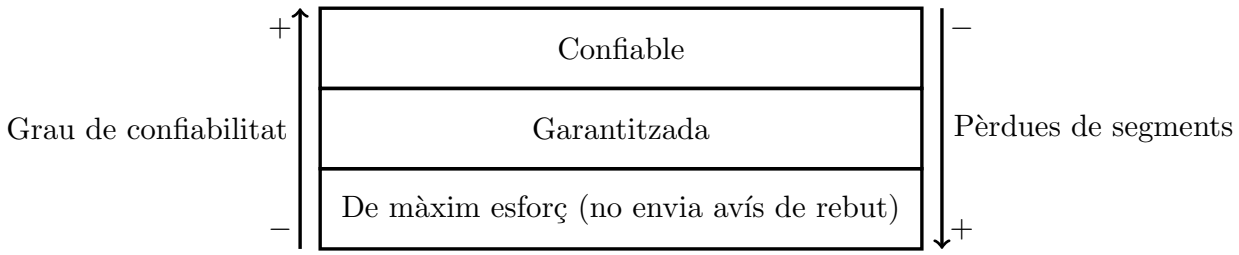
Per aconsguir la confiabilitat, la Capa de Transport s'ha d'encarregar de:

1. Fer el seguiment de les dades (mitjançant el número de seqüència dels segments).
2. Enviar l'avís de rebut (*acuse de recibo*) de les dades.
3. Retransmetre els segments perduts (aquells pels quals no s'ha rebut l'avís de rebut).

Per tant, el dispositiu de destí s'encarrega d'anar ordenant els segments que rep i, per cadascun d'ells, envia l'avís de rebut al dispositiu d'origen.

El dispositiu d'origen, al seu torn, manté un índex de tots els segments que ha enviat i de quins d'ells no ha rebut l'avís de rebut.

Tot el procés de confiabilitat genera un augment del trànsit considerable, A més a més, també es necessiten més dades de control que s'inclouen dins de la capçalera del Segment. A causa d'aquest augment del trànsit i de la capçalera, qualsevol desenvolupador d'aplicacions ha d'escollir el protocol de transport més addient depenent dels requeriments i de la càrrega de la xarxa. D'aquesta manera, tenim diversos nivells de confiabilitat:



Exemples:

1. Les consultes a bases de dades, a webs i a correus electrònics requereixen que les dades siguin entregades de manera completament confiable.
2. Un streaming de vídeo és tolerant a petites pèrdues de segments. En canvi, està molt afectat per la ineficiència que podria generar l'enviament constant de missatges d'avís de rebut i esperar a què els segments perduts arribessin. Per tant, l'streaming no necessita confiabilitat.

5 Protocols de la Capa de Transport: TCP i UDP

Els dos protocols més comuns de la Capa de Transport són

- TCP: Transport Control Protocol
- UDP: User Datagram Protocol

Tots dos protocols gestionen el control bàsic del transport, és a dir, la segmentació/multiplexació i la identificació de les aplicacions mitjançant els ports. Para però, tenen funcions específiques diferenciades.

UDP
És un protocol senzill que no està orientat a connexió (RFC 768).
Tot i que la PDU de la Capa de Transport s'anomena Segment, en el cas del protocol UDP s'utilitza la paraula Datagrama.
Envia els Datagrames amb el "màxim esforç" (no confiable).
Exemples: DNS, streaming de vídeo, VoIP.

TCP
És un protocol orientat a connexió i garangeix que les dades s'entreguin en ordre, una entrega confiable i el control de fluxe. És a dir, satisfà tant el control bàsic com el control adicional de les converses (RFC 793).
Tot plegat implica que la capçalera que necessita el TCP sigui més gran que la de l'UDP.
La seva PDU s'anomena Segment.
Exemples: web, mail, transferència d'arxius.

6 Adreçament de Ports

Com ja s'ha dit abans, la Capa de Transport identifica cada aplicació d'un dispositiu que necessita comunicar-se per xarxa mitjançant un identificador únic: el **número de port**.

Tant el Datagrama UDP com el Segment TCP contenen el port origen (identificador de l'aplicació local que genera el missatge) i el port destí (identificador de l'aplicació remota receptora del missatge).

L'assignació del número de port depèn de si el procés és una aplicació client o un servei en un servidor. Per un costat, els processos del servidor tenen assignats ports estàtics, és a dir, sempre són el mateix. Normalment, aquests ports estan predefinits pel servei corresponent, però també es poden configurar manualment. Per exemple, el port per al protocol HTTP i servei Web és el 80.

D'altra banda, el port del procés client s'assigna de manera aleatòria, vigilant que no entre en conflicte amb altres ports que ja estiguin en ús.

Quan una aplicació client vol enviar un missatge

1. Sol·licita un port.
2. Ha de conèixer el port del servei al qual vol enviar el missatge.

La Capa de Transport del client s'encarrega, doncs, de

1. Dur un registre de totes les aplicacions que tenen una comunicació oberta i, per tant, un port assignat.
2. Encapsular les dades segmentades de l'aplicació dins del Datagrama o Segment corresponent, tot indicant els ports d'origen i de destí.

Aleshores, quan el servidor enviiii la resposta, la seva Capa de Transport encapsularà aquesta resposta invertint l'ordre dels ports. És a dir, en aquest cas, el port de destí serà el de l'aplicació client que ha generat el primer missatge.

La combinació del port (Capa de Transport) i de la IP (Capa d'Internet) identifiquen de manera unívoca una aplicació dins d'un dispositiu determinat. Aquesta combinació s'anomena **socket**. És a dir,

SOCKET = IP : PORT

Una parella de sockets que consisteixi en la IP i el port d'origen (socket origen) i la IP i el port de destí (socket destí) identifiquen, de manera unívoca, una comunicació entre dos dispositius.

Hi ha vegades que la paraula socket s'utilitza com a sinònim de port.

6.1 Gestió dels ports

L'IANA és una organització d'estàndards que, entre altres responsabilitats, s'encarrega d'assignar els números de port.
IANA = Autoritat de Números Assignats d'Internet.

Els ports s'organitzen en 3 grups:

1. Ports Coneguts (0-1023): també s'anomenen Ports del Sistema i són els ports reservats per determinats serveis o aplicacions del sistema. Es tracta de serveis estàndard.

2. Ports Registrats (1024-49152): ports registrats per l'IANA que estan reservats per aplicacions del client. Per exemple, els ports de l'emule, el torrent, el postreSQL.

3. Ports Dinàmics o Privats (49152-65535): conjunt de ports que no estan registrats per l'IANA i que són d'ús privat per aplicacions customitzades, de prova, etc.

Trobareu un llistat d'aquests grups a la pàgina de la Wikipedia.

Quan la Capa de Transport necessita assignar un port a una aplicació client que demana iniciar una comunicació per xarxa, utiliza els Ports Registrats que no s'estan fent servir i els Ports Dinànimcs o Privats. És a dir, els Ports Registrats que no s'utilitzen i els Ports Dinàmics o Privats estan tots disponibles per tal que la Capa de Transport els assigni a les aplicacions que ho sol·licitin. Estan a la *pool* de ports disponibles.

Tots els ports poden ser utilitzats tant per UDP com per TCP i, de fet, hi ha serveis i aplicacions que treballen amb tots dos ports al mateix temps.

La Capa de Transport s'encarrega de què el port siguin únic per cada aplicació, és a dir, no assigna el mateix port a aplicacions diferents.

6.2 Comandes per saber quins ports s'estan utilitzant

La comanda

```
$netstat
```

que significa *network status*, dóna informació sobre

- el protocol
- l'adreça i els ports locals i remots
- l'estat de la connexió

Cal estar alerta perquè qualsevol port que estigui obert o, fins i tot, amb connexió establerta pot ser una amenaça pel sistema. Ens hem d'assegurar que totes les connexions són lícites i que no hi ha ports escoltant innecessàriament. D'altra banda, cada port consumeix recursos del sistema, per tant, com més delimitats tinguem els ports oberts, més eficient serà el nostre ordiador.

Per seguretat, cal restringir l'accés al servidor als ports amb serveis accessibles.

La comanda

```
$netstat --tcp --numeric-ports --program
```

ens indica els ports TCP que tenen connexió establerta. Ens dóna la informació en format númeric i, a més a més, ens indica quin programa està fent servir aquell port.

La comanda

```
$netstat --tcp --numeric-ports --program --listening
```

fa exactament el mateix que la comanda anterior però, en comptes de mostrar els ports amb connexió establerta, mostra els ports oberts que només estan escoltant.

La comanda

```
$netstat --tcp --numeric-ports --program --all
```

mostra tots els ports TCP, tant els que tenen connexió establerta com els que estan escoltant.

7 Segmentació i Reacoblament

Si no segmentem

1. Ocupem el medi molta estona i ningú més el pot fer servir.
2. Qualsevol error implica haver de tornar a reenviar totes les dades des del principi.
3. Els dispositius no tenen *buffers* tant grans com per emmagatzemar totes les dades d'un arxiu gran metre aquest s'està descarregant.

La segmentació ens permet

1. Un ús compartit del medi
2. Una recuperació fàcil davant dels errors
3. Adaptar-nos a les capacitats del medi i dels dispositius

Els protocols TCP i UDP tracten el procés de segmentació de manera diferent.

- TCP: un cop feta la segmentació de les dades, indica el número de cadascun d'aquests segments i, per tant, pot fer un reacoblament ordenat un cop les dades arriben al seu destí. A part d'això i, com ja hem vist, és un protocol confiable, per tant, s'assegura de què tots els Segments arribin al seu destí.
- UDP: no indica el número dels segments, per tant, el reacoblament al destí es fa seguint l'ordre d'arribada dels Datagrames. D'altra banda, és un protocol no confiable, però molt eficient i no genera tant trànsit com fa TCP. Cal tenir en compte que pot perdre dades, per tant, les aplicacions que l'utilitzin han de ser tolerants a pèrdues de dades.

8 Protocol TCP

En aquesta secció descriurem amb més detall el funcionament del protocol TCP.

8.1 Generació de Converses Confiables

Ja hem explicat que el protocol de transport que ens ofereix confiabilitat és el protocol TCP. Aquesta confiabilitat s'aconsegueix mitjançant 2 factors:

1. El número de segment de la capçalera (ens permet rastrejar els segments).
2. L'avís de rebut (ens permet saber que el destí ha rebut el segment).

En termes generals, cada cop que el destí rep un Segment, envia l'avís de rebut corresponent (després veurem que no sempre és així i que hi ha vegades que un mateix avís de rebut serveix per múltiples Segments).

El dispositiu d'origen té una llista amb tots els segments enviats. Cada cop que rep un avís de rebut, treu el segment corresponent de la llista. Si passat un temps encara té algun segment a la llista que no ha estat confirmat, el torna a reenviar. Veure la Figura 1.

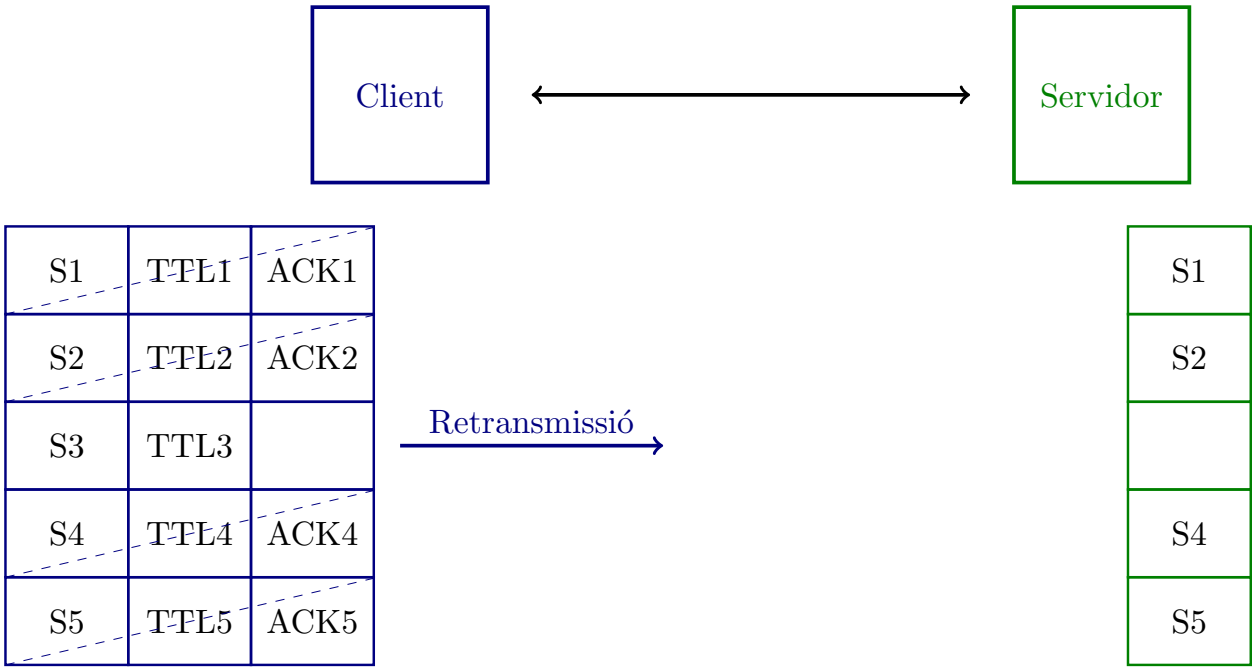


Figura 1: Llistes dels dispositius d'orgien i de destí per tal de poder fer la comunicació confiable.

Podem veure que el TCP provoca un augment de la càrrega de la xarxa i que necessita més recursos dels dispositius per tal de poder fer el seguiment dels segments.

8.2 Inici i Finalització de la Sessió TCP

En protocol TCP és un protocol orientat a connexió. Per tant, abans d'enviar qualsevol dada ha d'obrir sessió i, un cop enviades totes les dades, l'ha de tancar.

Per tal de fer-ho, fa ús dels flags de la capçalera. Alguns d'ells són:

- URG: indica que el Segment és urgent (urgent)

- ACK: indica que el Segment és un avís de rebut (acknowlegment)
- PSH: demana l'enviament del Segment (push)
- RST: indica que el Segment demana reiniciar o reconfigurar la sessió (reset)
- SYN: indica que el Segment demana fer una sincronització del número de seqüència entre els dispositius origen i destí (syncronization)
- FIN: indica que el Segment demana finalitzar la sessió

Podeu veure aquests flags a la Figura 2.

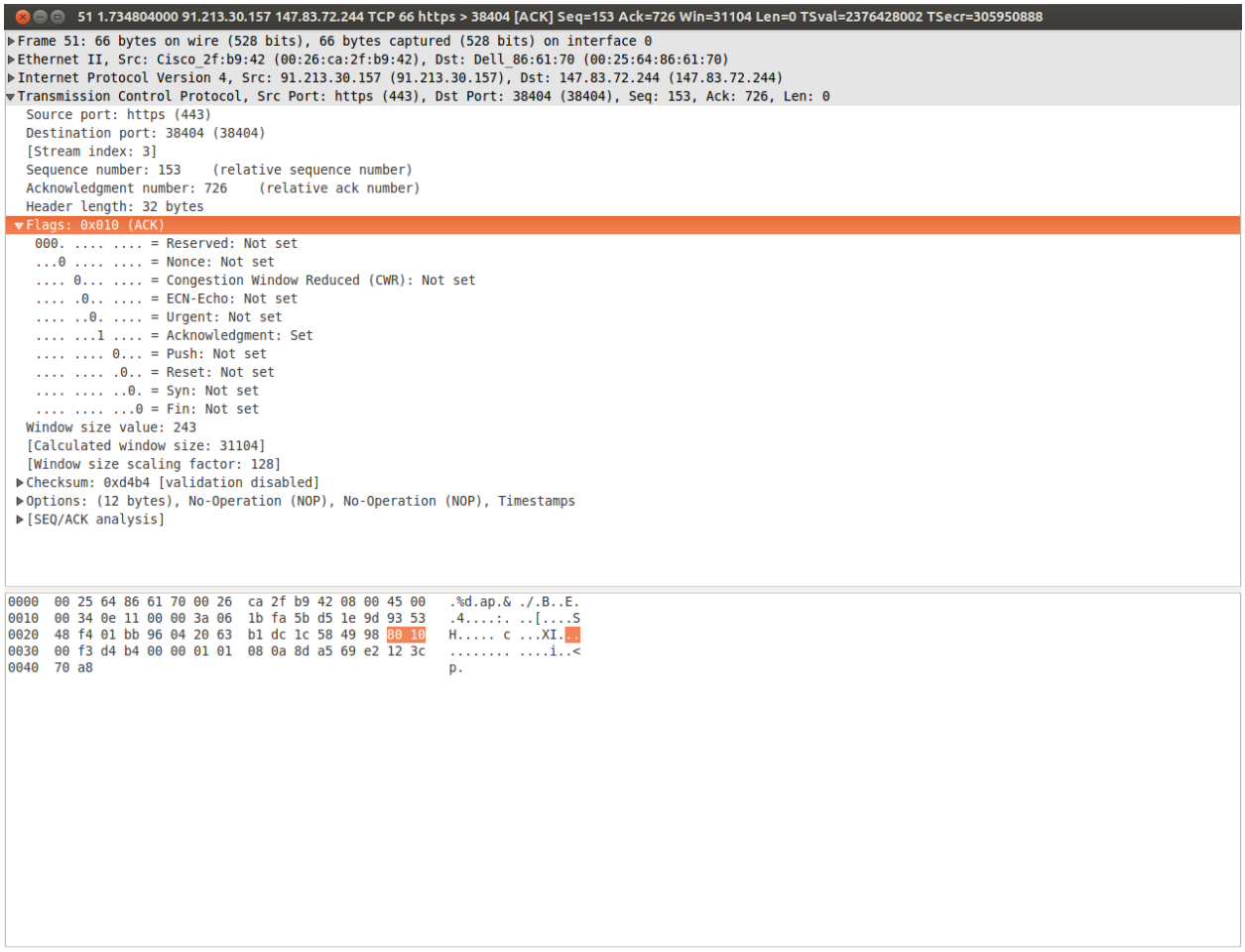


Figura 2: Captura del Wireshark per tal de visualitzar els flags de la capçalera d'una trama TCP.

8.2.1 Inici de Sessió: Intercanvi de Senyals a Tres Vies

L'inici de sessió mitjançant l'intercanvi de senyals a tres vies segueix els passos següents:

1. El client inicial la connexió enviant una trama TCP amb el flag SYN activat. El seu número de seqüència inicial (ISN_C) s'utilitza per fer la sincronització i rastrejar les trames que anirà enviant al servidor.

Aquesta primera trana actua com a sol·licitiud per a iniciar la sessió amb el servidor i comprovar que el servei realment està escoltant pel pert esperat.

2. El servidor envia una trama TCP al client amb el flag ACK activat, és a dir, fa un avís de rebut del SUN del client. A més a més, el número d'acknowledgment és igual a l'ISN_C+1.

   Amb aquesta segona trama queda oberta la sessió des del client cap al servidor. Per obrir el sentit contrari, el servidor també activa el flag SYN, fent que el seu número de seqüència inicial (ISN_S també s'utilitzi per a fer la sincronització.

   Tot plegat perlet al client saber que el servidor està llest.

3. Finalment, el client envia una trama amb el flag ACK per fer l'avís de rebut del SYN del servidor. El número de l'acknowledgment és igual a l'ISN_C+1.

   D'aquesta manera queden sincronitzat els números de seqüència. A més a més, amb aquesta tercera trama s'ha aconseguit obrir la comunicació des del servidor cap al client.

El dispositiu que inicia la sessió sempre és el client.

La Figura 3 il·lustra l'inici de sessió TCP.

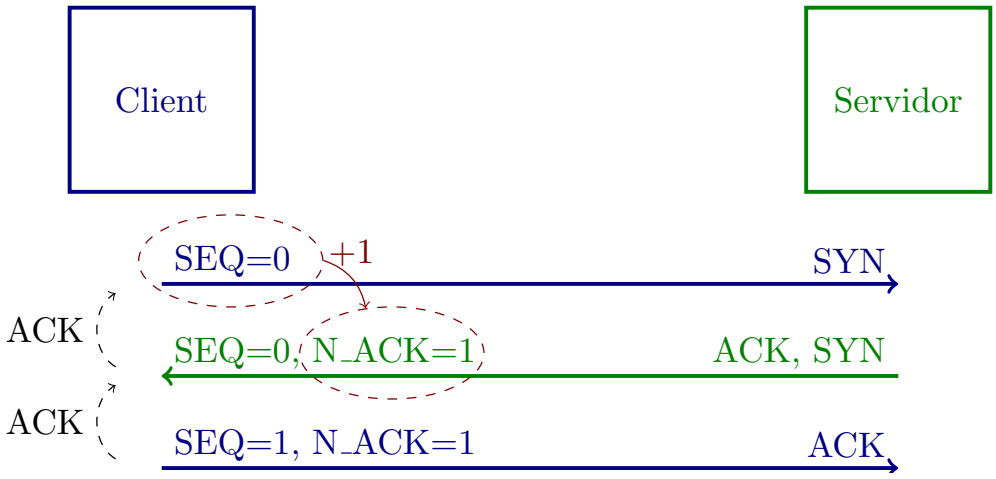


Figura 3: Inici de sessió TCP. Intercanvi de senyals a tres vies.

8.2.2 Finalització de la Sessió: Intercanvi de Senyals a Quatre Vies

Els passos per tancar la sessió són els següents:

1. Quan el client ja no té més dades per enviar, considera acabada la sessió i envia una trama amb el flag FIN activat.

2. El servidor envia la trama amb el flag ACK activat i la sessió des del client al servidor queda tancada.

3. El servidor envia la trama amb el flag FIN activat quan ja no té més dades per enviar al client.

4. El client respon amb una trama amb el flag ACK activat. En aquest moment la sessió des del servidor al client queda tancada.

Si per algun motiu s'iniciés el tancament de sessió i encara faltés algun avís de rebut per arribar, el tancament efectiu de la sessió es faria un cop rebut l'avís de rebut que faltava.

El procés de finalització es pot simplificar a tres vies en el cas que el servidor, en el moment que rep el FIN del client, tampoc té més dades per a enviar i, per tant, també vol tancar la sessió. Si és així, pot funcionar les trames 2 i 3 en una de sola activant els flags FIN i ACK alhora.

Cal tenir en compte que en l'exemple posat en la descripció del procés de finalització de sessió s'ha indicat que és el client qui comença. Ara però, comença el primer dispositiu que ja no té més dades per a enviar, ja sigui el client o el servidor.

La Figura 4 il·lustra la finalització de sessió TCP.

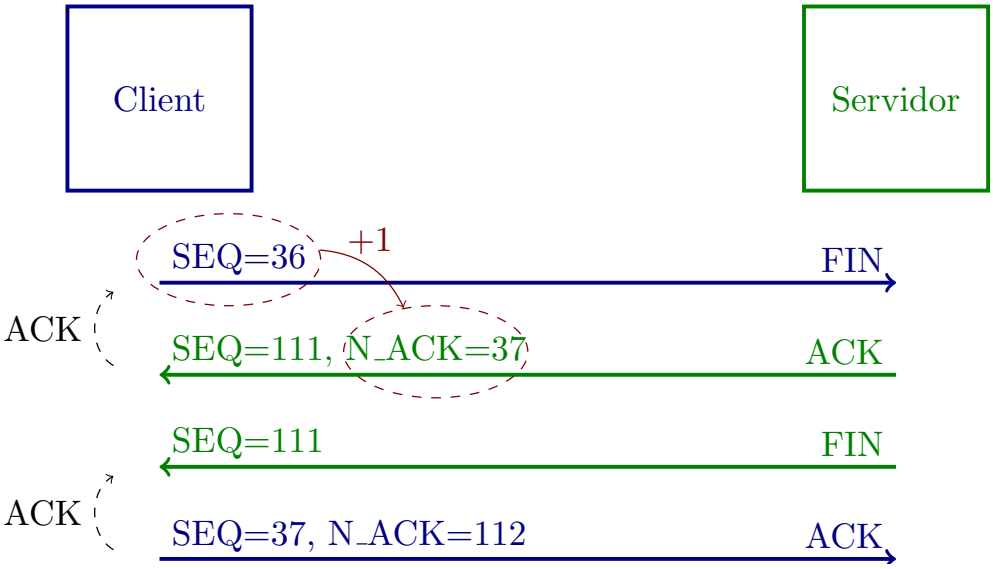


Figura 4: Finalització de sessió TCP. Intercanvi de senyals a quatre vies.

8.3 Reacoblament dels Segments TCP

El reacoblament es pot realitzar gràcies al número de seqüència que inclou la capçalera TCP.

Important: recordar que cada cop que iniciem una sessió TCP, els números de seqüència del client i del servidor se sincronitzen per tal de poder rastrejar tots els segments generats durant la comunicació. Aquesta sincronització és el que coneixem com *inici de sessió amb intercanvi de senyals a tres vies.*

8.4 Avís de rebut amb finestra

La capçalera TCP té un camp que s'anomena *Número d'Acknowledgment.* Aquest cap es refereix al següent byte que s'espera rebre. Per exemple, veure la Figura 5.

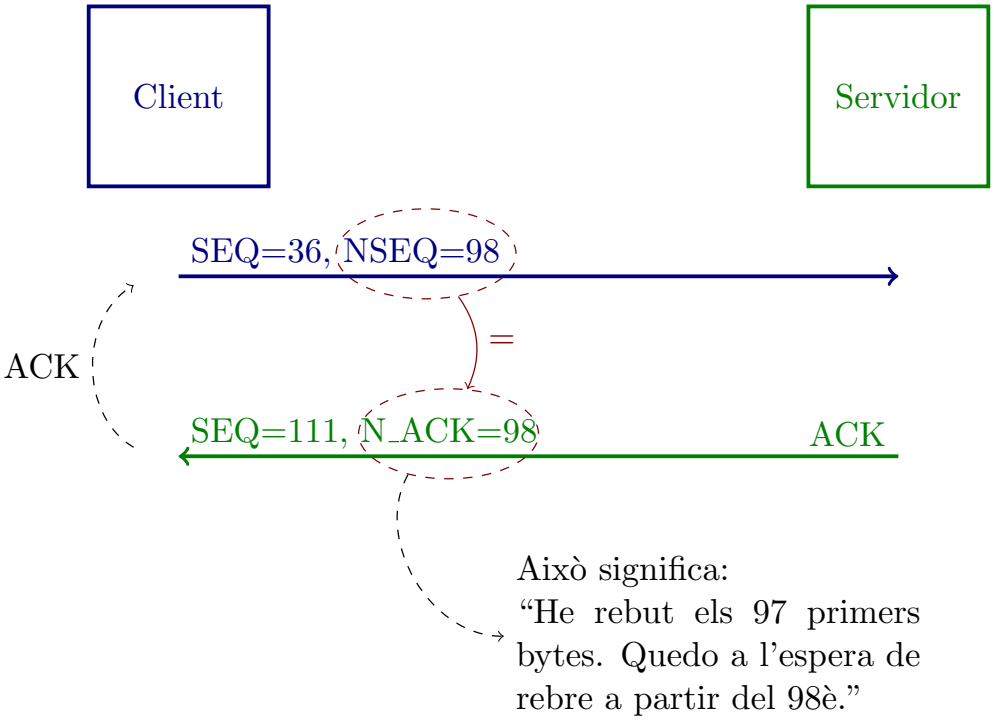


Figura 5: Trama TCP d'un ACK.

A part del camp *Número d'Acknowledgment*, les capçaleres TCP també tenen el *Valor de la Mida de Finestra* (Window Size Value, WSV) i el *Factor d'Escalat de la Finestra* (Window Size Scaling Factor, WSSF). Aquests dos valors també es sincronitzen durant l'inici de sessió i fan referència a la capacitat del buffer de recepció del dispositiu que general el missatge

$$\text{Buffer} = \text{WSV} \cdot \text{WSSF} \text{ (bytes)}$$

A mesura que el buffer es vagi omplint o buidant, el valor WSV pot anar canviant per adaptar-se a les noves circumstàncies. D'aquesta manera, aquest valor es pot utilitzar per reduir el trànsit de les trames TCP. D'aquesta manera, quan un dispositiu rep un missatge, mira el valor del WSV i sabrà quantes dades pot enviar a l'altre dispositiu sense sobrepassar la capacitat del seu buffer. Per tant, en comptes d'esperar rebre una trama TCP d'acknowledgment per cada trama que enviï ell, pot esperar a rebre'n una al final indicant que ha rebut X trames (les necessàries per omplir el buffer). Veure l'exemple de la Figura 6.

La finestra permet dues coses:

1. Gestió dels avisos de rebut i dels Segments perduts. No fa falta enviar un avís de rebut per cada trama, sinó que es pot fer en bloc. D'aquesta manera es redueix el trànsit per la xarxa.

2. Controlar el fluxe de la comunicació. Això ho veurem una mica més endavant però, per fer un breu resum, quan els recursos del dispositiu o del medi es comencen a saturar, modificar el valor de la finestra ens permet reduir el trànsit.

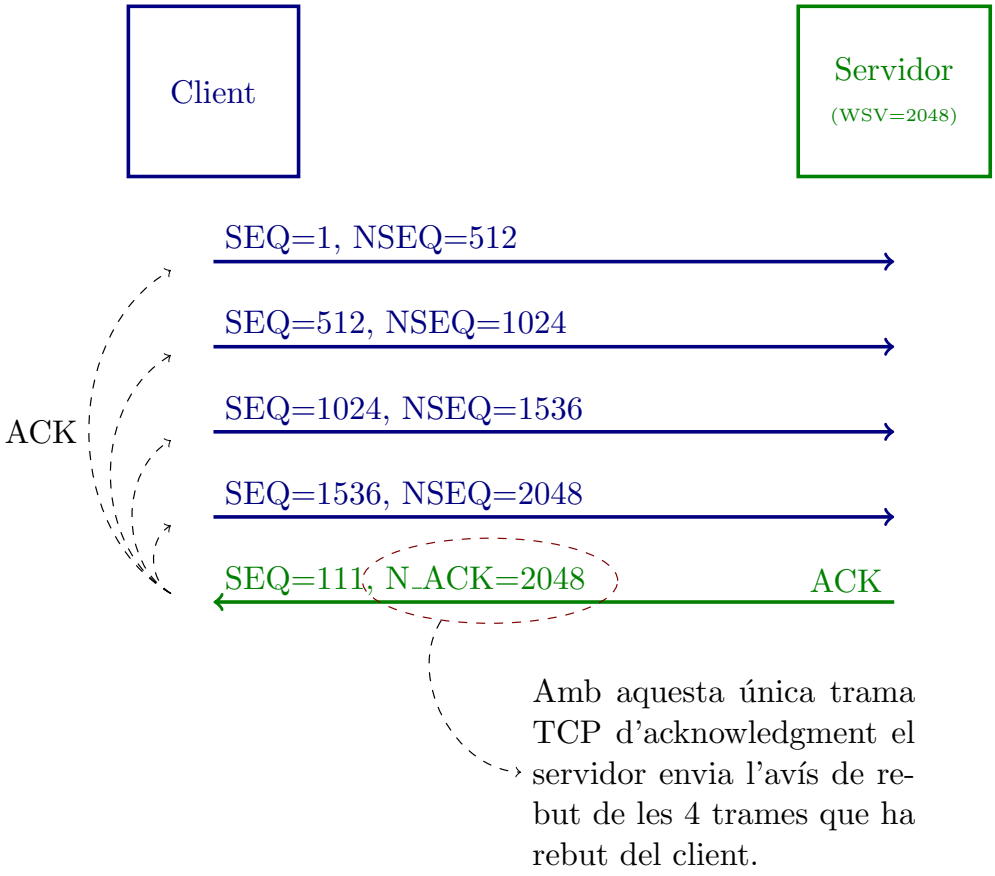


Figura 6: Avís de rebut TCP amb finestra.

8.5 Retransmissió

Normalment, TCP utilitza el valor de la finestra dels dispositius per enviar només un avís de rebut per un conjunt de trames **consecutives**. Per exemple, imaginem-nos que un client envia un conjunt de trames que contenen des del Segment 1 fins al Segment 5. No obstant això, el Servidor només rep les trames dels Segments 1, 2, 4 i 5. Com que no són tots consecutius, el servidor només enviarà l'avís de rebut per als Segments 1 i 2. Passat un temps, descartarà els Segments 4 i 5. Com que el client només haurà rebut la confirmació fins al Segment 2, tornarà a enviar els Segments 3, 4 i 5. Veure Figura 7.

Això genera més trànsit de retransmissió que si s'enviés un avís de rebut per cada Segment. Per intentar arribar a un equilibri, TCP també pot implementar els *Avisos de Rebut Selectius*, que són una mica més complexes. Aquest tipus d'avisos necessiten dos camps nous a la capçalera

1. El número d'inici de l'ACK.
2. El número de final de l'ACK (té el mateix significat que el *número d'acknowledgment* que hem estudiat fins ara.

Amb aquests dos camps, la trama TCP d'acknowledgment pot indicar des de quin byte i fins a quin byte està fent l'avís de rebut.

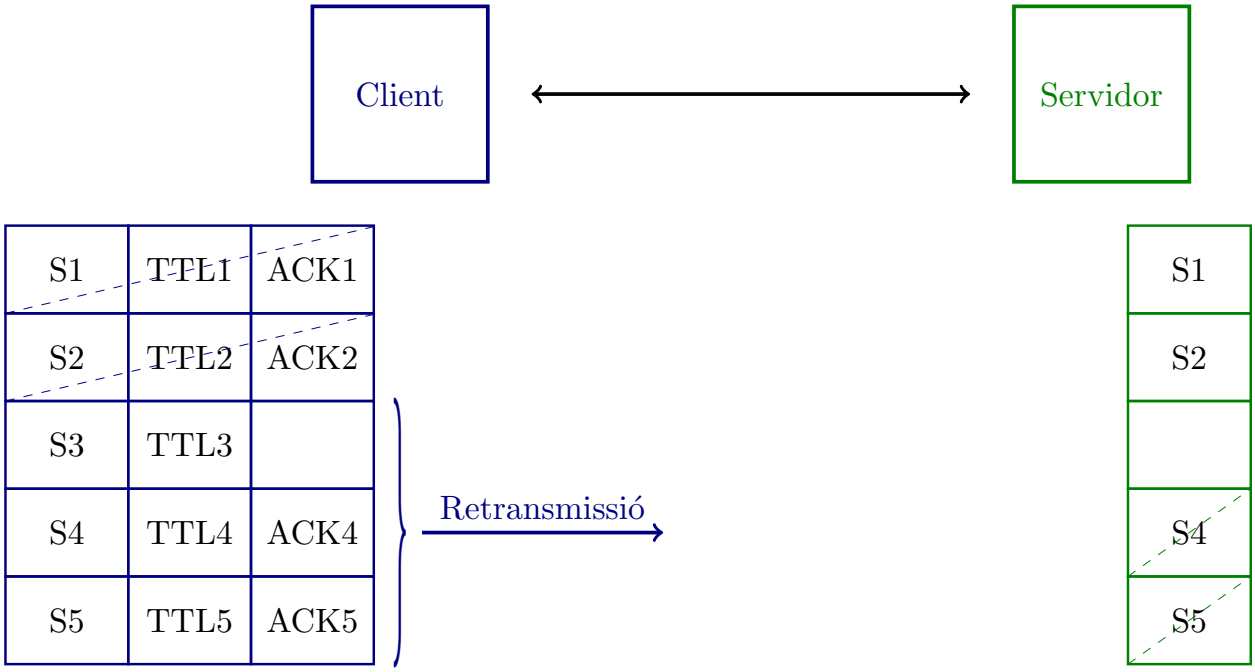


Figura 7: Retransmissió de trames TCP.

8.6 Control de Fluxe

Mitjançant el valor de la finestra WSV, el TCP és capaç de gestionar el màxim enviament de dades possible segons les capacitats del medi i dels dispositius involucrats.

El seu objectiu és:

- Reduir la pèrdua de trames i, per tant, la retransmissió
- Reduir el nombre de trames d'acknowledgment aprofitant al màxim la finestra

Quan el dispositiu emissor ha enviat suficients dades per omplir la finestra del receptor s'espera sense enviar res més fins que rep la trama d'avís de rebut. Això vol dir que el receptor ja ha buidat el seu buffer i processat els Segments i que, per tant, ja torna a tenir lloc. Alehsores, l'emissor torna a enviar dades.

A més a més, quan un dels dispositius queda saturat pot avisar a l'altre canviant la seva mida de finestra. Això també evitarà que es perdin trames. En el moment en què el dispositiu es recuperi, pot tornar a agumentar la finestra.

9 Protocol UDP

Només implementa les dues funcions de control bàsiques: la segmentació i la identificació de les aplicacions. Per tant, és un protocol que no garanteix la confiabilitat. Tot i això, les aplicacions

que utilitzen UDP poden ser confiables, sempre i quan siguin elles les encarregades d'implementar mecanismes que assegurin aquesta confiabilitat per sobre d'UDP (per exemple, incloent certs missatges de control dins de les pròpies dades de l'aplicació).

Alguns protocols i serveis de la Capa d'Aplicació que utilitzen UDP com a protocol de la Capa de Transport són: DNS, DHCP, RIP, TFTP i els jocs en xarxa.

UDP té una baixa sobrecàrrega de la xarxa

En el moment en què l'aplicació té les dades les envia. No crea cap sessió.

Reacobla els Datagrames tal i com arriben al destí, no en controla l'ordre.