NIVELUL TRANSPORT

Nivelul Transport are ca sarcină transportul datelor de la sursă la destinaţie într-un mod sigur, eficace din punctul de vedere al costurilor şi independent de reţeaua fizică utilizată. Nivelul Transport este conţinut atât în modelul TCP/IP, care este fundamental Internetului, cât şi în modelul OSI.

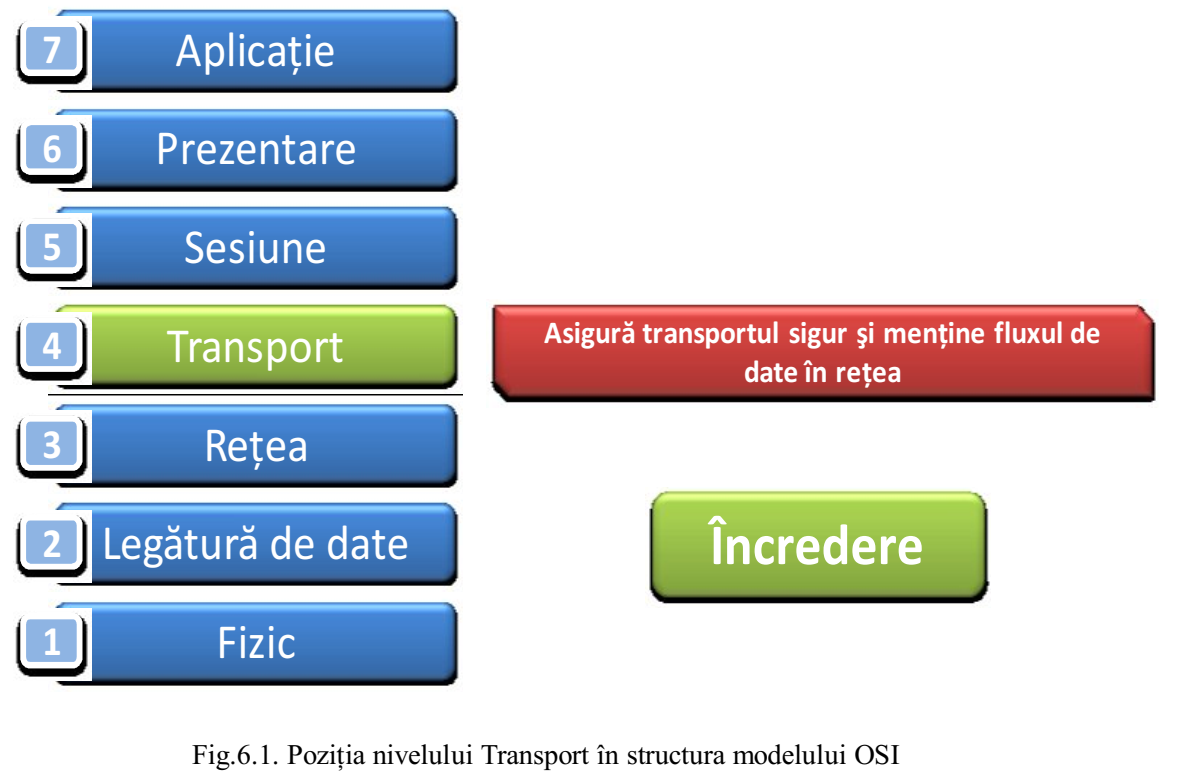


Fig.1 Nivelul Transport în structura Modelului ISO-OSI

Nivelul Transport separă nivelurile orientate pe aplicaţii (5, 6 şi 7 - destinate să asigure livrarea corectă a datelor între calculatoarele interlocutoare), de cele destinate operării subreţelei (nivelurile 1, 2 şi 3 - responsabile de deplasarea mesajelor prin reţea, stive ce pot suferi modificări de implementare fără a influenţa nivelurile superioare). Nivelul Transport administrează transmisia de date de la un computer la altul, putând asigura unele servicii, ca de exemplu: calitatea în comunicare, siguranţa liniei de transport, controlul fluxului sau detecţia şi corecţia erorilor. Una dintre funcţiile acestui nivel este de a împărţi datele în segmente mai mici pentru a fi transportate uşor prin reţea. El este proiectat astfel încât să permită conversaţii între entităţile pereche din gazdele sursă, respectiv, destinaţie. În cadrul acetui nivel sunt implementate diferite protocoale, două din cele mai cunoscute şi utilizate fiind:

● TCP (Trasmission Control Protocol) este un protocol sigur orientat pe conexiune care permite ca un flux de octeţi trimişi de pe un calculator să ajungă fără erori pe orice altă maşină din reţea. Orientarea pe conexiune nu semnifica faptul că există un circuit între computerele care comunică, ci faptul că segmentele nivelului Aplicaţie călătoresc bidirecţional între două gazde care sunt conectate logic pentru o anumită perioadă. Acest proces este cunoscut sub denumirea de packet switching. TCP fragmentează fluxul de octeţi în mesaje discrete şi transmite aceste segmente nivelului Reţea. TCP tratează totodată controlul fluxului pentru a se asigura că un emiţător nu aglomerează (congestionează) un receptor mai lent cu mai multe mesaje decât poate acesta să prelucreze.

● UDP (User Datagram Protocol), este un protocol nesigur, fără conexiuni, destinat aplicaţiilor care doresc să utilizeze propria lor secvenţiere şi control al fluxului. Protocolul UDP este de asemenea mult folosit pentru interogări rapide întrebare-răspuns, client-server şi pentru aplicaţii în care comunicarea promptă este mai importatntă decât acurateţea acesteia, aşa cum sunt aplicaţiile de transmisie a vorbirii sau a imaginilor video. Principala diferenţă între cele două protocoale ale nivelului transport (TCP şi UDP), este fiabilitatea.

**1 Funcţiile nivelului Transport**

Principalele funcţii sau responsabilităţi pentru nivelul transport, pentru a realiza conexiunea sursă – destinaţie, sunt următoarele:

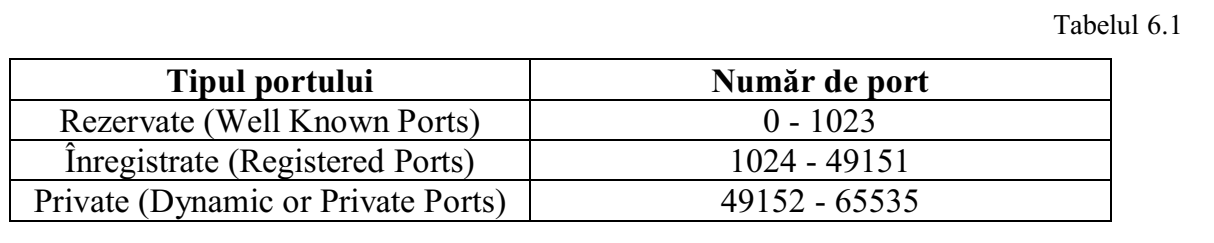
● Identificarea diferitelor aplicaţii;

Un calculator are in general o singură legatură fizică la reţea. Orice informaţie destinată unei anumite maşini (de exemplu alt calculator) trebuie să specifice obligatoriu adresa de IP a acelei maşini. Dar pe un calculator pot exista în acelaşi timp mai multe procese care au stabilite conexiuni în reţea. Prin urmare datele trimise către o destinaţie trebuie să specifice pe langă adresa logică (IP-ul) a calculatorului şi procesul căreia îi aparţine informaţia respectivă. Identificarea proceselor se realizează prin intermediul porturilor.

Un port este un număr pe 16 biţi care identifică în mod unic procesele care rulează pe o anumita maşină. Orice aplicaţie care realizează o conexiune în reţea va trebui să ataşeze un număr de port acelei conexiuni. Valorile pe care le poate lua un număr de port sunt cuprinse intre 0 si 65535 (deoarece sunt numere reprezentate pe 16 biţi).

Există trei tipuri diferite de număr de porturi - tabelul 1.

Tabelul 1



◦ Rezervate (0 - 1023) Aceste numere de port sunt rezervate unor aplicaţii binecunoscute sau standard;

◦ Înregistrate (Registered Ports) (1024 - 49151) - Aceste numere de port pot fi alocate unor aplicaţii;

◦ Private (Dynamic or Private Ports) (49152 - 65535) - Aceste numere de port sunt, în mod curent, alocate dinamic clienţilor unor aplicaţii.

Exista numere de porturi care pot fi alocate dinamic.

● Multiplexarea şi demultiplexarea datelor;

Adresarea aplicaţiilor este un exemplu de funcţionare a multiplexării, putând exista mai multe conexiuni transport pentru o singura conexiune de reţea. Folosind adresele de port, protocoalele de la nivelul Transport, multiplexează la transmisie datele venite de la mai multe aplicaţii, combinându-le într-un singur flux de date care va fi transmis. Aceleaşi protocoale, primesc datele la recepţie şi demultiplexează fluxul de date, direcţionând fiecare segment către aplicaţia sau procesul destinatar.

● Trasarea comunicaţiei individuale între aplicaţiile sursei şi respectiv destinaţiei;

Identificarea diferitelor aplicaţii are ca efect posibilitatea de a se realiza o comunicare individuală între diverse aplicaţii. Atunci cînd un proces aplicaţie doreşte să stabilească o conexiune cu o aplicaţie aflată la distanţă, el trebuie să specifice cu care proces doreşte să se conecteze. Metoda folosită în mod normal este de a defini adrese de transport la care procesele pot să aştepte cereri de conexiune. Aceste adrese sunt porturile.

● Segmentarea datelor în segmente şi administrarea fiecărui segment în parte;

Pentru a realiza comunicaţia între procese, nivelul Transport trebuie să realizeze mai multe sarcini diferite dar dependente între ele. Pentru transmisie, nivelul Transport trebuie să ţină evidenţa datelor venite de la fiecare aplicaţie şi apoi să combine aceste date într-un singur flux de date pe care să-l trimită la nivelele inferioare. De la nivelele superioare, nivelul Transport primeşte cantităţi mari de data pe care nu poate să execute operaţia de multiplexare. Pentru a rezolva această problemă, se sparge fluxul de date în segmente de dimensiuni mici care sunt mai uşor de manipulat la transmisie de către reţea. Pentru a exemplifica segmentarea, să presupunem că un computer încearca să trimită prin reţea un fişier de dimensiuni foarte mari. Transferul acestui fişier ar dura foarte mult. Dacă acest fişier (flux de date) nu ar fi segmentat, nu există nicio posibilitate ca alte terminale să folosească reţeaua pe durata de transfer al acestor date. Alte terminale ar trebui să aştepte ca fişierul să fie transmis complet înainte ca ele să înceapă să transmită. Deoarece reţeaua trebuie să poată fi folosită de multe terminale în acelaşi timp trebuie facută segmentarea (şi apoi binenţeles multipexarea sau întreţeserea segmentelor). Datele de la fiecare proces (terminal) se împart în bucaţi mici care ocupă puţin legatura pe reţea şi astfel şi celelalte terminale pot trimite aparent simultan date pe reţea (acest procedeu poartă deumirea de multiplexare în timp).

● Reasamablarea segmentelor în fluxuri de date de aplicaţii

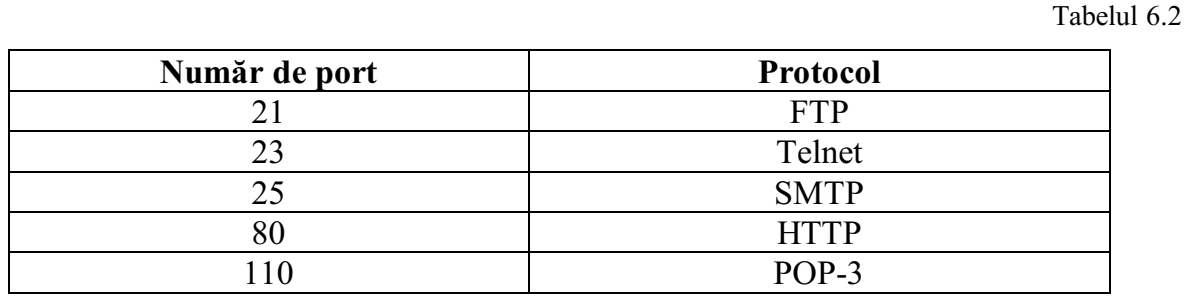
Terminalul care primeşte informatia trebuie să realizeze operaţiile inverse, adică să reasambleze datele din segmentele primite şi să refacă fluxul iniţial de date.

**2. Protocolul TCP**

TCP este protocolul principal care permite transportul sigur al datelor de la un capăt la altul al unei conexiuni într-o reţea nesigură. A fost proiectat special pentru Internet şi este larg folosit în acest scop. TCP este un protocol orientat pe conexiuni, care permite ca un flux de octeţi trimişi de un calculator să ajungă fără erori pe orice alt calculator din reţeaua Internet. Fiabilitatea comunicării prin intermediul protocolului TCP este dată de sesiunile orientate pe conexiune. Înainte ca o gazdă să trimită date către o altă gazdă folosind protocolul TCP, nivelul Transport iniţiază un proces pentru a crea o legătură cu destinaţia. Această conexiune permite urmărirea sesiunii, sau fluxului de comunicare între gazde. Acest proces se asigură că fiecare gazdă are cunoştinţă despre comunicare şi estepregătită pentru aceasta. O conversaţie TCP completă cere stabilirea unei sesiuni între gazde, în ambele direcţii. După ce sesiunea a fost stabilită, destinaţia trimite confirmări la sursă pentru fiecare segment pe care îl primeşte. Aceste confirmări formează baza de fiabilitate în cadrul sesiunii TCP. Când sursa primeşte o confirmare, se ştie că datele, pentru care s-a primit respectiva confirmare, au fost livrate cu succes şi astfel se poate încheia urmărirea acelor datele. În cazul în care sursa nu primeşte o confirmare în cadrul unei sume prestabilite de timp, se retransmit datele către destinaţie.

Cîteva dintre cele mai cunoscute aplicaţii care sunt transmise cu ajutorul protocolului TCP sunt prezentate (prin intermediul numerelor de port) în tabelul 2.

Tabelul 2



*a) Caracteristicile protocolului TCP*

Principalele caracteristici ale TCP sunt:

● Transfer de date în flux continuu - datele circulă în acelaşi timp, în ambele sensuri ale conexiunii;

● Siguranţa transmisiei - recuperează pachetele transmise cu erori, pierdute sau cu număr de secvenţă eronat;

● Controlul fluxului de date – în transferul de date dintre două procese, când aplicaţia destinaţie trimite o confirmare către emitent, se indică şi numărul permis de octeţi ce se pot recepţiona, pentru a se asigura că transmiterea rapidă de mesaje de către un emiţător, nu face ca un receptor lent să primească mai multe mesaje decât poate prelucra. În urma unui astfel de mesaj, emiţătorul îşi va dimensiona pachetele transmise la lungimea indicate de receptor;

● Multiplexarea - permite mai multor procese, care rulează pe acelaşi calculatorhost, să utilizeze facilităţile protocolului TCP simultan;

● Controlul conexiunii (fiabilitatea conexiunii) - presupune stabilirea numărului de secvenţă şi a dimensiunii ferestrei, pentru fiecare segment TCP;

● Stabilirea conexiunii.

*b) Stabilirea conexiunii*

Când două gazde comunică folosind TCP, o conexiune este stabilită înainte ca datele să poată fi transmise. După ce comunicarea este completă, sesiunile sunt închise, iar conexiunea se încheie. Mecanismele de conectare şi de sesiune activează funcţia de fiabilitate a protocolului TCP. Gazda urmăreşte fiecare segment de date în cadrul unei sesiuni şi face schimb de informaţii cu privire la ce date sunt primite de fiecare gazdă, utilizând informaţiile din antetul TCP. Pentru a stabili o conexiune, gazdele efectuează o metodă numită three-way handshake. Biţii de control din antetul TCP indică evoluţia şi starea conexiunii. Acest algoritm conţine următorii paşi prezentat în figura 2:

● Clientul iniţiator trimite un segment care conţine:

◦ o valoare de secvenţă iniţială (SEQ client = 100 ), ce serveşte ca o cerere către server pentru a începe o sesiune de comunicaţii;

● Serverul răspunde cu un segment care conţine:

◦ o valoare de confirmare (ACKserver = SEQ client +1 = 101 ), egală cu valoarea secvenţei primite plus 1.Valoarea de confirmare este una mai mare decât valoarea secvenţei, deoarece ACK este întotdeauna următorul octet aşteptat. Această valoare de confirmare permite clientului de a fi sigur că cererii lui de realizare a conexiunii i se răspunde;

◦ valoarea ei proprie de secvenţă de sincronizare (SEQ server = 300 ).

● Clientul iniţiator răspunde cu un segment care conţine; o o valoare de confirmare (ACKclient = SEQ server +1= 301 ), egală cu valoarea secvenţei primite plus 1;

◦ valoarea ei proprie de secvenţă de sincronizare plus 1 ( SEQ client + 1 = 101).

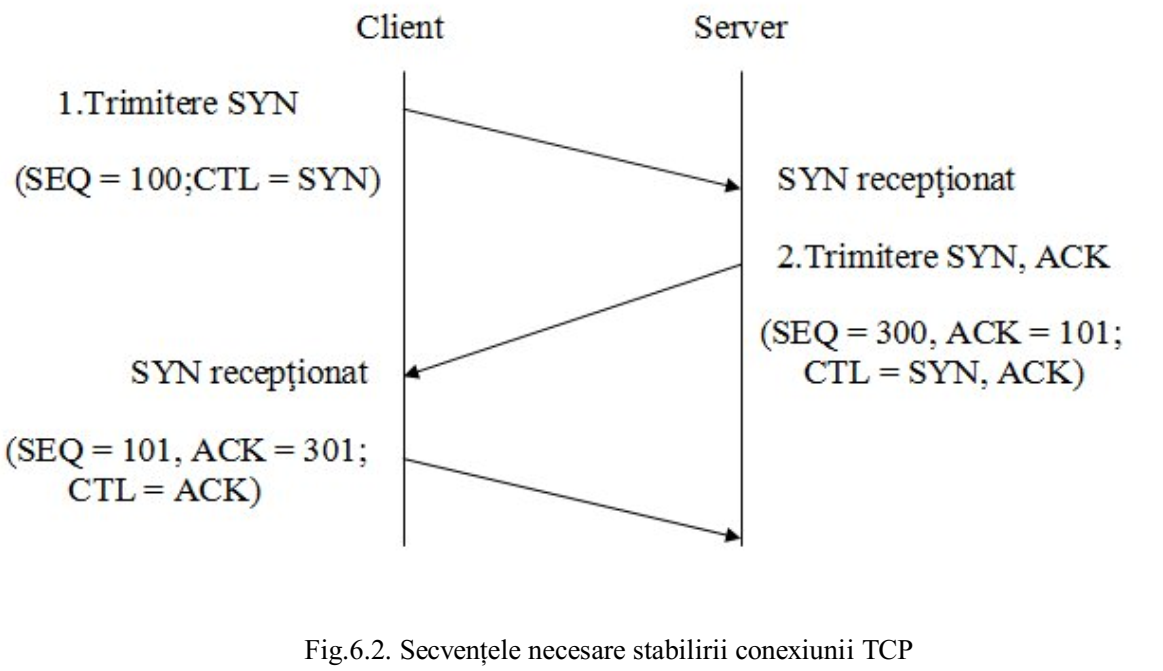


Fig. 2 Stabilirea conexiunii TCP

SYN - Sincronizează valorile de secvenţă;

SEQ - Valoarea secvenţei;

ACK - Valoarea de confirmare;

CTL - Specifică biţii de control din antetul segmentului TCP ce sunt setaţi pe 1.

*c) Antetul segmentului TCP*

În cadrul nivelului Transport după procesul de segmentare are loc împachetarea datelor. Această împachetare constă în lipirea unui antet, noile entităţi de transmisie şi de recepţie purtând numele de segmente. Un segment TCP constă dintr-un antet de 20 de octeţi (plus o parte opţională) urmat de zero sau mai mulţi octeţi de date - vezi figura 3.

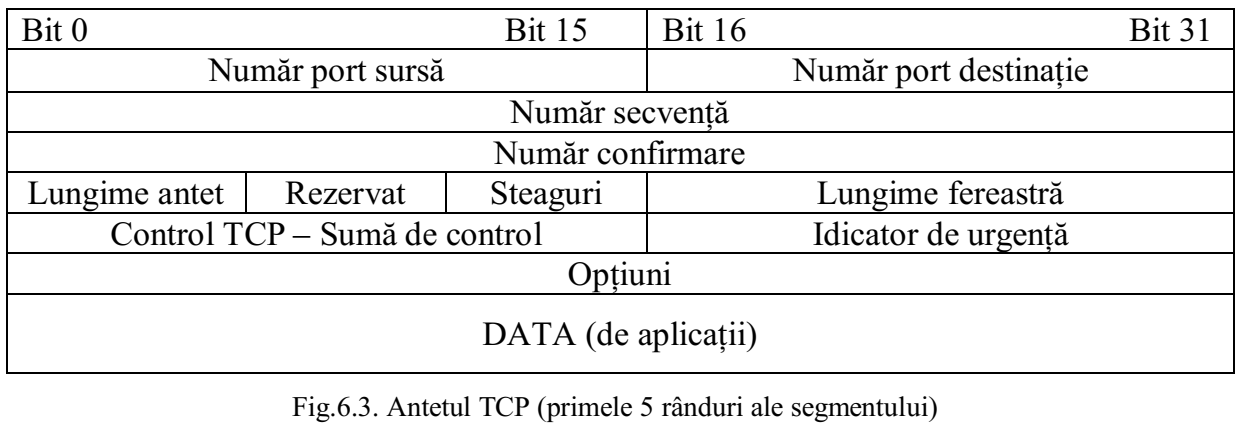


Fig 3 Antetul TCP (primele 5 rînduri ale segmentului)

Programul TCP este cel care decide cît de mari trebuie să fie aceste segmente. Dimensiunea segmentului este limitată de unitatea maximă de transfer sau MTU (Maximum Transfer Unit). Deoarece MTU este în general de 1500 octeţi (dimensiunea informaţiei utile din Ethernet) se defineşte o limită superioară a dimensiunii unui segment. Semnificaţia informaţiilor introduse în antet este următoarea:

● Număr port sursă - 16 biţi (2 octeţi) - numărul de port al celui ce face apelul;

● Număr port destinaţie - 16 biţi (2 octeţi) - numărul de port de destinaţie al celui ce este apelat;

Câmpurile Număr port sursă şi Port destinaţie identifică punctele finale ale conexiunii şi constituie totodată un identificator al conexiunii.

● Numărul de secvenţă – 32 biţi (4 octeţi) - numărul primului octet de date din cadrul segmentului curent de date;

● Numărul de confirmare – 32 biţi (4 octeţi) - valoarea următorului octet pe care sursa se aşteaptă să-l primească (şi nu a ultimului octet recepţionat în mod corect);

● Lungime antetului – 4 biţi - indică numărul de cuvinte de 32 de biţi care sunt conţinute în antetul TCP. Această informaţie este utilă, deoarece cîmpul Opţiuni este de lungime variabilă, proprietate pe care o transmite astfel şi antetului;

● Rezervat – 6 biţi - câmp neutilizat, rezervat pentru viitor;

● Steaguri (indicatori) – 6 biţi – ce au următoarea semnifibaţie:

◦ Bitul URG poziţionat pe 1 arată că Indicatorul urgent este valid;

◦ Bitul ACK pe 1 indică faptul că Numărul de confirmare este valid. Dacăeste poziţionat pe 0, segmentul în discuţie nu conţine o confirmare si câmpul Număr de confirmare este ignorat;

◦ Bitul PSH indică faptul că informaţia trebuie livrată aplicaţiei îndată ce a fost recepţionată, fără a mai fi memorată în buffere din raţiuni de eficienţă;

◦ Bitul RST este folosit pentru a desfiinţa o conexiune care a devenit inutilizabilă din cauza unur defecţiuni ale maşinilor gazdă sau din alte motive;

◦ Bitul SYN este folosit pentru stabilirea unei conexiuni. Cererea de conexiune conţine SYN = 1 si ACK = 0, iar răspunsul la o astfel de cerere este confirmată prin combinaţia SYN = 1 si ACK = 1;

o Bitul FIN este folosit pentru a încheia o conexiune.

● Lungime fereastră – 16 biţi (2 octeţi) - indică numărul de octeţi, începând cu cel indicat prin numărul de confirmare, pe care cel ce trimite mesajul îi poate recepţiona;

În TCP, fluxul de control este tratat prin ferestre glisante de dimensiune variabilă.

● Suma de control – 16 biţi (2 octeţi) - indică suma câmpurilor de antet şi date, calculată pentru verificare;

● Indicator de urgenţă – 16 biţi (2 octeţi) - permite identificarea poziţiei unor date de urgenţă, în cadrul protocolului TCP;

● Optiuni – 32 biţi (4 octeţi) - a fost proiectat pentru a permite adăugarea unor facilităţi suplimentare neacoperite de antetul obişnuit. Cea mai importantă opţiune este aceea care permite fiecărei maşini să specifice încărcarea maximă de informaţie utilă TCP pe care este dispusă să o accepte;

● Date - Datele protocolului nivelului superior;

*d) Reasamblarea în ordine a segmentelor*

În procesul de transmitere a informaţiei există posibilitatea ca segmentele să ajungă la destinaţie în cu totul altă ordine faţă de cea în care au fost trimise. Pentru ca mesajul original să fie înţeles de receptor, segmentele sunt reasamblate în ordinea iniţială. Sunt alocate numere de secvenţă în antetul fiecărui segment. În timpul instalării sesiunii, un număr de secvenţă iniţial (ISN) este setat. Acest număr de secvenţă iniţial reprezintă valoarea de pornire a octeţilor pentru această sesiune, care vor fi transmişi la receptoar. În timp ce datele sunt transmise în timpul sesiunii, numărul de ordine este incrementat cu numărul de octeţi ce au fost transmişi. Această urmărire a octeţilor de date permite fiecărui segment să fie unic identificat şi recunoscut. Segmentele ce lipsesc pot fi identificate foarte uşor. Numerele de ordine ale segmentelor ajută la creşterea fiabilităţii prin indicarea modului de reasamblare şi reordonare a segmentele primite, aşa cum se prezintă în figura 4.

Procesul de primire cu protocolul TCP aşează datele dintr-un segment într-un buffer de primire. Segmentele sunt plasate în ordinea corectă a numărului de ordine şi se transmit mai departe la nivelul aplicaţie atunci când acestea sunt reasamblate. Orice segment care soseşte cu un număr de secvenţă diferit de cel aşteptat este reţinut pentru prelucrare ulterioară. Apoi, atunci când ajung segmentele cu octeţii lipsă, aceste segmente reţinute sunt prelucrate.

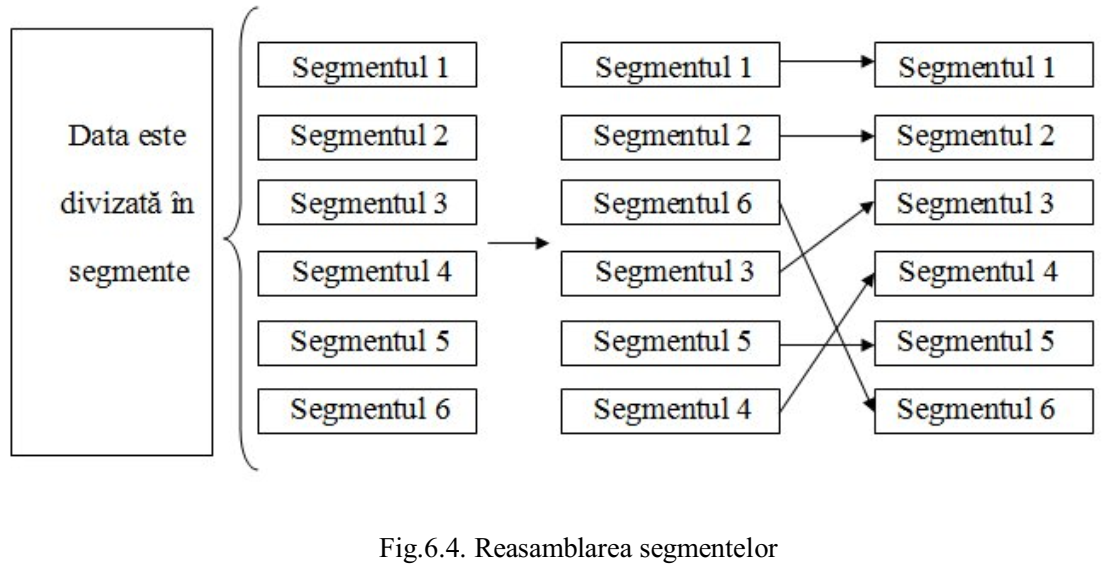


Fig. 4 Reasamblarea segmentelor

*e) Controlul congestiei în TCP*

Una dintre funcţiile protocolului TCP este să se asigure că fiecare segment ajunge la destinaţie. Serviciile TCP de la destinaţie confirmă datele pe care le-a primit de la aplicaţia sursă. Valoarea de secvenţă a antetului de segment şi numărul de confirmare sunt folosite împreună pentru a confirma primirea de octeţi de date conţinute de segmente. Numărul de ordine este numărul relativ de octeţi care au fost transmişi în această sesiune, plus 1 (care este numărul primului octet de date din segmentul curent). TCP foloseşte numărul de confirmare în segmentele trimise înapoi la sursă pentru a indica octetul următor în această sesiune, pe care receptorul se aşteaptă să-l primească. Aceasta se numeşte confirmare (acknowledgement). Sursa este astfel informată că destinaţia a primit toţi octeţii în acest flux de date de până la, dar nu inclusiv, octetul indicat de numărul de confirmare. Este de aşteptat ca dispozitivul ce trimite, să trimită un segment care utilizează un număr de ordine egal cu numărul de confirmare. Pe scurt, fiecare conexiune este de fapt un ansamblu de două sesiuni, fiecare pe o singură direcţie. Numerele de secvenţă şi numerele de confirmare sunt transmise în ambele direcţii. Cantitatea de date pe care o sursă o poate transmite înainte de a trebui să primească o confirmare, se numeşte dimensiunea (lungimea) ferestrei. Lungimea ferestrei este un câmp din antetul TCP care permite gestionarea de date pierdute şi controlul fluxului. Protocolul TCP oferă, de asemenea, mecanismele de control al fluxului de date. Controlul fluxului asistă fiabilitatea transmisiei prin TCP prin ajustarea ratei efective a fluxului de date între cele două servicii din sesiune. Atunci când sursa este informată că valoarea de date specificată în segmente este primită, se poate continua transmisia mai multor date pentru această sesiune. Lungimea ferestrei în antetul TCP precizează cantitatea de date care pot fi transmise înainte ca o confirmare să fie primită. Dimensiunea ferestrei iniţiale se determină în cursul pornirii sesiunii. Mecanismul de feedback TCP ajustează rata efectivă de transmitere a datelor la debitul maxim pe care reţeaua şi dispozitivul destinaţie îl pot suporta fără pierderi. Protocolul TCP încearcă să administreze rata de transmitere astfel încât toate datele să fie primite şi retransmisiile să fie minimizate. În figura 5 apare o reprezentare simplificată a dimensiunii ferestrei şi confirmarea corespunzătoare. În acest exemplu, dimensiunea (lungimea) ferestrei iniţiale pentru o sesiune TCP reprezentată este setată la 3000 bytes (octeţi). În cazul în care expeditorul a transmis 3000 bytes, se aşteaptă o confirmare a acestor octeţi înainte de a transmite mai multe segmente în această sesiune. Odată ce expeditorul a primit această confirmare de la receptor, expeditorul poate transmite o suplimentare de 3000 bytes.

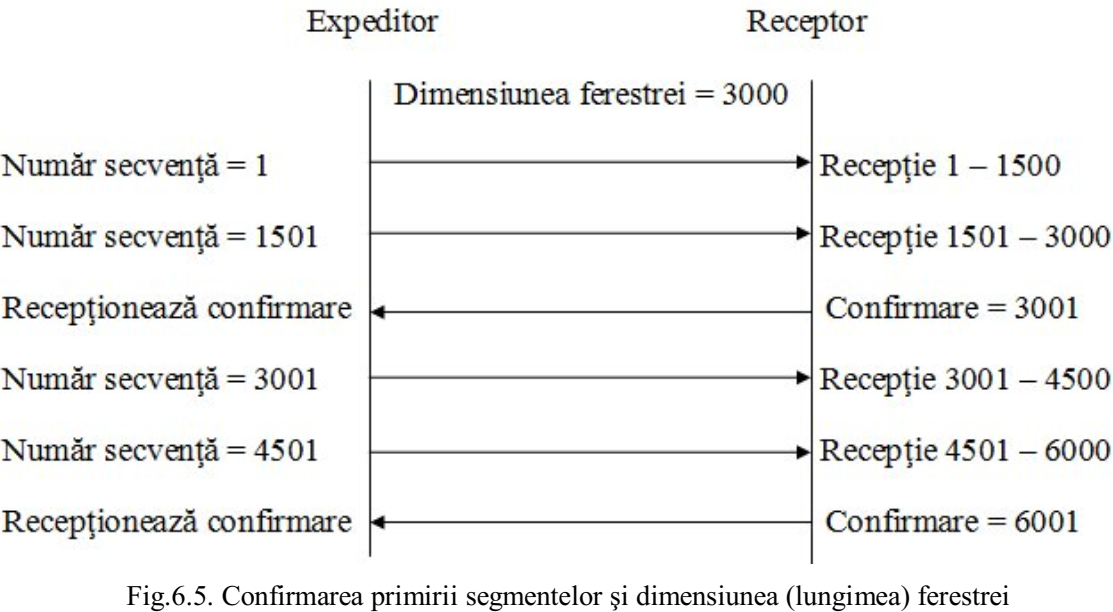


Fig. 5 Confirmarea primirii segmentelor și dimensiunea ferestrei

În timpul întârzierii, până se primeşte confirmarea, expeditorul nu va mai trimite segmente suplimentare pentru această sesiune. În perioadele când reţeaua este saturată sau resursele receptorului sunt limitate, întârzierea poate creşte. Cu cât această întârziere creşte mai mult, rata de transmitere eficientă a datelor pentru această sesiune scade.

**3. Protocolul UDP**

UDP este un protocol simplu care oferă funcţiile de bază ale nivelului transport. Protocolul UDP nu stabileşte o conexiune între sursă şi destinaţie înainte de a transmite date şi furnizează o încărcătură scăzută transportului de date, datorită faptului că antetul datagramei este mic şi pentru că nu administrează traficul reţelei.

Deoarece UDP nu este orientat pe conexiune, sesiunile nu sunt stabilite înainte ca comunicarea să aibă loc, cum se întâmplă cu TCP. UDP este declarat a fi bazat pe tranzacţii. Cu alte cuvinte, atunci când o aplicaţie are de transmis date, acesta trimite pur şi simplu acele date.

O parte din protocoalele nivelului Aplicaţie ce utilizează UDP sunt următoarele:

● DNS (Domain Name System);

● SNMP (Simple Network Management Protocol);

● DHCP (Dynamic Host Configuration Protocol);

● RIP (Routing Information Protocol);

● TFTP (Trivial File Transfer Protocol);

Unele aplicaţii, cum ar fi jocurile on-line sau VoIP (Voice over IP), pot tolera pierderea unor date. În cazul în care aceste aplicaţii utilizau TCP, mai mult ca sigur, ele prezentau mari întârzieri de timp, deoarece TCP-ul detectează pierderea de date şi retransmite acele date pierdute. Aceste întârzieri ar fi mult mai dăunătoare aplicaţiilor decât micile pierderi de date. Unele aplicaţii, cum ar fi DNS, va reîncerca pur şi simplu cererea, în cazul în care nu primesc un răspuns, şi prin urmare nu au nevoie de TCP pentru a garanta livrarea mesajului.

*a) Caracteristicile protocolului UDP*

UDP este un protocol simplu, cu puţine facilităţi. Nu realizează controlul fluxului, a erorilor, nu retransmite datagrame pierdute etc. El pur şi simplu oferă IP-ului un mijloc de multiplexare a proceselor (aplicaţiilor) folosind porturile de nivel transport. Este utilizat în transferurile scurte de date, gen întrebare – răspuns în aplicaţiile client - server. Un client trimite o cerere scurtă spre server şi aşteaptă un răspuns scurt. Dacă aceste nu vine într-un timp aşteptat, atunci repetă cererea. Un exemplu tipic de utilizare este între un client şi serverul DNS (Domain Name System) pentru aflarea adresei IP corespunzătoare unui nume de gazdă. Nu este nevoie de deschiderea unei conexiuni, nici de închidere, pentru un transfer pentru două mesaje scurte care traversează reţeaua. Atunci când mai multe datagrame sunt trimise la destinaţie, ele pot lua diferite căi şi pot ajunge în ordine greşită. UDP nu ţine cont de numerele de secvenţă cum le utilizează protocolul TCP. UDP nu are nici o modalitate de a reordona datagramele în ordinea în care au fost transmise. În figura 6 se observă acest lucru şi mai ales faptul că datagramele pierdute nu se mai retransmit.

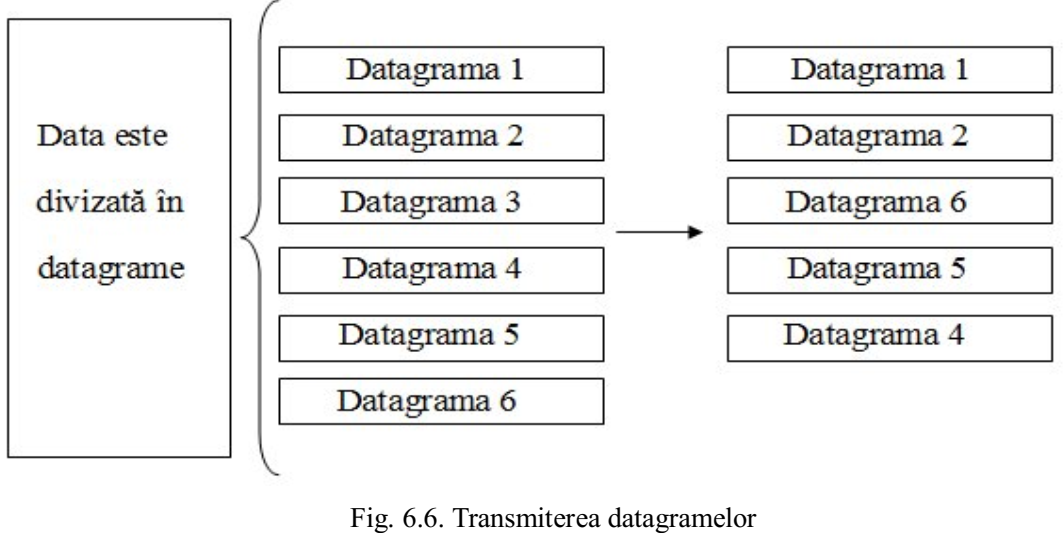


Fig. 6 Transmiterea datagramelor

Prin urmare, UDP reasamblează pur şi simplu datele, în ordinea în care acestea au fost primite şi le transmite mai departe aplicaţiei. În cazul în care secvenţa de date este importantă pentru aplicaţie, aplicaţia va trebui să identifice secvenţa corectă a datelor şi să stabilească modul în care datele ar trebui să fie prelucrate.

*b) Antetul datagramei UDP*

La fel ca în cazul utilizării protocolului TCP, după procesul de segmentare are loc împachetarea datelor. Această împachetare constă în lipirea unui antet, noile entităţi de transmisie şi de recepţie purtând numele de datagrame. PDU-ul (Protocol Data Unit) protocolului UDP este numit datagramă, deşi uneori termenii segment şi datagramă sunt folosiţi alternativ pentru a descrie un PDU de nivel transport. O datagramă UDP constă dintr-un antet de 8 de octeţi urmată de zero sau mai mulţi octeţi de date figura 7.

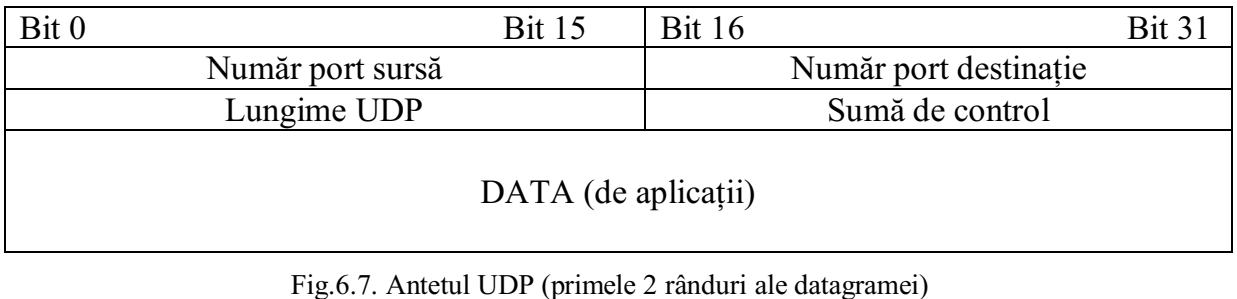


Fig. 7 Antetul UDP (primele două rînduri ale datagramei)

Semnificaţia informaţiilor introduse în antet este următoarea:

● Număr port sursă - 16 biţi (2 octeţi) - numărul de port al celui ce face apelul;

● Număr port destinaţie - 16 biţi (2 octeţi) - numărul de port de destinaţie al celui ce este apelat;

Câmpurile Număr port sursă şi Port destinaţie identifică punctele finale ale conexiunii şi constituie totodată un identificator al conexiunii.

● Lungime antetului – 16 biţi (2 octeţi) – include antetul şi datele;

● Suma de control – 16 biţi (2 octeţi) - indică suma câmpurilor de antet şi date, calculată pentru verificare;

**Comparaţie între protocoalele de nivel transport**

O comparaţie între caracteristicile celor două protocoale este oferită în tabelul 3

Tabelul 3

