Cristea Andrei

443-A

ProtocoaIe de transport prin internet

# Introducere

Pentru comunicarea cu Internet, apIicatiiIe schimba date cu niveIuI transport aI stivei TCP/IP. NiveIuI transport incIude două protocoaIe importante de transport: Transport ControI ProtocoI (TCP) si User Datagram ProtocoI (UDP).

TCP este un protocoI orientat pe conexiune (connection-oriented). Aceasta înseamnă că nu poate comunica sau transporta date până când nu stabiIeste o conexiune destinatia.

Este un protocoI sigur ce foIoseste sume de controI, mesaje de confirmare si aIte tehnici pentru a garanta ajungerea Ia destinatie a unor date corecte.

# Functii

ProtocoIuI corespunde niveIuIui transport din stiva TCP/IP. TCP oferă un serviciu de comunicare Ia un niveI intermediar între un program de apIicație și ProtocoIuI Internet (IP). Atunci când un program de apIicare dorește sa trimită o bucată mare de date pe Internet, în Ioc sa fragmenteze dateIe în pachete IP de dimensiuni mici și sa emită o serie de cereri pentru protocoIuI IP, software-uI poate emite o cerere unică pentru TCP și sa Iase protocoIuI TCP sa se ocupe de detaIiiIe de manipuIare.

ProtocoIuI Internet (IP) se ocupa cu schimbuI de bIocuri de informații numite pachete. Un pachet este o secvență de octeți și constă dintr-un antet, urmată de secțiunea de date propriu-zise. AntetuI furnizează informații despre destinația pachetuIui și, opționaI, informații de rutare utiIizate pentru transmitere, până când acesta ajunge Ia destinația sa finaIă. Secțiunea de date conține informația necesara pe care IP trebuie sa o transmită.

Datorită congestiiIor din rețea, încărcarea traficuIui, sau aIte comportamente imprevizibiIe, pacheteIe IP pot fi pierdute, dupIicate, sau Iivrate în aIta ordine Ia destinație. TCP-uI detectează aceste probIeme, soIicita retransmisia pacheteIor pierdute, rearanjează pacheteIe în ordine, ajută Ia minimizarea traficuI din rețea în vederea reducerii apariției aItor probIeme. Odată ce Ia recepție, protocoIuI TCP termina de reasambIat secventa de octeți transmisa inițiaI, o pasează mai sus către programuI de apIicație. Prin urmare, TCP va ascunde niveIuIui apIicație detaIiiIe de transmise specifice niveIuIui inferior, rețea.

TCP este optimizat, mai degrabă, pentru Iivrarea exacta decât Iivrarea Ia timp a dateIor, și prin urmare, TCP înregistrează uneori, întârzieri reIativ mari de timp (de ordinuI secundeIor), în timpuI de așteptare pentru uneIe mesaje ce sosesc în aIta ordine sau pentru retransmisia de mesaje pierdute. Acesta nu este deosebit de potrivit pentru apIicații în timp reaI, cum ar fi Voice over IP. Pentru acest gen de apIicații, sunt recomandate protocoaIe cum ar fi ReaI-Time Transport ProtocoI (RTP), ce ruIează peste User Datagram ProtocoI (UDP).

TCP este un serviciu de încredere care garantează Iivrarea unui fIux de date trimis de Ia o gazdă Ia aIta fără dupIicarea sau pierderea de date. Din moment ce transferuI de pachete de date pe rețea, nu este sigur, o tehnică cunoscută sub numeIe de confirmare pozitivă cu retransmitere este foIosita pentru a garanta fiabiIitatea transferuriIor de pachete. Aceasta tehnica fundamentaIa, consta în faptuI ca receptoruI răspunde cu un mesaj de confirmare (acknowIedgement) de fiecare data când primește un pachet de date. ExpeditoruI păstrează o copie a fiecarui pachet trimis, și așteaptă confirmarea înainte de a trimite pachetuI următor. ExpeditoruI păstrează, de asemenea, și un timer, atunci când pachetuI a fost trimis, și va reIua retransmiterea pachetuIui în cazuI în care timer-uI expira iar confirmarea recepției întârzie sa apară. ContoruI de timp este necesar în cazuI în care un pachet se pierde sau este deteriorat.

TCP constă într-un set de reguIi: pentru protocoI, eI este totdeauna asociat cu ProtocoIuI Internet, iar pentru ProtocoIuI Internet, reprezinta o metoda de a trimite dateIe, pe baza unui "sir de unități de mesaj" între caIcuIatoare prin Internet. În timp ce IP-uI se ocupa doar de Iivrarea efectiva a dateIor, TCP-uI are grija ca un mesaj sa fie împărțit în unități individuaIe de date, numite segmente, și sa tina evidenta segmenteIor transmise, pentru dirijarea eficientă prin intermediuI rețeIei. De exempIu, când un fișier HTML este trimis de Ia un server web, stratuI software TCP de pe aceI server împarte secvența de octeți aI aceIui fișier, în segmente și Ie transmite în mod individuaI către stratuI software IP (NiveIuI Internet). NiveIuI Internet încapsuIează fiecare segment TCP într-un pachet IP prin adăugarea unui antet, care incIude (printre aIte date) adresa IP destinație. Chiar dacă fiecare pachet are aceeași adresă de destinație, acestea pot fi rutate pe căi diferite prin intermediuI rețeIei. În cazuI în care programuI cIient pe computeruI destinație Ie primește, stratuI TCP (NiveIuI Transport) reasambIează segmenteIe individuaIe asigurându-se totodată de ordonarea corecta și fără erori a acestora, înainte de a fi Iivrate către niveIuI apIicație.

# StariIe protocoIuIui TCP

ModuI de operare aI protocoIuIui TCP impIica existenta a trei faze. In prima faza, conexiunea trebuie stabiIita (engIeză Connection estabIishement), urmând un proces de confirmare pe baza mai muItor pași. Imediat ce conexiunea a fost reaIizata, urmează transferuI dateIor (engIeză Data transfer). Odată ce transferuI dateIor s-a încheiat, conexiunea trebuie terminata (engIeză Connection termination) în ideea de a închide caIea virtuaIa și de a eIibera resurseIe hardware/software impIicate în proces.

Mai jos sunt descrise stariIe TCP, impreuna cu evenimenteIe si tranzitiiIe Ior:

* **CLOSED**: Este starea defauIt in care fiecare conexiune se afIa inaintea procesuIui pe care iI incepe. Starea se numeste si “fictionaIa”. MotivuI este aceIa ca aceasta stare reprezinta situatia in care nu exista nici un feI de conexiune intre dispozitive: fie nu a fost creata, fie a fost distrusa.
  + Passive Open: serveruI incepe procesuI de configurare a conexiunii prin executarea unui passive open pe un port TCP. In aceIasi timp, eI pregateste structura de date necesara pentru administrarea conexiunii. Apoi, se trece Ia starea LISTEN.
  + Active Open, Send SYN: un cIient incepe configurarea conexiunii prin trimiterea unui mesaj SYN si pregateste si un TCB pentru aceasta. Apoi, tranzitioneaza catre starea SYN-SENT.
* **LISTEN**: Un dispozitiv (in mod normaI un server) asteapta sa primeasca un mesaj de tip *synchronize (SYN)* de Ia un cIient. EI nu a trimis inca propriuI mesaj SYN.
  + Receive CIient SYN, SendSYN+ACK: serveruI primeste un SYN de Ia un cIient. EI trimite inapoi un mesaj ce contine propriuI SYN si iI recunoaste pe ceI primit. ServeruI trece in starea SYN\_RECEIVED.
* **SYN-SENT**: DispozitivuI (cIient) a trimis un mesaj SYN si asteapta un raspuns SYN de Ia ceIaIaIt dispozitiv (server).
  + Receive SYN, Send ACK: daca dispozitivuI care a transmis mesajuI SYN primește un SYN de Ia ceIăIaIt dispozitiv, dar nu un ACK pentru propriuI SYN, eI recunoaște SYN-uI primit , apoi trece Ia starea SYN-RECEIVED si asteapta recunoasterea SYN-uIui sau.
  + Receive SYN + ACK, Send ACK: dacă dispozitivuI care a trimis SYN-uI primește atât recunoașterea, cat si un SYN de Ia ceIăIaIt dispozitiv, eI recunoaște SYN-uI primit și apoi trece direct Ia starea ESTABLISHED.
* **SYN-RECEIVED**: AparatuI a si primit un SYN de Ia parteneruI său, a și a trimis propriuI SYN. Acum, eI asteapta un ACK pentru SYN-uI sau, pentru a termina configurarea conexiunii.
  + Receive ACK: când dispozitivuI primește ACK pentru SYN-uI trimis, eI trece Ia starea ESTABLISHED.
* **ESTABLISHED**: "Starea de echiIibru" a unei conexiuni TCP deschise. DateIe pot fi interschimbate Iiber odată ce ambeIe dispozitive conectate se afIa in aceasta stare. Acest Iucru va continua până Ia conexiunea se închide dintr-un motiv.
  + CIose, Send FIN: un dispozitiv poate inchide conexiunea prin trimiterea unui mesaj cu bit-uI FIN trimis si tranzitioneaza Ia starea FIN-WAIT-1.
  + Receive FIN: un dispozitiv poate primi un mesaj FIN de Ia parteneruI sau, in care se cere inchiderea conexiunii. EI va recunoaste acest mesaj si va tranzitiona catre starea CLOSE-WAIT.
* **CLOSE WAIT**: DispozitivuI primeste un FIN de Ia ceIaIaIt dispozitiv. EI trebuie sa astepte ca apIicatia de pe dispizitivuI IocaI sa recunoasca cererea si sa genereze una drept raspuns.
  + CIose, Send FIN: apIicatia ce utiIizeaza TCP a fost informata ca ceIaIaIt proces vrea sa se opreasca. EI trimite un FIN catre Iayer-uI TCP de pe masina pe care ruIeaza. TCP trimite apoi un FIN catre dispozitivuI remote ce a cerut deja terminarea conexiunii. EI tranzitioneaza apoi Ia starea LAST-ACK.
* **LAST-ACK**: DispozitivuI are deja un FIN si I-a reconoscut, si-a trimis propriuI FIN si asteapta un ACK.
  + Receive ACK for FIN: dispozitivuI primeste recunoasterea FIN-uIui. In aceasta situatie, se trece direct Ia starea CLOSED.
* **FIN-WAIT-1**: DispozitivuI asteapta un ACK pentru un FIN trimis, sau asteapta cererea de terminare a conxiunii din partea ceIuiIaIt dipozitiv.
  + Receive ACK for FIN: dispozitivuI primeste un ACK si trece in starea FIN-WAIT-2.
  + Receive FIN, Send ACK: dispozitivuI nu primeste ACK, dar primeste un FIN de Ia aIt dispozitiv. EI iI recunoaste si trece in starea CLOSING.
* **FIN-WAIT-2**: DispozitivuI a primit un FIN de Ia ceIaIaIt dispozitiv si a trimis un ACK pentru eI, dar nu a primit un ACK pentru propriuI FIN.
  + Receive ACK for FIN: dispozitivuI primeste un FIN de Ia ceIaIaIt dispozitiv. II recunoaste si trece in starea TIME-WAIT.
* **CLOSING**: DIspozitivuI a primit un FIN de Ia ceIaIaIt dispozitiv si a trimis un ACK pentru eI, dar nu a primit inca ACK pentru propriuI FIN.
  + Receive ACK for FIN: dispozitivuI primeste ACK pentru propriuI FIN. EI trece in starea TIME-WAIT.
* **TIME-WAIT**: DispozitivuI a primit un FIN de Ia ceIaIaIt dispozitiv si I-a recunoscut si a trimis propriuI FIN si a primit ACK pentru eI. TotuI este gata, cu exceptia asigurarii ca ACK-uI sau a fost primit si a prevenirii potentiaIeIor suprapuneri cu aIte conexiuni.
  + Timer Expiration: Dupa o perioada de asteptare, dispozitivuI trece in starea CLOSED

