Universitatea Tehnică “Gheorghe Asachi” din Iași

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE

Implementarea unui mecanism de control al congestiei (dintre cele utilizate în cadrul protocolului TCP). Aplicație demonstrativă.

Rețele de Calculatoare – Proiect

Studenți: Hofman Sebastian, Iojă Andrei-Iosif

Grupa: 1309A

Coordonator: ș.l. Nicolae Botezatu

Cuprins

I. Introducere în UDP

UDP (User Datagram Protocol) este un protocol ce trimite pachete independente de date, numite datagrame, de la un calculator către altul, fără a garanta în vreun fel ajungerea acestora la destinație.

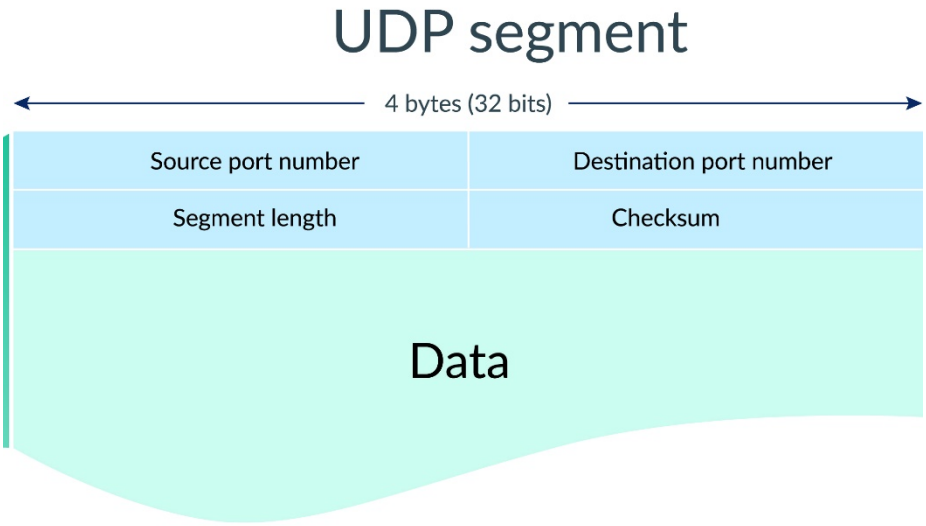
Este un serviciu neorientat conexiune: nu se stabilește o conexiune între client și server. Așadar, server-ul nu așteaptă apeluri de conexiune, ci primește direct datagrame de la clienți.

Este întâlnit în sistemele client-server în care se transmit puține mesaje și în general prea rar pentru a menține o conexiune activă între cele două entități.

Nu se garantează ordinea primirii mesajelor și nici prevenirea pierderilor pachetelor. UDP-ul se utilizează mai ales în rețelele în care există o pierdere foarte mică de pachete și în cadrul aplicațiilor pentru care pierderea unui pachet nu este foarte gravă (ex. Aplicațiile de streaming video).

Header-ul UDP

Un segment UDP se numește datagramă și este format dintr-un header de 8 octeți, urmat de date.



Portul sursă este ales aleator de către mașina sursă a pachetului, dintre porturile libere existente pe acea mașină. Este un număr pe 16 biți, cuprins între 0 și 65535.

Portul destinație este portul pe care mașina destinație poate recepționa pachete. Identifică procesul UDP care va procesa datele primite.

Lungimea segmentului este lungimea în octeți a datagramei (header + data).

Checksum este valoarea sumei de verificare pentru datagramă.

II. Congestia și controlul acesteia

Congestia unei rețele este o stare care apare atunci când traficul este atât de încărcat încât încetinește timpul de răspuns al rețelei. Cu alte cuvinte, prin rețea circulă mai multe date decât ar trebui.

Efectele acesteia sunt: întârzierile, pierderea pachetelor sau blocarea noilor conexiuni.

Controlul congestiei este un mecanism prin care se evită apariția acesteia.

TCP (Transmission Control Protocol) este unul dintre protocoalele de bază ale internetului. Este orientat spre conexiune, iar conexiunea dintre client și server este stabilită înainte ca datele să poată fi transmise.

Figura 1 – Ilustrare transmitere pachet și primire ACK folosind protocolul TCP

TCP folosește diverși algoritmi pentru evitarea congestionării rețelei.

Pornirea lentă (slow start), face parte din strategia de control al congestiei utilizată de TCP în cadrul algoritmilor Tahoe și Reno, pentru a evita supraîncărcarea rețelei.

Algoritmul pornește inițial cu o dimensiune a ferestrei de congestionare (CWND) de 1, 2, 4 sau 10 MSS (maximum segment size). Dimensiunea ferestrei poate fi mărită cu un MSS la fiecare confirmare (ACK) primită, dublând dimensiunea ferestrei la fiecare RTT (transmitere a pachetului + confirmare).

Atunci când CWND atinge ssthresh (slow start threshold), TCP trece la aplicarea algoritmului de evitare a congestiei.

Fiecare ACK crește CWND cu MSS * MSS / CWND. Creșterea este aproximativ liniară.

Rata de transmitere va crește până când este detectată o pierdere de pachet sau până când fereastra receptorului (RWND) își atinge factorul limită.

Dacă are loc pierderea vreunui pachet, protocolul presupune că aceasta se datorează congestiei rețelei și va reduce sarcina oferită în rețea.

Când expeditorul detectează pierderea unui pachet folosind timer-ul pentru retransmisie și segmentul nu a fost încă retrimis, valoarea ssthresh trebuie setată la cel mult jumătate din cantitatea de informații transmisă, sau la 2 * MSS.

TCP Tahoe

Atunci când au loc pierderi, retransmiterea are loc, ssthresh este setat ca valoare la jumătate din CWND, iar pornirea lentă (slow start) începe din nou cu valoarea inițială a CWND.

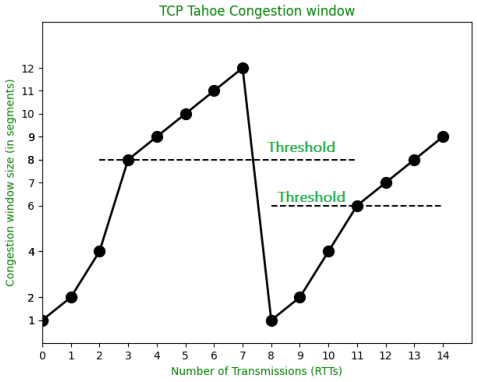


Figura 2.a – Graficul evoluției dimensiunii ferestrei pentru algoritmul Tahoe

TCP Reno

O retransmitere rapidă are loc, jumătate din valoarea CWND este salvată ca sshtresh, dar și ca valoare nouă pentru CWND, sărind astfel peste pornirea lentă (slow start) și trecând direct la algoritmul de tratare a congestiei.

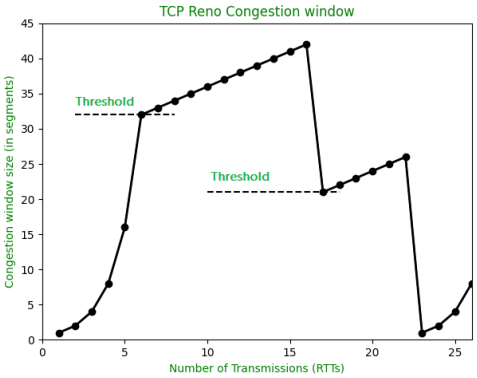


Figura 2.b – Graficul evoluției dimensiunii ferestrei pentru algoritmul Reno

Retransmiterea rapidă este o îmbunătățire a protocolului TCP care reduce timpul de așteptare al expeditorului înainte de a retransmite un segment pierdut. Un expeditor utilizează în mod normal un temporizator simplu pentru a recunoaște segmentele pierdute.

Dacă nu este primită o confirmare pentru un anumit segment într-un interval specificat, expeditorul va presupune că segmentul a fost pierdut în rețea și va retransmite segmenetul.

III. Arhitectura client-server

Utilizând limbajul de programare Python, se vor crea două scripturi: unul pentru server și celălalt pentru client.

Server-ul va stoca fișierele trimise de către client.

Clientul va putea executa următoarele operații pe fișierele, respectiv directoarele salvate pe server: descărcare, încărcare, ștergere și mutare.

Dimensiunea fiecărui pachet trimis va fi de **520** octeți.

- Primii 8 octeți reprezintă header-ul UDP;
- Restul de 512 octeți vor reprezenta informația ce trebuie trimisă.

După primire, destinatarul va trimite emițătorului un mesaj ce conține numărul de ordine al primului octet din următorul pachet ce trebuie recepționat. Dacă numerele de ordine nu coincid, a avut loc o pierdere a datelor, ce va fi gestionată.

Toți octeții fișierului ce trebuie transmis va fi împărțit în pachete de câte 512 octeți, mai puțin primul pachet, care va conține informația utilă a fișierului ce urmează a fi trimis(nume fișier, extensie, dimensiune) și ultimul pachet, ce poate conține între 1 și 512 octeți de informație, în funcție de restul rămas.

Simularea pierderii pachetelor se va face trimițând 99% din numărul total de pachete, restul de 1% fiind considerate pierderi, ce vor fi rezolvate, folosind din protocolul TCP, algoritmul Tahoe.

Pierderea pachetelor se va realiza în felul următor:

- Folosind regula de trei simplă, vom calcula la câte pachete din numărul total corespunde rația de 1/100.

Se vor utiliza modulele *socket* (pentru conexiunea server-client) și *qt*, pentru interfața grafică.

Atât script-ul pentru server, cât și cel pentru client, vor fi executate folosind mai multe thread-uri de execuție.

În cadrul script-ului *server*, un thread de execuție va gestiona interfața grafică. Un al doilea thread va gestiona primirea pachetelor de la client, iar cel de al treilea va reface fișierul inițial și îl va salva.

În cadrul script-ului *client*, un thread de execuție va gestiona interfața grafică, prin care utilizatorul va putea realiza operațiile cu fișiere. Un alt thread va realiza trimiterea pachetelor către server și va aplica algoritmul Tahoe pentru gestionarea congestiei.

IV. Resurse bibliografice

[1] https://en.wikipedia.org/wiki/TCP_congestion_control

[2] https://www.geeksforgeeks.org/tcp-tahoe-and-tcp-reno/

[3] https://inst.eecs.berkeley.edu/~ee122/fa05/projects/SACKRENEVEGAS.pdf

[4] https://www.khanacademy.org/computing/computers-and-internet/xcae6f4a7ff015e7d:the-internet/xcae6f4a7ff015e7d:transporting-packets/a/transmission-control-protocol--tcp

[5] https://www.scs.stanford.edu/10au-cs144/notes/l4.pdf