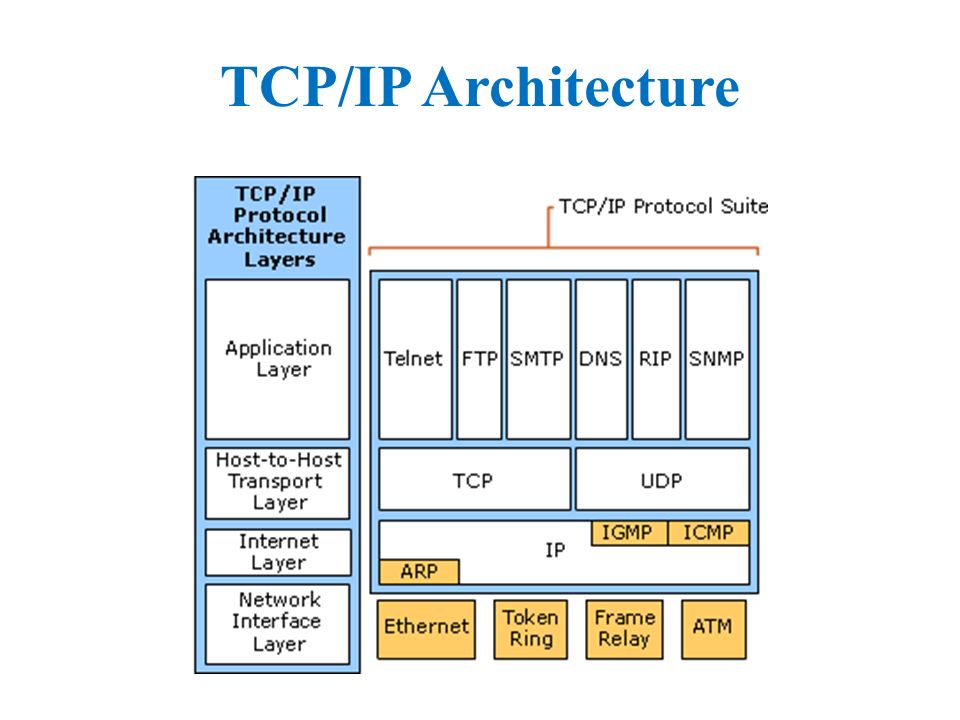
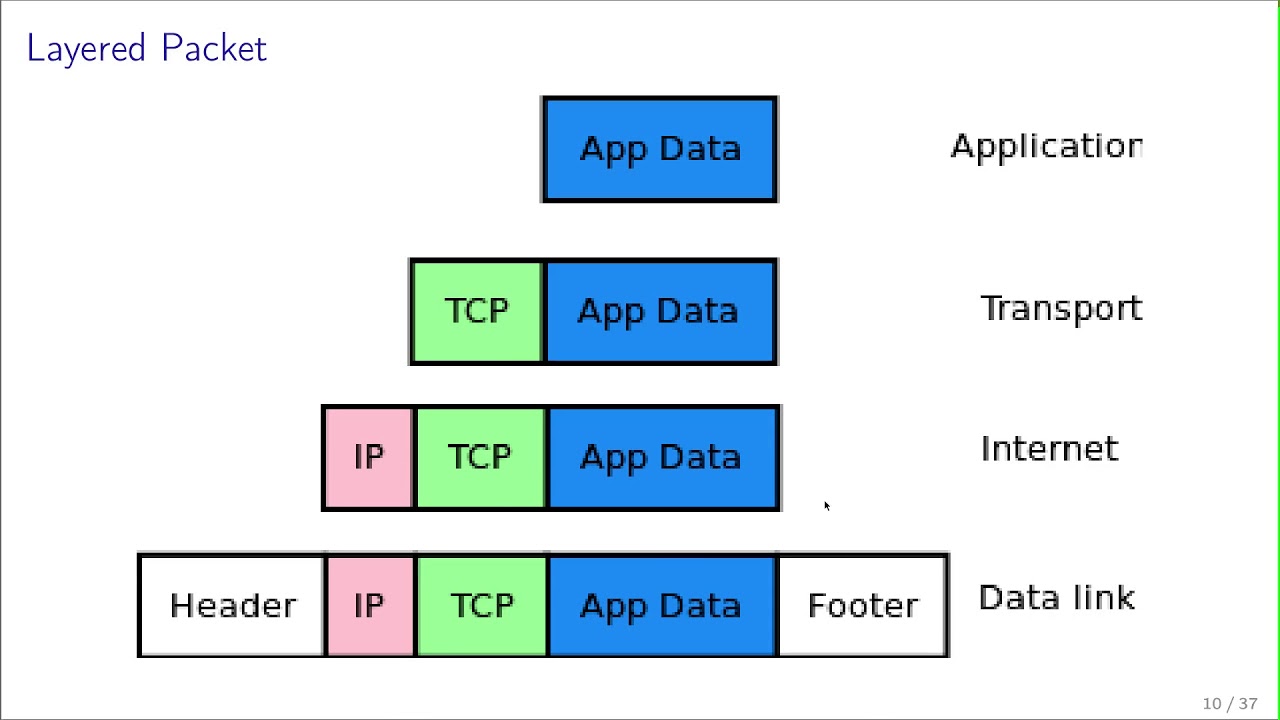
**Transferul fișierelor – Controlul congesiei**

1. **Introducere**

În decursul timpului, Internetul a cunoscut o dezvoltare foarte puternică, ajungând să fie o necesitate primară pentru o mare parte a planetei. Odată cu această evoluție, au fost întâmpinate și diverse probleme soldate cu prăbușirea unor mari părți de rețea, una dintre cauze fiind supraîncărcarea rețelei, rezultat al algoritmilor folosiți în protocoalele de control a transmisiei.   
 Ne vom preocupa de analiza protocoalelor de transmisie (TCP și UDP), de principiul congestiei și vom oferi o soluție a implementării unui mecanism de control a congestiei, prin folosirea protocolului UDP și algoritmul Tahoe de control al congestiei.

1. **TCP. UDP. Controlul congestiilor**

****

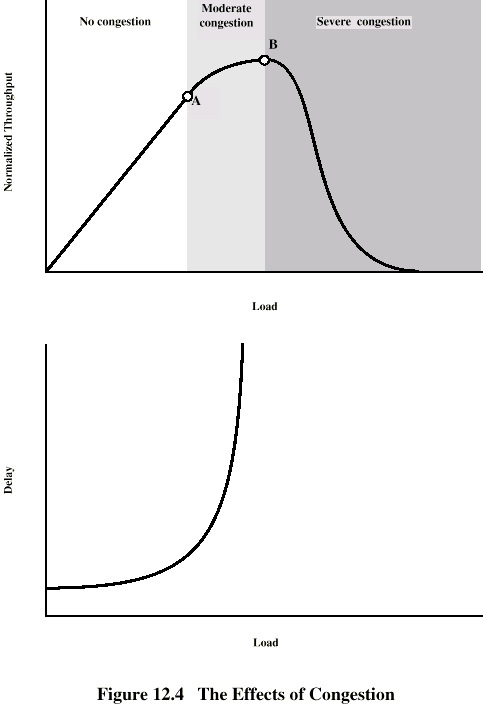
****

TCP este unul dintre protocoalele de bază ale suitei de protocoale Internet și are ca scop asigurarea controlului transferului pachetelor, prin livrarea ordonată a fluxului de octeți de la o sursă la o destinație, în rețea. Pe lângă sarcinile sale de gestionare a traficului, TCP controlează și mărimea segmentului de date, debitul de informație, rata la care se face schimbul de date, precum și evitarea congestionării traficului de rețea.

Astfel, cu siguranță cel mai folosit protocol de control a traficului de date, TCP joacă un rol integral în determinarea performanțelor generale a rețelei.

Modul de operare a protocolului TCP implică existența a trei faze : realizarea conexiunii, transferul de date, închiderea conexiunii. Dintre acestea, vom analiza faza transferului de date.

Transferul de date reprezintă trimiterea datelor de la sursă la destinație, sub formă de pachete. Pachetul este unitatea de informație de bază când ne referim la transferul datelor în rețea și este alcătuit dintr-un header conținând sursa și destinația pachetului, alte secvențe pentru control și datele efective ce trebuie transmise. Premisa de la care plecăm este faptul că destinația este responsabilă pentru controlul rata fluxului de date. Trebuie să ne asigurăm de faptul că rata de transmisie se potrivește cu rata de recepție, rată care trebuie să fie potrivită pentru a asigura o performanță bună, și pentru a nu supraîncărca rețeaua sau destinația. Pentru aceasta, trebuie să ținem cont de controlul congestiilor.

Congestia este un fenomen care apare atunci când pe o rețea se transmit date la capacitatea maximă a rețelei. Acest fenom va produce probleme, cum ar fi pierderea pachetelor sau chiar blocarea rețelei. Există diverși algoritmi implementați în cadrul TCP pentru evitarea congestiei, aceștia bazându-se pe retransmiterea pachetului, ceea ce trebuie foarte atent analizat, deoarece retransmiterea agresivă poate crește congestia, chiar dacă rata de transfer a fost redusă la un nivel la care, în mod normal, nu ar trebui să apară congestie.

Pentru a cunoaște starea conexiunii, emițătorul folosește un timer, astfel încât, pentru fiecare pachet trimis, emițătorul se așteaptă să primească un acknowledgement în perioada de timp impusă, altfel, este generată o eroare, care poate sugera faptul că există congestie. În astfel de situații, TCP trebuie să corecteze această problemă, în primul rând, prin retransmiterea pachetului și să dubleze timpul în care se așteaptă semnal de recepție pentru acest pachet. By default, acest timp este de 1.5 secunde. Se permite încercarea retransmiterii pachetului, până la un maxim de 64 secunde. Totodată, se folosește acest timp și pentru a modifica timpul de așteptare pentru celelalte pachete.

*(scrie tu despre pachete ..ce continut vor avea.. un rezumat al transferului ...adica ..folosim buffer la receptie..ce facem cu el ?.. si mai poti adauga ce mai citesti tu... si daca poti scrie si despre UDP..*

*Eu mai completez diseara cu Algormitul tahoe..)*

☺

1. **UDP (User Data Protocol)**

* are pachete mai mici header = 8 bytes (versus 20 bytes TCP)
* nu e nevoie de conexiune inainte de transmise (se elimina pasii cu realizarea si inchidrea conexiunii)
* control sporit asupra transmiterii (datele se transmit imediat)
* implementeaza detectarea erorilor (optional), dar sterge pachetele cu erori sau seteaza warning flag
* trimite pachetul, daca se pierde din cauza retransmiterii, nu retransmite
* pachetele pot ajunge in ordine diferita => vom folosi un camp pentru ordine
* fara control al congestiei
* message-oriented -> aplicatii face impartirea in pachete (TCP = stream-oriented isi impare singur fisierul in pachete)
* voce, data streaming, (uneori blocat de firewall), DNS lookup, aplicatii care tolereaza pierderi pachete vs text, file transfer, aplicatii pentru care acuratetea e importanta

1. **Proiectarea Aplicatiei**

* Python in PyCharm IDE
* UDP
* alogritm asemanator TCP

Transmitator:

* citeste fisier la nivel de octet
* sparge fisier in segmente de dimensiune fixa
* adaugare simbol inceput/sfarsit de fisier
* impachetare suplimentara
* ID pachet
* ID fisier
* suma de control
* initializare conexiue UDP
* transmisie prin UDP
* (CONTROLUL CONGESTIEI - TAHOE)

Receptor

* initializare buffer
* initializare conexiune UDP
* astept sa primesc (intrerupere?) date
* (CONTROLUL CONGESTIEI - TAHOE)
* extragere pachet din buffer
* despachetare
* reconstruiesc fisier la nivel de octet (in functie de ID file, ID packet)

**Idei scurte:  
-** nivelul transport permite ca mai multe aplicatii sa foloseasca simultan aceeasi retea **-** adresa si portul sunt impachetate la nivel de UDP, nu scriem noi cod in aplicatie pentru asta