MULTITHREADED PROXY-SERVER

PROIECT - PSO

Contribuitori:

Std. sg. Dicu Andrei

Std. sg. Enache Stefan

Grupa C113-D

CUPRINS**:**

1. Introducere
   1. Scopul proiectului
   2. Structura DCS
2. Descrierea generală a produsului software
   1. Descrirea produsului software
   2. Detalierea platformei HW/SW
   3. Constrângeri
3. Detalierea cerințelor software
   1. Cerințele funcționale
   2. Cerințele ne-funcționale

1.Introducere

1.1.Scopul proiectului

Proiectul implica dezvoltarea unui program de tip Proxy-Server a carui sarcina este sa trimita mai departe, in numele clientului si catre serverul de origine al acestuia, request-uri de tip:

* GET
* POST
* CONNECT

Server-ul trebuie sa poata gestiona mai multe solicitari simultane, de aceea se vor utiliza mai multe thread-uri pentru a trata in paralel request-urile de pe fiecare conexiune.

Motivul principal pentru implementare consta in permiterea accesului la surse externe din interiorul retelei protejate de firewall sau direct inaccesibile si deserveste drept gateway pentru intermedierea traficului de retea al protocolului aplicatiei.

1.2.Structura DCS

Structura va fi definita in jurul clasei principale **Proxy** ce foloseste metode prin care se va realiza comunicarea in retea, extragerea informatiei din browser si postarea acestora, astfel vom accepta un socket din browser cu system call, metoda va fi realizata de un main thread, urmand setup-ul socket-ului proxy-ului nostru. Se vor extrage hostname-ul si port-ul , realizand conexiunea cu server-ul cerut, actiune realizata de catre alte thread-uri pentru eficienta.

Alti membri din implementarea vor fi: clasa **Request** pentru siruri de caractere request, conceputa pentru coeziune ridicata, va obtine sirul de solicitare recv client, va analiza numele gazdei si portul in sine si salvarea sirului ; clasa **Response** realizata pentru raspunsuri , ce vor stoca tipul si informatiile necesare.

2.Descrierea generală a produsului software

2.1. Descrirea produsului software

Server-ul proxy va fi utilizat pentru functionalitatea , securitatea pentru comunicarea in retea dintre parti, principiul de baza al functionarii fiind de a primi cereri de client, aceste cerinte sunt analizate si se executa trimiterea catre serverele tinta. Acesta actioneaza asupra nivelului 7 din modelul OSI (aplicatie) pentru a analiza cererile primite. Pe langa acest proxy de aplicatie, exista si alte independente de aplicatie, care furnizeaza numai pachete de transport fara cunostintele protocoalelor de nivel de aplicatie. Folosirea lor necesita totusi utilizarea unui protocol de comunicare specific pentru a comunica cu server-ul proxy. Ca reprezentant tipic al protocolului folosim SOCKS ce lucreaza pe al 5-lea strat al modelului ISO/OSI al sesiunii, acesta va directiona pachetele de retea intre un client si un server.

2.2. Detalierea platformei HW/SW

Produsul este alcatuit dintr o singura aplicatie software, ce va fi atajata retelei in realizarea legaturii in acest spatiu dintre sarcina de lucru intre furnizorii unei resurse sau serviciu si solicitantii de servicii. Ca instrument de lucru vom folosi : VS Code , intr un mediu Linux.

2.3. Constrângeri

* Serverul implementeaza suport complet pentru standart-ul HTTP 1.0, adica metodele GET , POST, HEAD si accepta un subset al standartului HTTP 1.1 pentru a permite comunicarea fara probleme a implementarilor existente ale clientului si serverului(de exemplu: browsere web si servere web).
* Serverul utilizeaza un minim de memorie globala alocata pe heap. La reducerea consumului de memorie, comunicatiile HTTP dintre client si serverul tinta sunt rezolvate prin intercalare. Nu asteapta preluarea intregii cereri a clientului (continand cerinte pentru entitatile HTTP, cum ar fi POST) sau raspunsuri intregi ale server-ului proxy.
* Serverul este eficient in ceea ce priveste consumul de timp al procesorului. Aceasta eficacitate este redata de proxy pentru ca realizeaza toate operatiie de intrare si iesire din retea. Aplicatia petrecand de cele mai multe ori asteptand sosirea de date suplimentare sau conexiuni ulterioare.

3.Detalierea cerințelor software

3.1.Cerințele funcționale

* Permiterea a cati mai multi clienti sa poata accesa serverul proxy.
* Toate raspunsurile la solicitari trec prin serverul proxy, iar daca serverul stocheaza continutul raspunsurilor poate imbunatatii timpul de raspuns si poate reduce latimea de banda de repetat.
* Permiterea procesarii a cerintelor mai fine. Organizatia poate restrictiona accesul la computerele client individuale prin adresa de destinatie, protocol sau tip de resursa. Acesta poate accepta o limita de timp intr-o zi sau un control al ratei, ceea ce duce poate reduce utilizarea ineficienta.
* Securitatea sporita poate fi obtinuta prin utilizarea proxy-ului pentru inregistrarea si auditarea cererilor clientului, spre deosebire de logarea la straturile inferioare, permite inregistrarea la nivelului aplicatiei acces usor la toate proprietatile tranzactiei client/server.
* Folosirea pentru acces anonim la serverul tinta. Acest efect poate fi utilizat pentru a ocoli restrictiile site -ului aplicabile adresei sursa relevante a clientului, proxy-ul configurat corespunzator mascheaza , de asemenea, atributele sistemului client, cum ar tipul si versiunea software-ului etc.
* Arhitectura programata pentru mai multe fire de executie.Thread-urile creeaza job-uri si le insereaza in coada, noile job-uri din coada sunt procesate de propriul thread consumator. Aceasta procesare se efectueaza in paralel pentru atatea solicitari cate fire de executie sunt in curs de desfasurare.
* Dupa pornirea functiei main() se deschide socket-ul acceptand noi conexiuni.Noua comunicare de intrare este verificata daca nu depaseste limita maxima de conexiuni.

3.2.Cerințele ne-funcționale

* Produsul poate functiona pe sisteme de operare distribuite Linux.
* Cerintele de performanta pentru functionarea optima sunt minime.
* Aplicatia va avea nevoie de o comunicatie intr o retea pentru realizarea legaturilor intre utilizatorii acesteia.