**Universiteti i Prishtinës “Hasan Prishtina”**

**Fakulteti Inxhinierisë Elektrike dhe Kompjuterike**

****

**Dokumentim teknik i projektit**

**Lënda: Sisteme Operative**

**Titulli i projektit: Distributed System using Inter Process Comunications**

**(IPC), Threads, and Synchronization**

**Emri profesorit/Asistentit Emri & mbiemri studentëve / email adresa**

|  |  |  |
| --- | --- | --- |
| Prof. Dr. Artan Mazrekaj  Ass. Dalina Vranovci | 1. Gentrit Bytyci | gentrit.bytyci1@student.uni-pr.edu |
| 2. Laura Tafaj | laura.tafaj@student.uni-pr.edu |
| 3. Suhejla Hoxha | suhejla.hoxha@student.uni-pr.edu |
| 4.Ylljete Kicaj | ylljete.kicaj@student.uni-pr.edu |

Prishtinë, 2023

Përmbajtja

[**Abstrakti** 3](#_Toc136348001)

[**I.** **Hyrje** 4](#_Toc136348002)

[**II.** **Qëllimi i punimit** 5](#_Toc136348003)

[**III.** **Implementimi** 6](#_Toc136348004)

[**IV.** **Konkluzione** 8](#_Toc136348005)

[**Referencat** 9](#_Toc136348006)

# **Abstrakti**

Ky dokumentacion ofron një përmbledhje dhe shpjegim të detajuar të një sistemi të shpërndarë të zbatuar duke përdorur komunikimet ndër-procesore (IPC), threads dhe sinkronizimin në gjuhën e programimit C në platformën Linux. Sistemi përbëhet nga një program server që mund të trajtojë kërkesat nga shumë klientë njëkohësisht, duke përdorur mekanizmin shared memory.

Programi i serverit është krijuar për të trajtuar një numër maksimal klientësh, me kufirin e specifikuar si një konstante. Ai përdor threads për të trajtuar kërkesat nga klientët në të njëjtën kohë. Mekanizmat e sinkronizimit, duke përfshirë mutexes dhe semaforë, përdoren për të siguruar ndarjen dhe mbrojtjen e duhur të burimeve midis temave.

Funksionalitetet kryesore të programit të serverit përfshijnë marrjen dhe përpunimin e kërkesave nga klientët, kthimin e të dhënave të kërkesës së klientit në përgjigje dhe demonstrimin e teknikave të duhura të sinkronizimit. Programi i serverit është i mirë-dokumentuar dhe i lehtë për t'u kuptuar, duke ofruar shpjegime të qarta të mekanizmave të zbatuar dhe strategjive të sinkronizimit.

Përveç programit të serverit, dokumentacioni mbulon zbatimin e programeve të klientit që mund të lidhen me serverin duke përdorur mekanizmat IPC dhe të dërgojnë kërkesa. Trajtimi i gabimeve për të gjitha funksionet IPC të përdorura në implementim trajtohet gjithashtu, duke siguruar qëndrueshmëri dhe besueshmëri.

Dokumentacioni përfshin copa kodi, shpjegime të zgjedhjeve të projektimit dhe zbatimit, dhe udhëzime hap pas hapi për ekzekutimin dhe testimin e sistemit të shpërndarë. Projekti shërben si një shembull gjithëpërfshirës i zhvillimit të një sistemi të shpërndarë duke përdorur IPC, threads dhe sinkronizimin në C, duke ofruar njohuri mbi konceptet dhe teknikat e përfshira në ndërtimin e sistemeve të tilla.

# **Hyrje**

Në fushën e sistemeve të shpërndara, Komunikimi Ndër-Procesor (IPC), thread-at dhe sinkronizimi janë koncepte themelore që luajnë role jetike në arritjen e komunikimit efikas dhe të njëkohshëm midis proceseve ose thread-ave. Kuptimi i këtyre koncepteve është thelbësor për zhvillimin e sistemeve të shpërndara të fuqishme dhe të shkallëzueshme.

IPC i referohet mekanizmave dhe teknikave të përdorura për komunikimin midis proceseve ose thread-eve që funksionojnë në të njëjtat ose makina të ndryshme. Ai mundëson shkëmbimin, koordinimin dhe sinkronizimin e të dhënave, duke lejuar që komponentët e shpërndarë të bashkëpunojnë në mënyrë efektive. Mekanizmat IPC sigurojnë mjete për proceset ose thread-at për të shkëmbyer informacion, për të shkembyer mesazhe dhe për të hyrë në burimet e përbashkëta, duke lehtësuar komunikimin ndër-procesor. Mekanizmat e zakonshëm IPC përfshijnë pipes, message passing dhe shared memory. Çdo mekanizëm ka karakteristikat, avantazhet dhe kufizimet e veta, duke e bërë atë të përshtatshëm për skenarë dhe kërkesa të ndryshme në sistemet e shpërndara.

Thread-et janë njësi ekzekutimi të lehta brenda një procesi që mund të funksionojë njëkohësisht, duke mundësuar paralelizëm dhe shfrytëzim efikas të burimeve të sistemit. Thread-at ndajnë të njëjtën hapësirë memorie brenda një procesi dhe kanë stack-in dhe kontekstin e tyre të ekzekutimit, duke i lejuar ata të kryejnë detyra njëkohësisht. Thread-et mund të komunikojnë dhe sinkronizohen me njëri-tjetrin përmes variablave të përbashkëta dhe primitivëve të sinkronizimit. Në sistemet e shpërndara, thread-et përdoren shpesh për të trajtuar kërkesa të shumta klientësh në të njëjtën kohë, duke përmirësuar reagimin dhe shkallëzimin e sistemit. Megjithatë, bashkëkohësia e bazuar në thread-a paraqet sfida të tilla si sinkronizimi, konsistenca e të dhënave dhe kushtet e garës, të cilat duhet të trajtohen me kujdes.

Mekanizmat e sinkronizimit janë thelbësorë për koordinimin e aksesit në burimet e përbashkëta dhe sigurimin e sigurisë së thread-ave në një mjedis të njëkohshëm. Ato parandalojnë garat e të dhënave dhe konfliktet që mund të lindin kur thread-at ose procese të shumta aksesojnë të dhënat e përbashkëta në të njëjtën kohë. Primitivët e zakonshëm të sinkronizimit përfshijnë mutexes dhe semaforët. Këta mekanizma ofrojnë mjete për të zbatuar përjashtimin, sinjalizimin dhe koordinimin e ndërsjellë ndërmjet thread-ave ose proceseve, duke siguruar korrektësi dhe qëndrueshmëri në sistemet e shpërndara.

# **Qëllimi i punimit**

Qëllimi i këtij punimi është të demonstrojë sistemet e shpërndara në përdorimin e mekanizmave IPC, konkurencën e bazuar në thread dhe sinkronizimin për të arritur komunikim dhe koordinim efikas midis një serveri dhe klientëve të shumtë.

**Avantazhet e Distributed System using Inter Process Comunications(IPC), Threads, and Synchronization:**

* Mekanizmat IPC mundësojnë komunikim efikas dhe shkëmbim të të dhënave ndërmjet komponentëve të shpërndarë, duke lejuar shkallëzueshmëri të pandërprerë.
* Fijet në sistemet e shpërndara lejojnë përpunimin e njëkohshëm të kërkesave të klientit. Kjo do të thotë që shumë klientë mund të shërbehen njëkohësisht, duke përmirësuar reagimin e sistemit dhe duke reduktuar vonesën.
* Mekanizmat IPC si memoria e përbashkët lehtësojnë ndarjen efikase të të dhënave dhe burimeve midis komponentëve të shpërndarë
* Ofrojnë një arkitekturë fleksibile.
* Lejojnë zbatimin e mekanizmave tolerantë ndaj gabimeve siç janë radhët e mesazheve, kontrollet e tepricës dhe trajtimi i gabimeve.
* Ndarja dhe koordinimi efikas i burimeve nëpërmjet mekanizmave të sinkronizimit parandalojnë pengesat dhe grindjet, duke rritur më tej performancën.
* Zgjerohen lehtësisht për të inkorporuar komponentë, protokolle ose teknologji të reja

**Disavantazhet:**

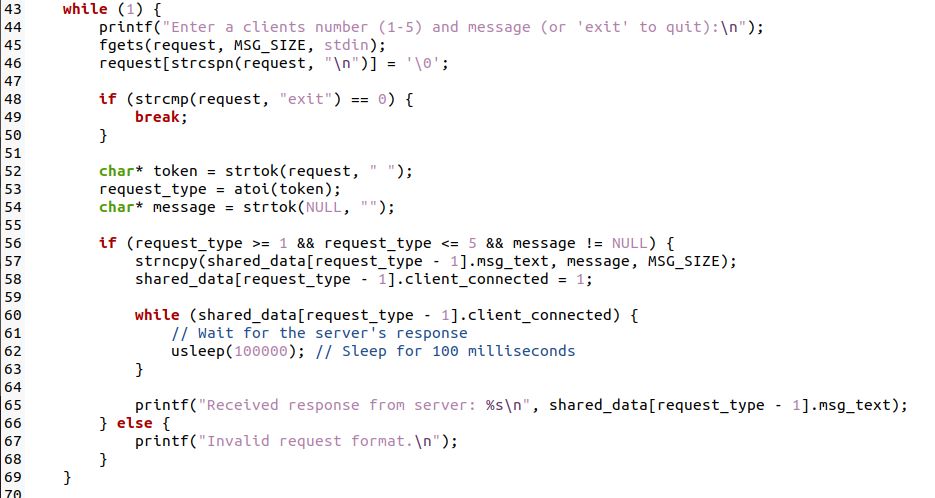
* Zhvillimi dhe menaxhimi i sistemeve të shpërndara mund të jetë kompleks për shkak të sfidave të qenësishme të koordinimit të proceseve ose thread-ave të shumta.
* Shpenzime të larta shtesë në krahasim me komunikimin lokal ndër-procesor.
* Përdorimi i tepërt i semaforeve dhe mutex ose i gabuar mund të sjellë pengesa në performancë dhe të rrisë kompleksitetin, duke çuar potencialisht në bllokime ose bllokime të gjalla.
* Korrigjimi dhe zgjidhja e problemeve të sistemeve të shpërndara mund të jetë më komplekse sesa korrigjimi i sistemeve të centralizuara

# **Implementimi**

client.c



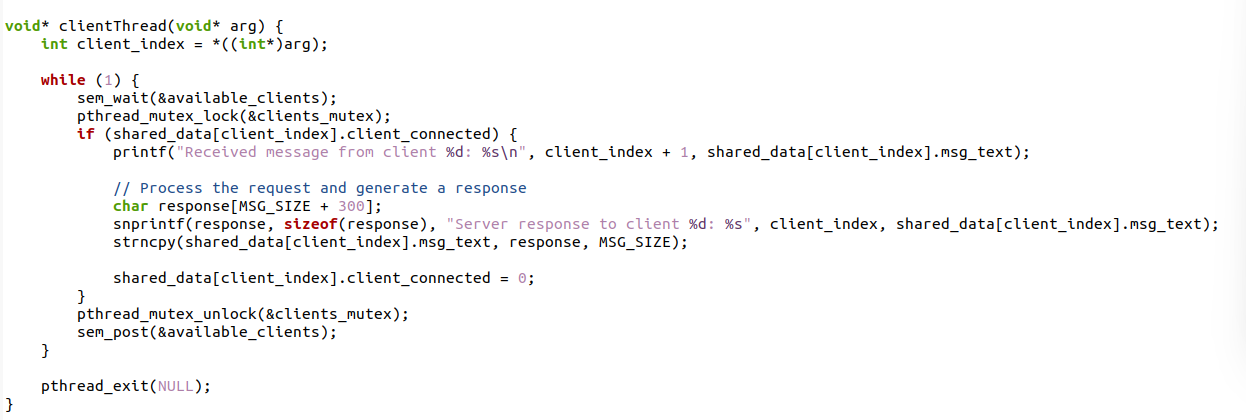
Në fillim, kodi përdor funksionin ftok për të krijuar një çelës unik për ndarjen e memories. Pastaj, përdor funksionin shmget për të marrë identifikuesin e segmentit të memorisë së ndarë duke përdorur çelësin dhe madhësinë e caktuar. Në fund, përdor funksionin shmat për të lidhur segmentin e memorisë së ndarë me hapësirën e adresave të klientit.



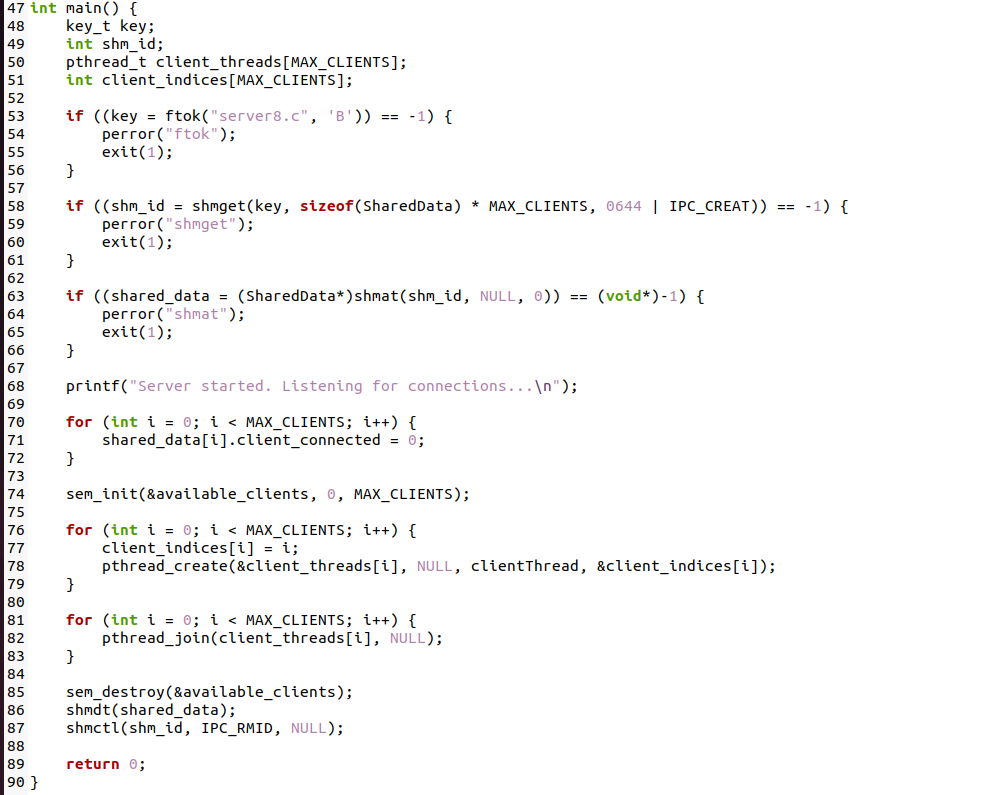
Ky pjesë e kodit përfaqëson një cikël pafund në të cilin klienti pranon dhe dërgon kërkesa në server. Klienti kërkon nga përdoruesi të jep numrin e klientit dhe mesazhin për t'u dërguar. Nëse kërkesa është "exit", cikli thyhet dhe programi mbyllet. Përndryshe, kërkesa dhe mesazhi ndahen dhe validohen. Nëse kërkesa dhe mesazhi janë valide, ato kopjohen në hapësirën e ndarë të memorisë për klientin përkatës. Pas dërgimit të kërkesës, klienti pret për përgjigjen nga serveri. Për ta bërë pritjen e kontrolluar, përdoret funksioni usleep për të pritur për 100 milliseconda.

Në fund, afishohet përgjigja e pranuar nga serveri. Nëse kërkesa dhe mesazhi nuk janë valide, shfaqet një mesazh për formatin e pavlefshëm të kërkesës. Programi vazhdon të pranojë dhe dërgojë kërkesa derisa të marrë kërkesën "exit" nga përdoruesi. Pas daljes nga cikli, klienti shkëputet nga segmenti i ndarë i memorisë dhe programi mbyllet.

server.c



Funksioni clientThread është funksioni që ekzekutohet nëpërmjet thread-ave të klientëve në serveri. Ky funksion është përgjegjës për të pranuar mesazhe nga klientët, të përpunojë kërkesat dhe të krijojë përgjigjet për ta.



Kjo pjese e kodit implementon një server që pranon lidhje nga klientët dhe përdor thread-a të veçanta për të shërbyer secilin klient në mënyrë paralele.

# **Konkluzioni**

Dokumentacioni ofron një shpjegim të detajuar të funksionaliteteve kryesore të programit të serverit, duke përfshirë marrjen dhe përpunimin e kërkesave nga klientët, kthimin e të dhënave të kërkesës së klientit në përgjigje dhe demonstrimin e teknikave të duhura të sinkronizimit. Gjithashtu, dokumentacioni mbulon implementimin e programeve të klientit që lidhen me serverin duke përdorur mekanizmat IPC dhe trajtimin e gabimeve të mundshme në funksionet IPC.

# **Referencat**

|  |  |
| --- | --- |
| [1] | <https://sectigostore.com/blog/42-cyber-attack-statistics-by-year-a-look-at-the-last-decade/> |
| [2] | <https://heimdalsecurity.com/blog/10-surprising-cyber-security-facts-that-may-affect-your-online-safety/> |
| [3] | [https://null-byte.ëonderhoëto.com/hoë-to/discover-xss-security-flaës-by-fuzzing-ëith-burp-suite-ëfuzz-xsstrike-0189971/](https://null-byte.wonderhowto.com/how-to/discover-xss-security-flaws-by-fuzzing-with-burp-suite-wfuzz-xsstrike-0189971/) |
| [4] | [https://null-byte.ëonderhoëto.com/hoë-to/hack-like-pro-hack-ëeb-apps-part-4-hacking-form-authentication-ëith-burp-suite-0163642/](https://null-byte.wonderhowto.com/how-to/hack-like-pro-hack-web-apps-part-4-hacking-form-authentication-with-burp-suite-0163642/) |
| [5] | [https://portsëigger.net/ëeb-security/sql-injection/union-attacks](https://portswigger.net/web-security/sql-injection/union-attacks) |
| [6] | https://ëëë.youtube.com/ëatch?v=n2YxdgX5SJA&list=PLuyTk2\_mYISLaZC4fVqDuË\_hOk0dd5rlf&index=11 |
| [7] | [https://github.com/rkhal101/Ëeb-Security-Academy-Series/blob/main/sql-injection/lab-09/sqli-lab-09.py](https://github.com/rkhal101/Web-Security-Academy-Series/blob/main/sql-injection/lab-09/sqli-lab-09.py) |