ACADEMIA TEHNICĂ MILITARĂ "FERDINAND I"

FACULTATEA DE SISTEME INFORMATICE ŞI SECURITATE CIBERNETICĂ

Specializarea: Calculatoare şi sisteme informatice pentru apărare şi securitate naţională

SERVER DNS

Prof. coord. Slt. Ing. Adina Vaman

Aldea Alexandra-Marilena
Graure Dariana-Gabriela
Grupa C 113E

Cuprins:

1. INTRODUCERE

Acest proiect urmărește implementarea unui server DNS (Domain Name System), un sistem esențial în infrastructura internetului, responsabil pentru traducerea numelor de domenii (ex. www.example.com) în adrese IP (ex. 192.168.1.1) pe care dispozitivele le pot utiliza pentru comunicare. DNS-ul este un serviciu fundamental care asigură că utilizatorii pot accesa site-uri web, aplicații și alte resurse de pe internet, utilizând nume de domenii ușor de reținut în loc de adrese numerice complexe. DNS-ul permite o navigare fluidă și rapidă pe internet, eliminând nevoia de a reține adrese IP complexe și oferind totodată posibilitatea de redirecționare și balansare a încărcării prin asocierea mai multor adrese IP cu același nume de domeniu.

2. OBIECTIVE PRINCIPALE

- Dezvoltarea unui server DNS principal (**MASTER**) capabil să răspundă cererilor de tip DNS pe baza unui fișier principal (Master File), un cache local (structura de date), cât și pe baza unei funcții de forwarding către alt server DNS (Google DNS), și un server DNS secundar (**SLAVE**) pe baza unui fișier secundar (Slave File), fiind un server de backup.
- Gestionarea expirării intrărilor în cache printr-un mecanism de temporizare.
- Adăugarea funcționalității de reverse DNS lookup (pe baza adresei IP primite identificăm numele de domeniu).

3. CERINȚE FUNCȚIONALE

1. Crearea unui server principal.
2. Realizarea conexiunii între client și server prin sockets(socket de tip UDP pentru transmiterea și primirea datagramelor, nu mai mult de 512 BYTES).
3. Creearea unui threadpool la începutul programului, fiecare cerere a clientului fiind redirecționată către un thread disponibil, urmând ca după transmiterea răspunsului către client să fie eliberat thread-ul.
4. Crearea unui socket de tip UDP pentru forwardarea cererilor către GOOGLE DNS server.
5. Crearea unei structuri de tip cache pentru stocarea temporară a adresei IP, a numelui de domeniu, a type-ului și a timer-ului.
6. Crearea unui fișier MasterFile pentru identificarea zonelor de domeniu, unde are acces serverul DNS.
7. Crearea fișierelor aferente fiecărei zone de domeniu.
8. Crearea unui server secundar.
9. Crearea unui fișier SlaveFile fiind o copie a fișierului MasterFile.
10. Crearea unui Makefile și a unui script atât pentru client, cât și pentru server.
11. Crearea unor structuri pentru stocarea datelor conform RFC 1035.

4. STRUCTURA PROIECTULUI

Se pornesc cele doua servere locale, cel Master fiind autoritatea principală care primește cererile de la clienți, iar cel Slave acționează în momentul în care serverul principal nu mai funcționează.

Clientul trimite o cerere folosind comanda *mydns*:

Exemple:

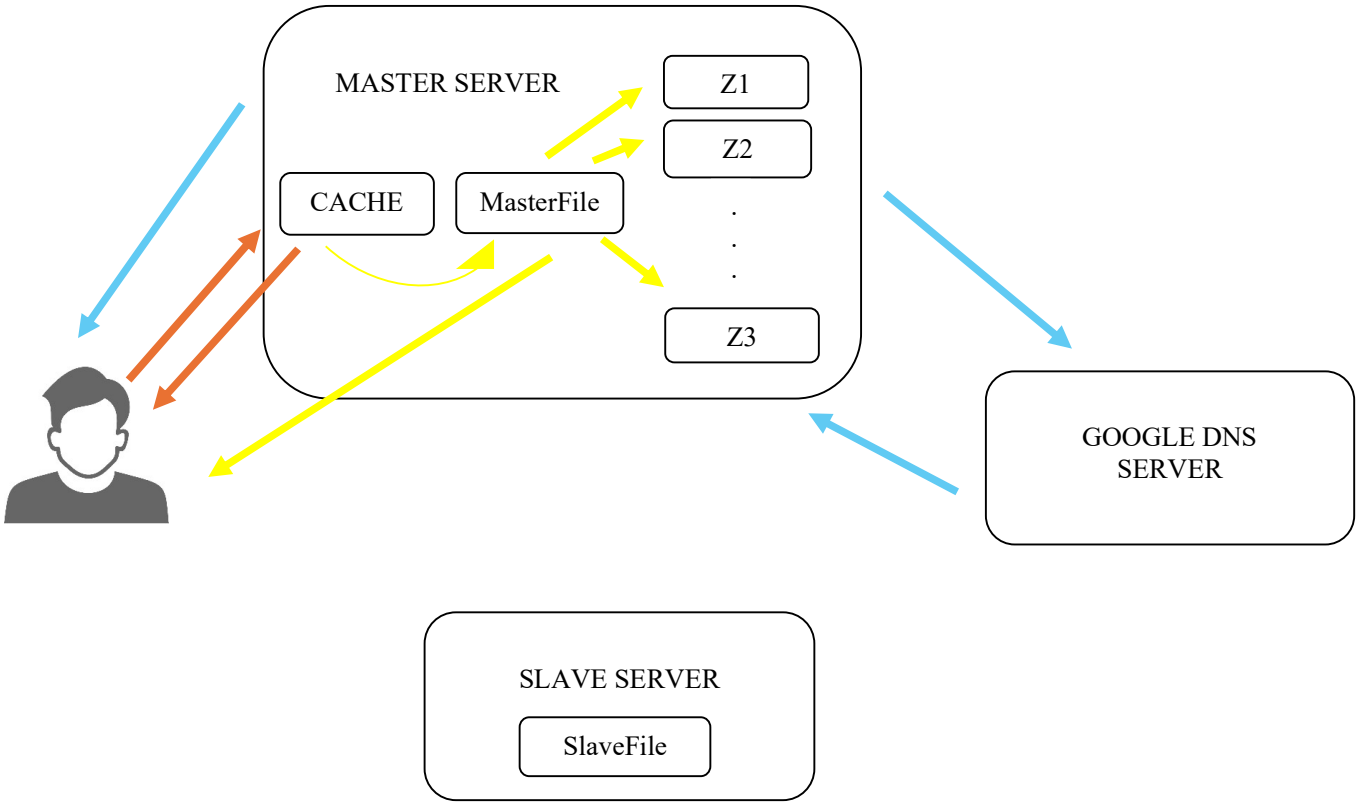
-mydns www.google.com
-mydns -MX www.google.com (pentru mail)
-mydns -NS www.google.com (pentru name server)
-mydns -AAAA www.google.com (IPv6)
-mydns -X 8.8.8.8 (pentru reverse)

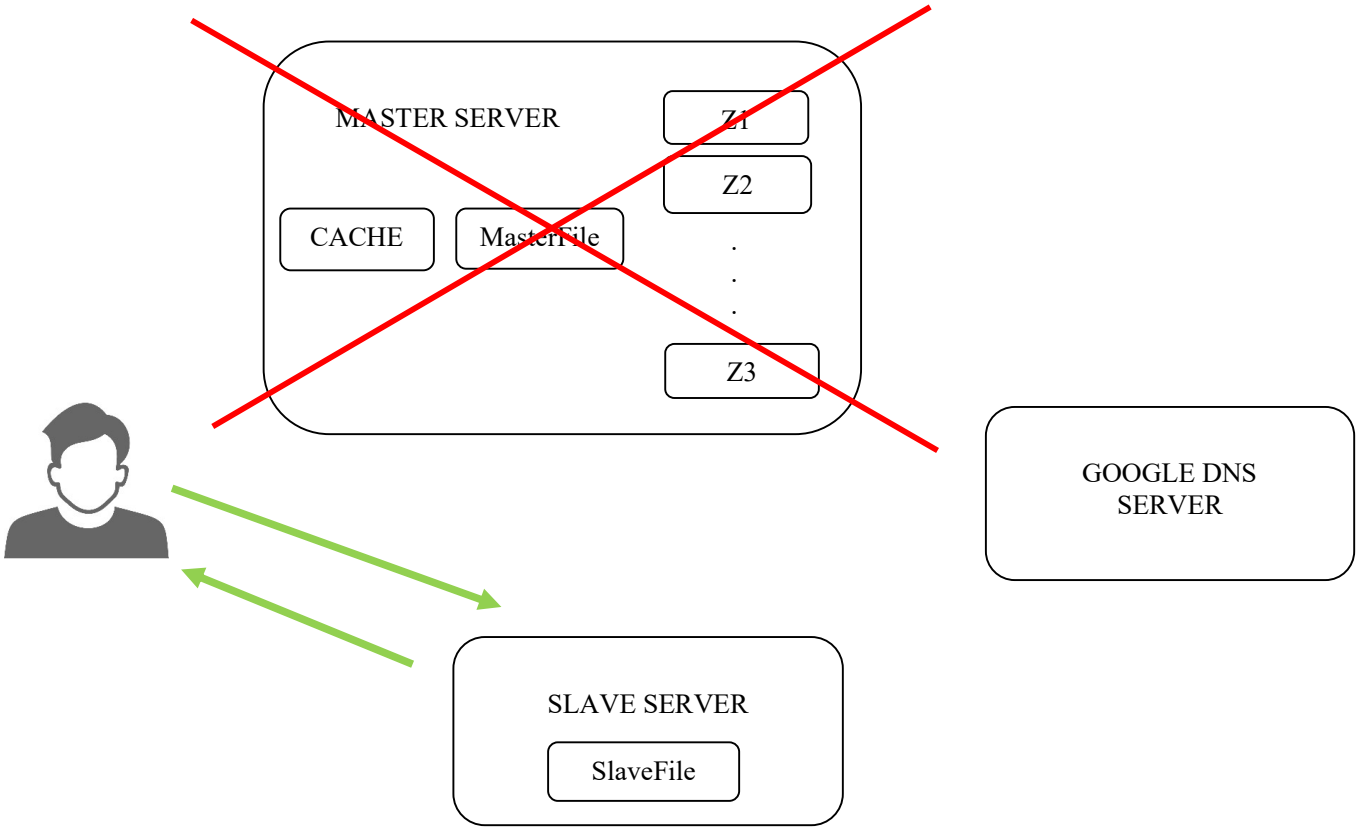
Pentru a folosi *mydns* clientul are implementat un script și un Makefile pe care îl execută pentru a putea folosi această comandă (ex: dig). De asemenea, se poate folosi dig împreună cu adresa IP și portul serverului local.

Serverul principal primește cererea de la client și verifică în structura CACHE dacă există. În cazul în care există, clientul primește de la server răspunsul, iar timer-ul specific este actualizat (datele rezistă in cache maxim 86400 s). În cazul în care nu există, serverul caută în Masterfile și încearcă să identifice zona de domeniu. În cazul în care este găsit în Masterfile, deschide fișierul aferent zonei și trimite răspunsul către client. În cazul în care nu există în Masterfile, face o cerere către GOOGLE DNS server, iar răspunsul primit este redirecționat către client.

În cazul în care serverul principal nu mai funcționează din diferite motive, cererea clientului este trimisă la serverul secundar. Aceasta este verificată în fișierul SlaveFile. Dacă există, clientul primește răspunsul, iar în caz contrar primește un mesaj de informare cu faptul că răspunsul nu poate fi generat.

5. DIAGRAMA FLUXULUI DE DATE





Exemplu de fișier MasterFile:

```
zone "example.com"
{
    type master;
    file "/home/student/Proiect_PSO
};

zone "mta.ro"
{
```

Exemplu de zonă de domeniu (example.com):

```
$TTL 86400
@   IN  SOA ns.icann.org.   noc.
    2024102901  ;   Serial
    7200        ;   Refresh
    3600        ;   Retry
    1209600     ;   Expire
    3600    )   ;   Minimum TTL
@ IN  NS  a.iana-servers.net.
@   IN  NS  b.iana-servers.net.
@ IN A 93.184.215.14
```

6. CONCLUZIE

Implementarea acestui server DNS aduce o serie de beneficii semnificative care îmbunătățesc atât performanța, cât și flexibilitatea și scalabilitatea infrastructurii DNS. Aceste avantaje sunt esențiale în asigurarea unei experiențe de navigare rapidă și sigură pentru utilizatori și în optimizarea gestionării cererilor de rezoluție DNS la nivel de rețea.

Eficiență: Prin folosirea cache-ului local și a unui fișier de zonă principal, serverul răspunde mai rapid cererilor frecvente.

Redirecționare flexibilă: Serverul este capabil să redirecționeze cererile necunoscute către un server DNS public, permițând obținerea rapidă a adreselor pentru domeniile necunoscute.

Scalabilitate: Arhitectura bazată pe cache poate fi extinsă pentru a include politici avansate de expirare și actualizare.