Raport Tehnic - myDNS

Tamaș Luca-Ștefan - 2E3

Facultatea de Informatică din cadrul Universității "Alexandru Ioan Cuza", Iași, Romania

lucastefantamas@gmail.com

Abstract:

Acest raport prezintă o abordare asupra tematicii problemei "myDNS" și o schematizare a întregului proiect susținută atât prin detalii tehnice de abordare a problemei și de implementare a soluțiilor prin intermediul tehnologiilor studiate, cât și prin diagrame reprezentative și secvențe importante de cod.

1. Introducere:

Proiectul ales este intitulat "myDNS" și are ca cerință proiectarea și implementarea unui sistem DNS propriu, luându-se în calcul atât modelul server-client, cât și diverse aspecte legate de optimizarea procesului aplicate în sistemele reale. Serverele create trebuie să funcționeze concurent pentru a oferi raspunsuri în mod simultan câtor mai mulți clienți.

Domain Name System (abreviat DNS, în traducerea literară Sistemul Numelor De Domenii) este un sistem distribuit de nume utilizat pentru identificarea calculatoarelor din Internet sau din alte rețele pe bază de Internet Protocol.

Motivul alegerii acestui proiect este înțelegerea mecanismelor folosite în producții comerciale de sisteme DNS și aprofundarea tehnologiilor studiate în cadrul disciplinei aferente.

2. Tehnologii utilizate:

Pentru comunicarea dintre servere și client se va utiliza protocolul TCP deoarece acesta este un protocol orientat pe conexiune, astfel realizându-se o conexiune stabilă între cele două instanțe pentru a se asigura persistența datelor transmise. De asemenea se vor folosi socket-pair – uri pentru fiecare canal de comunicare dintre clientți și servere.

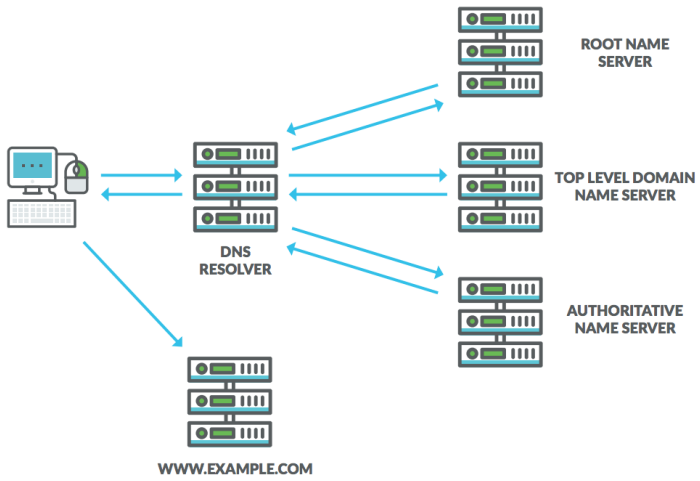
Pentru rularea concurentă a serverelor se va folosi api-ul "fork" care va realiza procese copil pentru fiecare instanță de client care se conectează la server, dar și pentru legăturile paralele dintre servere.

Pentru eficientizarea sistemului DNS se va folosi paradigma de "cache" atât la nivelul clientului cât și la nivelul anumitor servere, cu scopul de a memora adresele IP legate de anumite domenii, astfel putând fi evitat procesul de "lookup" pentru adresa aferentă domeniului căutat.

3. Arhitectura aplicației:

Aplicația va funcționa după modelul server-client, utilizatorul având la dispoziție o consolă pentru realizarea căutării adresei IP a domeniului introdus de la tastatură.

Întreg sistemul DNS creat va replica pe cât posibil sistemul DNS pus în aplicare în întreaga lume, ce urmează următoarea diagramă:



Se poate observa că modelul propus urmează rezoluția iterativă, astfel fiecare server v-a returna serverului resolver ori adresa IP a domeniului căutat, ori adresa IP a următorului server care va realiza căutarea, cu precizarea faptului că în ultima situație, serverul va reține adresa domeniului într-un fișier de cache.

Pașii urmați de întreg sistemul sunt următorii:

1) Căutarea adresei IP în cache-ul clientului. Dacă aceasta nu este găsită se apelează la serverul resolver;

2) Căutarea adresei IP în cache-ul resolverului. Dacă aceasta nu este găsită se apelează la serverul root;

3) Serverul root va redirecționa resolverul către unul dintre serverele top level domain în funcție de numele domeniului căutat.

4) Serverul top level domain va redirecționa resolverul către un server authorative name pentru a primi adresa IP a domeniului căutat dacă acesta se află în fișierul de date ale serverului, sau o eroare în caz contrar.

5) Salvarea adresei IP în cache-ul resolverului și în cache-ul clientului.

4. Detalii de implementare:

Serverele vor rula în mod concurent, astfel pentru fiecare nouă conexiune se va realiza un nou proces copil din procesul părinte, folosit exclusiv receptării intențiilor de conexiune. Astfel de procedeu va fi construit pe baza codului:

```
while(true){
   waitingNewConnection();

   if(newConnection()){
      createChild();
   }
}
```

Un astfel de mecanism trebuie susținut de o metodă pentru terminarea proceselor inactive (copii care și-au terminat execuția) pentru a elibera PID-urile neutilizate. O soluție la această problemă poate fi ascultarea semnalului de oprire al oricărui copil și dealocarea PID-ului său prin folosirea api-ului wait();

Atât la nivelul clientului cât și la nivelul serverului resolver se va implementa un mecanism de caching, care va urma următorul pseudocod:

```
if(!foundInCache(domainName)){
   cache.push(searchFor(domainName));
 }
return cache(domainName);
```

Fișierele pentru cache vor fi curățate după un anumit interval de timp, acestea stocând structuri de date ce rețin un timestamp pentru fiecare valoare salvată, cu scop în eliberarea spațiului utilizat după un anumit timp de nefolosire. Acest mecanism va rula în paralel sub forma unui copil ce va updata continuu cache-ul aplicației:

```
if(foundInCache(domainName)){
   if(now() - cache(domainName).timeStamp >
maxTimeStamp){
      cache.remove(domainName);
```

Memoria cache va consta dintr-un fișier text în care se vor realiza toate operațiile de prelucrare a datelor.

5. Concluzii:

Așadar, modelul propus al sistemului DNS este unul iterativ. O altă posibilă abordare a acestui sistem este modalitatea recursivă de a realiza căutarea adresei IP a domeniului cerut prin parcurgere din server în server.

O metodă de a spori performanța întregului sistem ar fi utilizarea caching-ului la nivelul mai multor servere pentru eficientizarea căutării.

O altă metodă interesantă de optimizare ar fi căutarea adresei domeniului introdus și în cache-ul rețelelor apropiate, presupunând o oarecare similitudine dintre căutările utilizatorilor celor două rețele.

De asemenea, o mapare pentru redirecționările tuturor serverelor ar crește performanța căutărilor considerabil.

6. Bibliografie:

[1] Explicații - What is DNS (Domain Name System)?

https://www.youtube.com/watch?v=nyH0nYhMW9M&t=365s&ab_channel=IBMTechnology

[2] Clarificări: https://cloudinfrastructureservices.co.uk/what-is-dns-hierarchy/

[3] Detalii: https://ro.wikipedia.org/wiki/Domain_Name_System

[4] Schiță: https://www.appneta.com/assets/Screen-Shot-2018-12-20-at-4.25.01-PM.png