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.

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 răspunsuri 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.

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 clienți și servere.

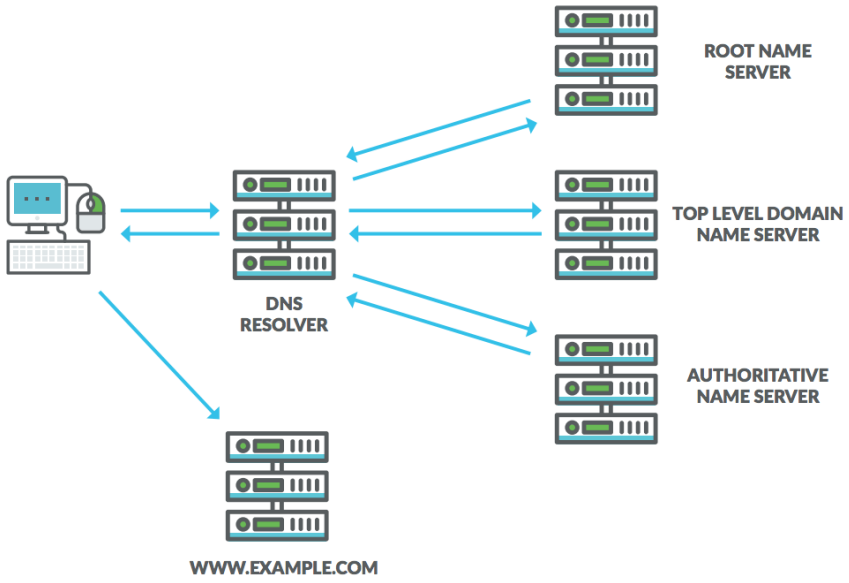
Pentru rularea concurentă a serverelor se va folosi api-ul “create_thread()” care va realiza threaduri pentru fiecare instanță de client care se conectează la server, dar și pentru legăturile paralele dintre servere.

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.

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:



Aplicația propune ambele abordări, atât recursivă cât și incrementală.
În varianta iterativă 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:

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

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

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

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.

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

În varianta recursivă clientul apelează resolverul, iar începând cu acesta fiecare server apelează următorul server până la descoperirea adresei IP, sau până se termină căutarea returnând un cod de domeniu negăsit.

Pașii urmați de întreg sistemul sunt următorii:

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

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

Serverul root va apela serverul top_level.

Serverul top level va apela serverul auth.

De la serverul auth se returnează datele în mod recursiv până la client și resolver, care memorează datele în memoriile lor cache.

Detalii de implementare:

Pentru rularea întregului serviciu trebuie ca toate serverele sa fie pornite: *autthorative_1.sh, autthorative_2.sh, autthorative_3.sh, top_level_com.sh, top_level_net.sh, top_level_org.sh, root.sh, resolver.sh.* Pentru pornirea aplicației trebuie rulat programul client.sh. Acesta va cere utilizatorului domeniul dorit, urmat de tipul de căutare (i-incrementală / r-recursivă). La final, aplicația va afișa ori adresa IP salvată în memoriile cache sau în datele serverelor sau mesajul "404" pentru domeniile care nu au fost înregistrate.

Lista cu domeniile înregistrate:

netflix.net 54.73.148.110
amazon.net 205.251.242.103
intel.org 13.91.95.74
amd.org 88.221.168.84
youtube.com 142.250.184.238
reddit.com 151.101.1.140
wikipedia.com 91.198.174.192
google.com 142.250.184.238
facebook.com 185.60.218.35
samsung.org 211.45.27.231
sony.org 52.54.18.9
apple.net 17.253.144.10
adobe.net 23.45.112.48
nvidia.net 34.194.97.138
instagram.com 185.60.218.174
twitter.com 104.244.42.65
ebay.net 209.140.149.232
microsoft.net 20.203.85.33
nintendo.org 3.218.0.127

mcdonalds.org 93.115.28.104
starbucks.org 34.102.138.180

Serverele vor rula în mod concurent, astfel pentru fiecare nouă conexiune se va realiza un nou thread din threadul principal, folosit exclusiv receptării intenţiilor de conexiune. Astfel de procedeu va fi construit pe baza codului:

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

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

Un astfel de mecanism trebuie susţinut de o metodă pentru terminarea threadurilor inactive pentru a reutiliza resurse stagnante. O soluţie la această problemă poate fi îndepărtarea threadului nou creat de threadul principal folosind api-ul "pthread_detach()", astfel la terminarea executiei threadului resursele acestuia se vor recicla automat de către sistem.

Atât la nivelul clientului cât şi 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);
```

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

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

}}
```

Memoria cache va consta dintr-un fişier text în care se vor realiza toate operaţiile de prelucrare a datelor. Aceasta conţine trei coloane ce memorează domeniul, adresa ip si "timestamp-ul" înregistrării. Acest timestamp este memorat sub forma numărului de milisecunde trecute de la "epoch". Datele sunt şterse din cache când se face o verificare şi se observă că au trecut mai mult de 5 minute de la înregistrare. Verificările se efectuează la finalul fiecărui lookup.

Concluzii:

Aşadar, modelul propus al sistemului DNS este unul atât recursiv cât şi incremental, folosind TCP ca tip de conexiune dintre servere si client.

O metodă de a spori performanţa întregului sistem ar fi utilizarea cachingului la nivelul mai multor servere pentru eficientizarea căutării.

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

De asemenea, o mapare pentru redirecţionările tuturor serverelor ar creşte performanţa căutărilor considerabil.

Bibliografie:

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