IPK – Projekt Č.2

Varianta 3: DNS lookup nástroj

Maroš Orsák (xorsak02)

27.3.2018

Obsah

Problematika DNS(domain name system)

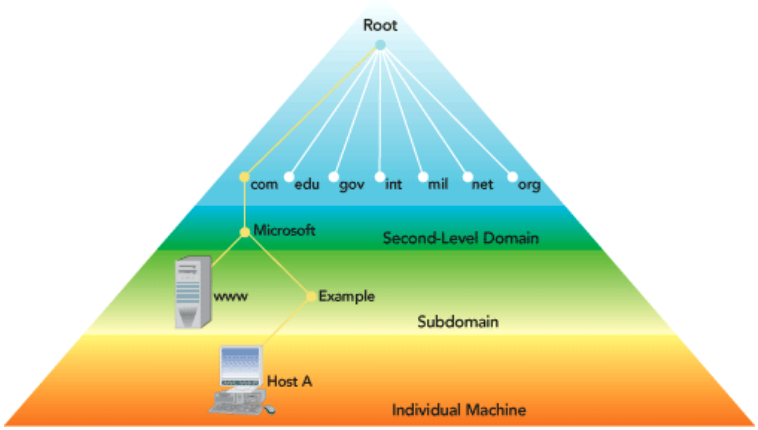
Stručne povedané, Domain Name System(nazývaný DNS) prekladá ľuďské čitateľné názvy domén do IP adries. DNS, domain name server , domain name system, a name server sa vzťahujú na rovnakú všeobecnú službu. Ide o protokol aplikačnej vrstvy pre výmenu správ medzi klientmi a servermi. Prenos svojich paketov necháva transportnim protokolom UDP a TCP.Otázka a odpoveď sú prenášané vždy tým istým protokolom. U otázok na preklad je dávaná prednosť protokolu UDP. V prípade, že je odpoveď DNS serveru dlhšia ako 512B, vloží sa do odpovede iba časť informácie nepresahujúcu túto veľkosť.Komunikácia priebeha na portu 53/UDP a 53/TCP.

Požiadavky

- Každý hostiteľ je identifikovaný podľa adresy IP, ale pamatujúc si, že čísla sú pre ľudí veľmi ťažké a tiež adresy IP nie sú statické , preto je potrebné zmapovať názov domény na adresu IP. Takže DNS sa používa na konverziu názvu domény webových stránok na ich číselnú IP adresu.

Doména

- Existujú rôzne druhy domén:
    - Generické domény:
        - .com(komerčná)
        - .edu (vzdelávacia)
        - .mil(vojenská)
        - .org(nezisková organizácia)
        - .net(podobne ako komerčné)
    - Krajinové domény:
        - .in (india)
        - .us (united states)
        - .uk(united kingdom)
    - Inverzné domény:
        - ak chceme vedieť , aký je názov domény webových stránok.
        - Ip na mapovanie doménových mien
        - Takže DNS môže poskytnúť ako mapovanie napríklad nájsť IP adresy google.com potom musíme v konzoli/termináli použiť nástroj nslookup
            - Príklad : nslookup google.com



Obr.č.1. Organizácia domén

Je veľmi tažké zistiť adresu IP priradenú k webovým stránkam, pretože existujú milióny webových stránok a so všetkými týmito webovými stránkami by sme mali byť schopní generovať IP adresu okamkžite, nemalo by sa stať že by sme čakali príliš dlho.Organizácia databázy je veľmi doležítá.

DNS record(záznam)

Názov domény, adresa IP, platnosť domény , životnosť a všetky informácie týkajúce sa tohoto názvu domény.Tieto záznamy sú uložené v stromovej štruktúre.

Namespace

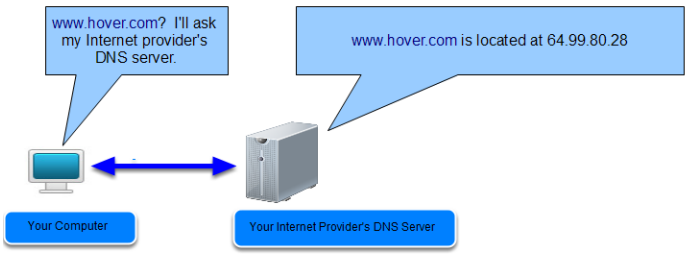
Je to množina možných mien , plochý alebo hierarchický. Pomenovací systém udržiava súbor väzieb názvov na hodnoty – daného názvu , mechanizmus rozlíšenia vráti zodpovedajúcu hodnotu.

Name server

Ide o implementáciu mechanizmu riešenie problémov. DNS (Domain Name System) = názov služby na internete – Zóna je administratívna jednotka, doména je podstrom.

Name to Address Resolution

Hostiteľ požiada DNS name server aby vyriešil názov domény. Name server vráti hostiteľovi IP adresu zodpovedajúcu tomuto názvu domény, aby sa hostiteľ mohol v budúcnosti pripojiť k tejte IP adrese.Toto je v prípade ak už DNS name server túto IP adresu má v cache a teda hneď odpovedá/vracia IP adresu požadovanú doménu.

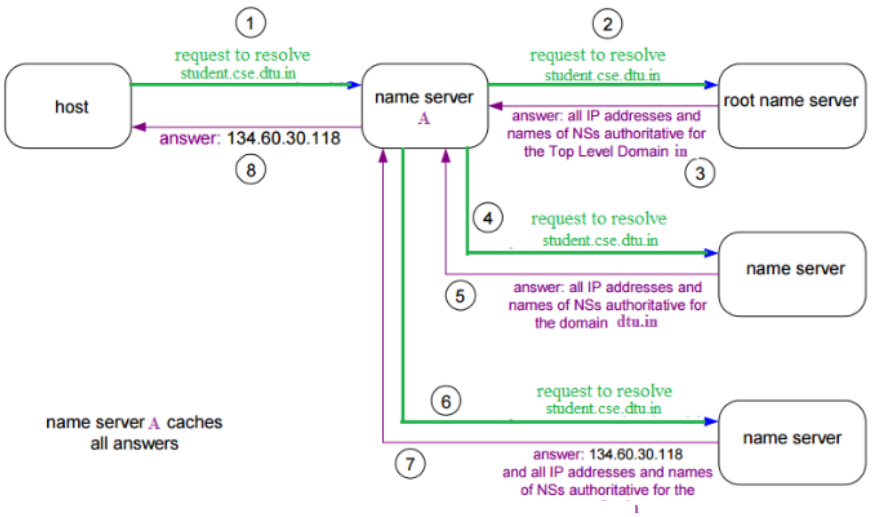


Obr.č.2. Name to address resolution

Hierarchy of Name Servers

- Root name servers
  - sú to servery postavené najvyššie v hierachií
  - pre každú webstránku teda platí že musí mať “root” ktorý je neviditeľný ako .(bodka na konci celého názvu webstránky)
    - príklad: www.google.com.
  - tieto servery sú ovládané 12 rôznymi organizáciami
  - poznáme presne 13 root serverov po celom svete na všetkých 6 kontinentoch pričom ešte každý z nich je mnohonásobne istený.
- Top level server (TLD)
  - sú to servery, ktoré sú zodpovedné pre com, org, edu a všetky ostatné vysoko levelové domény (top level) ako napríklad uk, fr, ca a podobne.
  - Majú informácie o authoritatívnych doménových serveroch a vedia mená a IP adresy každého jedného authoritatívneho serverového názvu pre „second level domains“

- Authoritative name servers
  - jedná sa o server DNS orgánizácie, ktorý poskytuje authoritatívny názov hostiteľa (hostname) na mapovanie IP pre organizačné servery
  - môže ho udržiavať organizácia alebo poskytovateľ služieb
  - aby sme sa dostali do adresára cse.dtu.in, musíme sa opýtať na koreňový server DNS, potom poukáže na doménový server najvyššej úrovne(TLD) a potom na autoritatívny server názvov domén, ktorý skutočne obsahuje IP. Takže autoritatívny doménový server vráti asociačnú IP adresu.



Obr.č.3. Domain Name Server

DNS dotazy

Už v predchádzajúcej kapitole boli zmienené DNS dotazy.O čo v podstate ide sa v tejto časti dozviete.

DNS dotazy sa objavújú pri komunikácií medzi clientom a serverom a taktiež aj pri komunikácií server - server

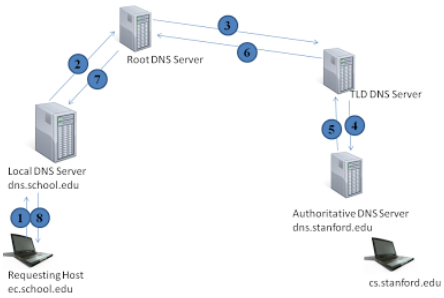
- napríklad: name server ------ autoritativny name server

Popis cyklu:

1. Client(host) sa bude dotazovat a po nás bude chcieť www.google.com
2. Tento dotaz poputuje ku serveru nazývaného ako “name server” a name server sa pozrie do cache pamati
   a. bude mať v pamati(cache) a teda vrati prislušnú IP adresu
   b. nebude mať v pamati(cache) a bude sa musieť pokračovať ďalej v hladani
3. Akonáhle name server zistí, že danú doménu nemá v pamati prejde ku serveru známemu ako “root name server”
4. Bude sa ho pýtať na danú doménu www.google.com on však nebude vedieť ale bude ho odkazovať na další typ serverom a tým sú TLD(top level domain) servers.Name server si všetky potrebné informácie zapíše do cache a pokračuje.

5. Name server sa teraz bude pýtať TLD servers, či pozná www.google.com on však zase nebude vedieť ale odkáže ho na další typ serverom a tým sú „Authoritative name servers“.Name server si zapamatá všetky potrebné informácie do cache a pokračuje.
6. Teraz sa bude pýtať ANS na danú doménu www.google.com v tomto prípade už dané servery budú vedieť o čo sa jedná a vrátia príslušnú IP adresu (napr:8.8.8.8).Name server si toto zapatamá do cache.
7. Name server vracia klientovi príslušnú IP adresu danej domény a teda 8.8.8.8.

Je nutné dodať že akonáhle by sa porušilo jedno spojenie zo zmienených serveroch(ROOT, TLD , ANS) tak by žiadna informmácia neprišla klientovi a vracala by sa chyba.



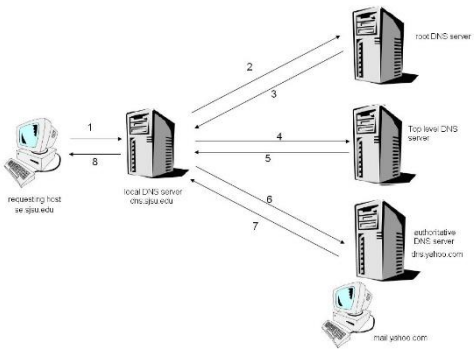
Obr.č.4. Rekurzivny spôsob

Rekurzivny spôsob

- Rekurzivny sposob sa vyskytuje ked client poziada name server(lokálny server) o zistenie ip adresy pre danu domenu.

Iterativny spôsob

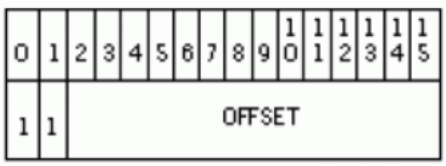
- Iterativny spobo je akonahle name server nenajde v cache prislusnu domenu tak sa bude odkazovat na servery (ROOT, TLD, ANS) a tie mu povedia aku ip adresu ma dana domena.



Obr.č.5. Iteratívny spôsob

Spôsob kompresie správ

- Správa DNS odpovede môže obsahovať rovnaký názov domény niekoľkokrát. Toto opakovanie je plytvanie bitov v správe. Technika kompresie sa môže použiť na zníženie počtu použitých bitov a nahradiť opakovaný názov domény ukazovateľom.
- Kompresné navestie je ukazovateľ, ktorý zaberá pole NAME v sekcii Odpoveď (16 bitov). Takže ukazovateľ je napísaný na 16 bitov a má nasledujúci formát:



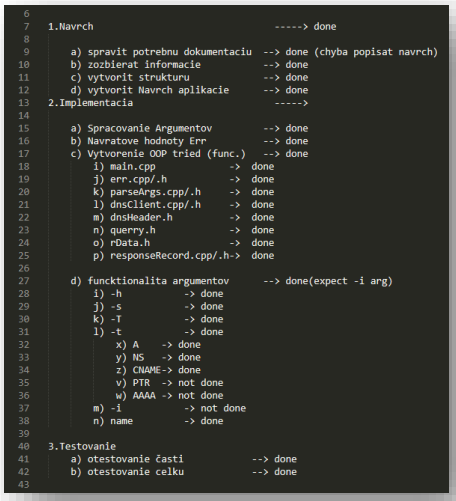
Obr.č.6. Spôsob kompresie správ

- Dĺžka navestia dátového navestia je jeden (byte long) a jeho hodnota je medzi 0 a 63? 63 je 00111111 v binárnom formáte. Navestie s kompresiou však má prvé dva bity nastavené na 1, aby sa odlišovali od navestia s údajmi.
- Kompresné navestie sa môže použiť iba vtedy, ak už bol spomenutý doménový názov (tzv. Kompresný cieľ) už spomenutý v správe DNS (nemôžete poukázať na niečo, čo ešte neexistuje).

- Význam kompresie navestia je nasledujúci: prvé 2 bity sú nastavené na 1, 14 zostávajúcich bitov opisuje posun, t.j. polohu kompresného cieľa od začiatku správy DNS.

Návrch a implemetácia aplikácie

Ako už je v nadpisu jasné vybral som si variantu 3 čo je DNS lookup.Táto aplikácia je založená a inšpirovaná nástrojmi nslookup a dig.



Obr.č.7. Milníky aplikácie

Prvá časť

Prvou časťou bol návrch a teda s tým spojené preštudovanie rozsiahlych informácií či už z platforiem ako youtube alebo google ale taktiež aj z knižných zdrojov ktoré som spomenul. Musím povedať že za zmienku stojí kniha DNS and BIND od Paul Albitza a Cricketa Liu, ktorá mi otvorila celkový nadhľad na danú problematiku. Keď som sa už cítil, že o danom protokole viem pomerne dosť začal som premýšlať ako to celé bude mocť vypadať. Ako už v minulom projekte začal som vytvorením súboru Milníky, ktoré mi pomáhali k dosiahnutiu cielu a podporovali motiváciu. Pri vytváraní milníkoch som o celej scheme návrhu premýšlam a vytvoril som si class diagram. V tejto pozícii už som bol pripravený na dalšiu časť a to implementáciu.

Druhá časť

Druhou časťou bola implemtácia aplikácie, ktorá bola značne tažšou. Začínal som tým, že som si vytvoril parseArgs.cpp a začal som spracovávať argumenty. Bol som veľmi milo prekvapený ako možu regulárne výrazy tak efektívne pomocť.Prikladám obrázok regulárneho výrazu na matchnutie IPv6 a IPv4 adresy. Ostatok až tak zaujimavý nebol a teda takto sa moja implementácia pri spracovaní argumentov ukončila.Dalšiou peknou vecou bolo vytvorenie si tzn. Err.cpp ktorý mi presne vypísal správu aj s príšlušnou hodnotou návratu. Riešenie timeoutu som riešil pomerne dosť dlho až pokým som nenašiel veľmi jednoduchý spobob cez funckiu getTime() a to tak že som predtím ako som išiel odoslať správu serveru tak som si zistil aktuálny čas a uložil ho do pomocnej premenny a potom za poslaní správy som si checkol koľko trvalo posielanie správy a to tak že som si znovu zistil čas a akonáhle ten čas bol vačší ako daný prepínač -T tak som vyhodil chybu.

```
regex e("([0-9a-fA-F]{1,4}:){7,7}[0-9a-fA-F]{1,4}|([0-9a-fA-F]{1,4}:){1,7}:|([0-9a-fA-F]{1,4}:){1,6}:[0-9a-fA-F]{1,4}|([0-9a-fA-F]"
"{1,4}:){1,5}(:[0-9a-fA-F]{1,4}){1,2}|([0-9a-fA-F]{1,4}:){1,4}(:[0-9a-fA-F]{1,4}){1,3}|([0-9a-fA-F]"
"{1,4}:){1,3}(:[0-9a-fA-F]{1,4}){1,4}|([0-9a-fA-F]{1,4}:){1,2}(:[0-9a-fA-F]{1,4}){1,5}|"
"[0-9a-fA-F]{1,4}:((:[0-9a-fA-F]{1,4}){1,6})|:((:[0-9a-fA-F]{1,4}){1,7}|:)|fe80:(:[0-9a-fA-F]{0,4})"
"{0,4}%[0-9a-zA-Z]{1,}|::(ffff(:0{1,4}){0,1}:){0,1}((25[0-5]|(2[0-4]|1{0,1}[0-9]){0,1}[0-9])\\.)"
"{3,3}(25[0-5]|(2[0-4]|1{0,1}[0-9]){0,1}[0-9])|([0-9a-fA-F]{1,4}:){1,4}:((25[0-5]|(2[0-4]|1{0,1}"
"[0-9]){0,1}[0-9])\\.){3,3}(25[0-5]|(2[0-4]|1{0,1}[0-9]){0,1}[0-9]):");
regex ipv4("\\b((25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)(\\.|$)){4}\\b");
```

Obr.č.8-9. Regulárny výraz Ipv4 , Ipv6

```
double sendTime = getTime();
```

```
double elapsedtime = getTime() - sendTime;
if(elapsedtime > ttl){
    throw Err("Spojenie ku serveru zlyhalo",1);
}
```

Obr.č.10-11. Riešenie timeoutu

Tretia časť

Tretiou časťou bolo testovanie celého projektu. Mal som vyčlenené testy na určité časti a potom testy ktoré pokrývali celok projektu.

Návod na použitie

Použitie tejto aplikácie je veľmi jednoduché a taktiež je v implementácií zahrnutá funckia help0, ktorá danému uživateli aké má možnosti pri používaní danej aplikácie.

Príklad č.1 :

- IN

```
./ipk-lookup -s 8.8.8.8 -T 2 www.facebook.com
```

- OUT

```
www.facebook.com IN CNAME star-mini.c10r.facebook.com
star-mini.c10r.facebook.com IN A 185.60.216.35
```

Uživateľ má na výber z prepínačov

- h (help) - voliteľný parameter, pri jeho zadaní sa vypíše nápoveda a program sa ukončí.
- s (server) - povinný parameter, DNS server (IPv4 adresa), na ktorý sa budú odosielať otázky.
- T (timeout) - voliteľný parameter, timeout (v sekundách) pre dotaz, predvolená hodnota 5 sekúnd.
- t (type) - voliteľný parameter, typ respondenta záznamu: A (predvolené), AAAA, NS, PTR, CNAME.
- i (iterative) - voliteľný parameter, vynútenie iteratívního spôsobu rezolúcie, viď ďalej.
- name - prekladané doménové meno, v prípade parametra -t PTR program na vstupe naopak očakáva IPv4 alebo IPv6 adresu.

Záver

Téma DNS je obrovská a jej pochopenie si vyžaduje veľa času. Avšak ak sa jedná o implementáciu tu platí sa toto pravidlo ešte umocnuje. Na záver by som chcel povedať , že DNS protokol sú veľmi obsiahla a zaujimavá téma pri , ktorej nie je možné začať akonáhle s implentáciou.Veľmi efektívnou časťou bolo využitie softwaru wifesharku , ktorý mi pomohol objasniť a taktiež si predstaviť celú abstrackciu danej problemiky.