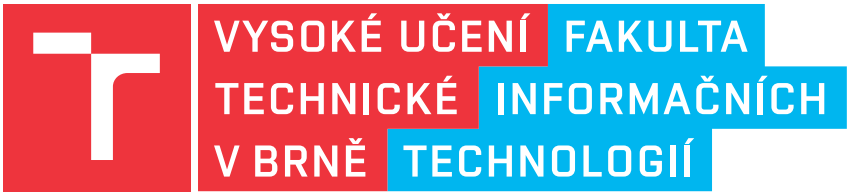


ISA – Sieťové aplikácie a správa sieti

Tunelovanie dátových prenosov cez DNS dotazy

13. novembra 2022

Patrik Sehnoutek (xsehno01)

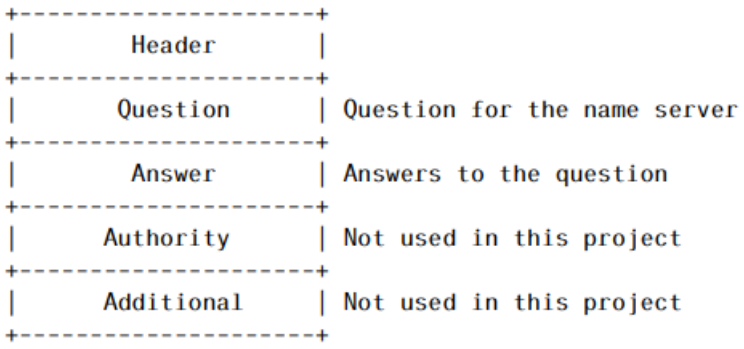
Obsah

1 O projekte

V dokumente je popísaná implementácia serverovej a klientskej aplikácie, ktoré sú implementované v jazyku C. Ich úlohou je komunikovať medzi sebou a simulovať tunelovanie dátových prenosov pomocou DNS dotazov.

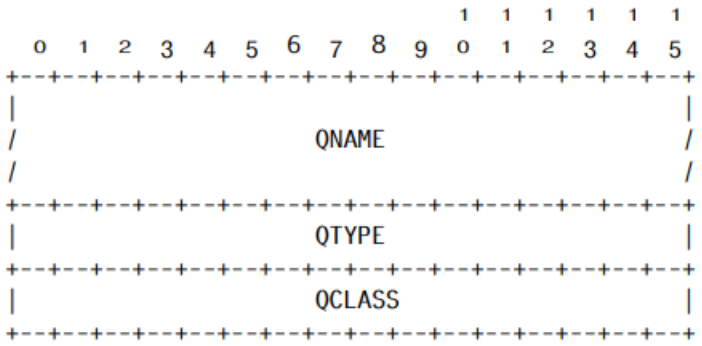
2 DNS protokol

DNS (Domain Name System) je protokol slúžiaci primárne na preklad doménových mien na IP adresy. DNS paket sa skladá z 5 sekcii. Prvou sekciou je hlavička **(Header)**, v ktorej je zahrnutý typ paketu a aké ďalšie sekcie obsahuje DNS paket. Za hlavičkou nasleduje niekoľko sekcii dotazov **(Question)**, odpovedí **(Answer)**, autoritatívnych name serverov **(Authority)** a doplňujúcich informácií **(Additional)**.



Obr. 1: Štruktúra DNS paketu

Sekcia **Answer** sa delí na ďalšie tri časti. Pre tento projekt je najdôležitejšia časť QNAME, ktorá obsahuje doménové meno a taktiež v nej budeme posielať zakódvané dáta. V časti QNAME sa nepoužíva bodková notácia doménového mena. Upravené doménové meno sa skladá z čísiel reprezentujúcich dĺžku nasledujúceho reťazca, textových reťazcov a je zakončený nulovým znakom. Maximálna dĺžka reťazca je 63 znakov. Bližšie podrobnosti k DNS paketu nájdete na dokumente RFC1035[4].



Obr. 2: Štruktúra DNS question

3 DNS tunelovanie

DNS tunelovanie[1] je metóda, ktorou sme schopný posielať akékoľvek dáta cez DNS protokol. DNS dotazy sú zneužívané, tak, že do nich vložíme jeden smer komunikácie a v odpovediach bude zahrnutý druhý smer

komunikácie. DNS protokol na to nebol pôvodne navrhnutý a tento typ útoku býva často prehliadnutý.

Útočníci musia mať k dispozícií doménu vo verejnom doménovom strome a server, ktorý sa bude správať ako autoritatívny server. Útočníci následne infektujú počítač, ktorý "sedí"za bránou firewall. DNS requesty sú vždy povolené firewallom, a preto je infektovaný počítač schopný posielať DNS query tzv. DNS resolveru. DNS resolver prepošle DNS query na útočníkov autoritatívny server. Tento server neodpovedá bežných spôsobom, ale do odpovedí vkladá dáta, ktoré nám chce poslať. Celá komunikácia je autorizovaná, takže DNS resolver preposiela všetko ďalej.

4 Návrh a implementácia

DNS komunikácia používa na transportnej vrstve protokol UDP a dotazy sú posielané na port 53. Implementovaná klientská aplikácia teda predstavuje UDP klienta a serverová aplikácia UDP server[2]. Na vytvorenie UDP klienta a serveru slúžia funkcie z knižnice **<sys/socket.h>**.

4.1 Hlavičkové súbory

- **<stdlib.h>, <stdio.h>, <string.h>, <sys/types.h>** - štandartné knižnice jazyka C
- **<arpa/inet.h>** - definície pre internetové operácie, funkcie ntohl, ntohs a štruktúry pre porty a adresy
- **<sys/socket.h>** - funkcie, makrá a konštanty pre prácu so soketmi
- **dns_sender_events.h** - deklarácia preddefinovaných funkcií na testovanie a funkcií potrebných na fungovanie klientskej aplikácie
- **dns_receiver_events.h** - deklarácia preddefinovaných funkcií na testovanie a funkcií potrebných na fungovanie serverovej aplikácie
- **error.h** - chybové kódy
- **dns.h** - štruktúra pre načítané dáta a štruktúru DNS hlavičky. Taktiež sú v ňom zahrnuté použité konštanty.

4.2 Klientská aplikácia

Postup vykonávania klientskej aplikácie:

1. Načítanie, spracovanie a uloženie programových parametrov pomocou funkcie `checkParameters()`.
2. Inicializovanie soketu a nastavenie jeho parametrov. V parametroch soketu sa taktiež nastaví dĺžka časového intervalu, počas ktorého bude klientská aplikácia čakať na odpoveď od servera.
3. Vytvorenie bufferu slúžiaceho pre DNS paket.
4. Vytvorenie DNS hlavičky a uloženie na začiatok bufferu.
5. Načítanie dát zo súboru alebo zo štandartného vstupu.
6. Pokiaľ bol zadaný parameter `UPSTREAM_DNS_IP` pokračuje sa na bod č. 7. Načítanie IP adresy predvoleného DNS serveru nastaveného v systéme zo súboru `/etc/resolv.conf`.
7. Konvertovanie domény `BASE_HOST` do formátu akceptovateľného v sekcii QNAME pomocou funkcie `changeHostToDnsFormat()`.
8. Zistenie dĺžky konvertovanej domény a vypočítanie voľného priestoru pre vstupné (nezakódované) a zakódované dáta v sekcii QNAME.

9. Zobratie vstupných dát o maximálnej možnej dĺžke na základe výpočtu z predchádzajúceho kroku. Zakódovanie dát pomocou funkcie `base32_encode()` a konvertovanie do prijateľného formátu v sekcii QNAME. Zakódované dáta sú rozdelené do sekcií po maximálne 63 bajtoch (1 bajt = 1 znak). Pred každou sekciou je číslo identifikujúce počet bajtov v danej sekcii.
10. Uloženie dát a konvertovanej domény do bufferu za zakódované dáta.
11. Nastavenie QTYPE a QCLASS.
12. Odoslanie DNS paketu pomocou funkcie `sendto()`.
13. Čakanie na odpoveď od serverovej aplikácie. Po uplynutí času na prijatie odpovede sa pokračuje bodom č. 12. Pri prijatí odpovede pomocou funkcie `recvfrom()` sa program ukončí, ak boli odoslané všetky dáta. Ak neboli odoslané všetky dáta, program pokračuje bodom č. 9.

4.3 Serverová aplikácia

Postup vykonávania serverovej aplikácie:

1. Načítanie, spracovanie a uloženie programových parametrov pomocou funkcie `checkParameters()`.
2. Inicializovanie soketu `socket()` a nastavenie jeho parametrov `setsockopt()`.
3. Konvertovanie domény `BASE_HOST` do formátu akceptovateľného v sekcii QNAME pomocou funkcie `changeHostToDnsFormat()`.
4. Prijatie DNS paketu pomocou funkcie `recvfrom()`.
5. Porovnanie domény zo sekcii QNAME a domény, ktorá bola zadaná ako parameter programu. V pripade nezhody pokračovanie bodom č. 4.
6. Dekódovanie sekcie QNAME pomocou funkcie `base32_decode()`. Spracovanie dekódovaných dát na základe typu paketu.
   (a) init paket – vytiahnutie cesty z dekódovaných dát, kde sa budú ukladať prijaté dáta, vytvorenie potrebných adresárov `createDirectory()` a otvorenie súboru pre zápis
   (b) dátový paket – uloženie dát do súboru
   (c) end paket – zatvorenie súboru
7. Odoslanie odpovedi klientovi pomocou funkcie `sendResponse()`. Pokračovanie bodom č. 4.

4.4 Komunikačný protokol

Zakódované dáta sa posielajú v DNS question v sekcii QNAME spoločne s doménou. Implementoval som tri rozdielne tipy DNS paketov, ktoré sa líšia obsahom sekcie QNAME.

- Komunikácia klienta so serverom začína poslaním špecialnej správy v QNAME, ktorá obsahuje reťazec `"DST_FILEPATH[%s]"`, `%s` je nahradené parametrom `DST_FILEPATH` z klientskej aplikácie. Server na základe tohoto reťazca zistí, že ide o prvý paket a vytiahne si z neho cestu k súboru. Uloženú cestu k priečinku spojí z cestou k súboru. Pokiaľ je to nevyhnutné vytvorí požadované podpriečinky a súbor otvorí pre zápis.
- Nasledujú dátové pakety. V sekcii QNAME sa nachádzajú iba kódované dáta spoločne s doménou. Server dáta dekóduje a zapíše do súboru.

- Posledný paket je opäť špecifický a v sekcii QNAME obsahuje reťazec "[END_CONNECTION]". Na základe tohoto reťazca vie server, že už neprídu žiadne ďalšie dáta od klienta a zatvorí súbor, do ktorého zapisoval dáta[3].

Server posiela pri každom prijatom DNS pakete odpoveď klientovi. Odpoveď je opäť DNS paket, obsahujúci skopírovanú DNS question od klienta nasledovanú odpoveďou DNS answer. Odpoveď slúži na overenie, že server prijal a spracoval DNS question. Klient má nastavený časový interval, v ktorom čaká na prijatie odpovedi. V prípade neúspechu, tj. neprijatí odpovedi v časovom intervale, klient opätovne odošle paket.

4.5 Kódovanie dát

Dáta som kódoval pomocou kódovacieho algoritmu Base32[5]. V jazyku C som na to využil knižnicu `base32.h` spoločne so zdrojovým kódom `base32.c`. Dáta sa v klientovi pred odoslaním zakódujú pomocou funkcie `base32_encode` a následne sa v aplikácii serveru pri prijatí dekódujú pomocou funkcie `base32_decode`.

Kódovací algoritmus Base32 využíva sadu 32 tlačiteľných znakov. Najčastejšia znaková sada je zložená z písmen A – Z nasledovaných číslicami 2 – 7.

5 Návod na použitie

Všetky príkazy spúšťajte v koreňovom adresári projektu.

5.1 Preklad

Klientskú aj serverovú aplikáciu je najprv nutné preložiť. Na preklad slúži súbor `Makefile`. Príkazom `make` alebo `make all` preložíte obidve aplikácie súčasne. Príkazom `make sender` preložíte iba klientskú aplikáciu a príkazom `make receiver` zas iba serverovú aplikáciu.

5.2 Spustenie

```
$ make

$ sudo ./dns_receiver {BASE_HOST} {DST_DIRPATH}

       {BASE_HOST}          is used to set the base domain to receive data
       {DST_DIRPATH}        path under which all incoming data/files will be saved (the
           path specified by the client will be created under this directory)

$ sudo ./dns_sender [-u UPSTREAM_DNS_IP] {BASE_HOST} {DST_FILEPATH} [SRC_FILEPATH]

       -u UPSTREAM_DNS_IP   is used to force a remote DNS server, default value: DNS
           server set in the system
       {BASE_HOST}          is used to set the base domain of all transfers, i.e. queries
            will be sent to addresses *.{BASE_HOST}
       {DST_FILEPATH}       path under which the data will be saved on server
       [SRC_FILEPATH]       path to the file that will be sent, if not specified
           application reads data from STDIN
```

5.3 Príklady spustenia

Otvorte dve terminálové okná. V jednom spustite klientskú aplikáciu a v druhom serverovú aplikáciu.

Spustenie serverovej aplikácie:

```
$ sudo ./dns_receiver example.com test-directory/
```

Spustenie klientskej aplikácie:

```
$ sudo -u 127.0.0.1 ./dns_sender example.com output-data.txt input-data.txt
```

Pri jednom spustení sa odošlú dáta zo štandartného vstupu alebo špedifikovaného súboru. Pre odoslanie ďalších dát je potrebné opätovné spustenie klienta.

6 Testovanie

Na testovanie vytvorených aplikácií som použil program Wireshark. Testovanie som realizoval na operačnom systéme Ubuntu 20.04 na sieti lo (loopback). V mojej aplikácii som na špecifických miestach zavolal funkcie rozhrania aplikácie, ktoré služia na hodnotenie projektu. Porovnával som výstupy funkcií a obsahy paketov vo Wiresharku.

Vo Wiresharku som taktiež mohol otestovať, či mám dáta v DNS pakete v správnom formáte. Na základe chybových správ z Wiresharku sa dalo ľahko určiť, čo je potrebné zmeniť v DNS pakete, napríklad zlý formát sekcie QNAME alebo rovnaké ID všetkých DNS paketov.

7 Obmedzenia

- aplikácia nepodporuje protokol IPv6
- server nie je konkuretný, čiže nie je implementované paralélne spracovanie dotazov z rôznych klientských aplikácií

8 Literatúra

[1] Dizdar, A.: DNS Tunneling: How it Works, Detection and Prevention. [online], 2021-10-11, [vid. 2022-10-24].
Dostupné z: https://brightsec.com/blog/dns-tunneling/

[2] Geeksforgeeks: UDP Server-Client implementation in C. [online], 2022-05-04, [vid. 2022-10-25].
Dostupné z: https://www.geeksforgeeks.org/udp-server-client-implementation-c/

[3] Krčmář, P.: Tunelujeme provoz pomocí DNS: cesta ven ze sítě. [online], 2022-05-26, [vid. 2022-10-31].
Dostupné z: https://www.root.cz/clanky/tunelujeme-provoz-pomoci-dns-cesta-ven-ze-site/

[4] Mockapetris, P.: DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION. [online], 1987, [vid. 2022-10-25].
Dostupné z: https://www.ietf.org/rfc/rfc1035.txt

[5] O'Whielacronx, Z.: Human-oriented base-32 encoding. [online], 2009, [vid. 2022-10-25].
Dostupné z: http://philzimmermann.com/docs/human-oriented-base-32-encoding.txt