VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

TUNELOVANIE DÁTOVÝCH PRENOSOV CEZ DNS DOTAZY

DNS TUNNELLING

SEMESTRÁLNÍ PROJEKT
TERM PROJECT

AUTOR PRÁCE **PAVEL KRATOCHVÍL**
AUTHOR

BRNO 2022

Obsah

1 Úvod do problematiky **2**
1.1 Systém DNS 2
1.2 Riziká systému DNS 2

2 Popis fungovania **3**
2.1 Spôsob exfiltrácie v DNS pakete 3
2.2 Kódovanie dát 4
2.3 Potvrdenie príjmu dát 4
2.4 Odosielanie dát 5
2.5 Klientská časť aplikácie 5
2.6 Serverová časť aplikácie 6
2.7 Testovanie 7

Literatura **9**

Kapitola 1

Úvod do problematiky

1.1 Systém DNS

Systém DNS(Domain Name System) sa používa na preklad doménových mien na IP adresy. V praxi sa užívateľ dotazuje na doménové mená, ktoré su ľahko čitateľné a zapamätateľné, napríklad pri otvorení webového prehliadača a vyhľadaní adresy `www.google.com`. Následne prebieha preklad doménového mena na IP adresu v sérií DNS požiadavkov na nakonfigurovaný DNS server na danom počítači a v prípade, ak nedostane vyžadovaný preklad pokračuje v dotazovaní na servery v hierarchickej štruktúre DNS. Tento systém je veľmi robustný a rezilientný voči útokom.

Keďže ide v dnešnej dobe o základnú službu, ktorá uľahčuje používanie internetu, DNS pakety obsahujúce dotazy a odpovede sa stali takmer samozrejmou súčasťou internetovej prevádzky a môžeme ich nájsť ľahko nájsť vo výstupoch sieťových analyzátorov(napr. Wireshark). Využívanie systému DNS vyžaduje, aby mohli prechádzať cez rôzne stupne zabezpečenia internej siete do globálneho internetu. Na rozlíšenie DNS paketov vo výpise slúži niekoľko jeho charakteristických znakov ako napríklad: pakety využívaju nespoľahlivý protokol UDP, sú posielané na cieľový port 53 ako aj telo UDP paketu, ktoré má stanovenú štruktúru[1].

1.2 Riziká systému DNS

Keby chcel niekto dostať citlivé dáta von zo zabezpečenej siete, tak na to má niekoľko možností. Ak pominieme možnosť využiť fyzické médium, zostala by mu jediná možnosť - využiť sieťové služby a odoslať ich prostredníctvom internetu. Na to by mohol využit mnoho rôznych protokolov, prípadne aj dodatočné šifrovanie ich obsahu(napr. TLS), no excesívna prevádzka odchádzajúcich paketov by mohla pôsobiť podozrivo. Práve DNS pakety sa dajú využiť na exfiltráciu citlivých dát vytváraním dotazov na vonkajšie DNS servery relatívne nenápadne. Neblokovanie prichádzajúcich a odchádzajúcich DNS paketov, prípadne absencia kontroly ich obsahu predstavuje hrozbu pre zabezpečenie siete. No keďže prevádzka DNS často nie je monitorovaná, vytvára to priestor pre typ útoku s názvom DNS Tunnelling(Tunelovanie dátových prenosov cez DNS dotazy). Tento typ zraniteľnosti a útoku naň bol prvý krát publikovaný v roku 1998[2] a bude sa ňou zaoberať tento projekt.

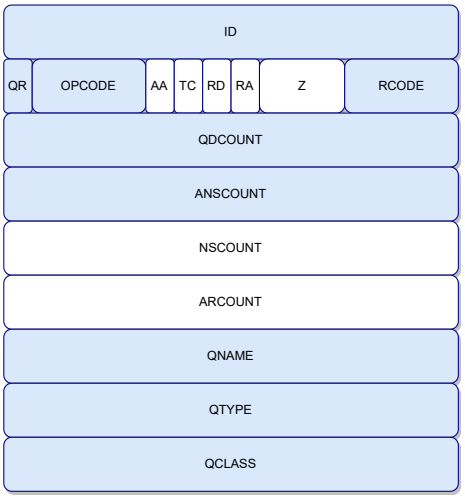
Kapitola 2

Popis fungovania

Projekt pozostáva z dvoch častí - odosieľateľa(`dns_sender`), prijímateľa(`dns_receiver`) a testovacieho programu(`dns_tester`). Časť `dns_sender` predstavuje klienta na internej sieti odosielajúceho dáta a `dns_receiver`, ktorý plní funkciu DNS serveru. Dáta môžu byť v akomkoľvek formáte, teda aj binárne. To znamená, že ich je potrebné zakódovať do znakov povolených v DNS pakete. V rámci komunikácie medzi odosieľateľom a príjemcom musí tiež prebehnúť odoslanie názvu pre prenášaný súbor a tiež správa na konci prenosu o odoslaní všetkých dát.

2.1 Spôsob exfiltrácie v DNS pakete

Jediné miesto v DNS pakete, kam sa dá uložiť vašie množstvo dát je do dotazovanej domény, v terminológii DNS systému nazývanej `QNAME`[1]. Podľa definície formátu DNS paketu, je možné sa dotazovať aj na viacero domén, ktorých počet je potom zaznamenaný v poli `QDCOUNT`. Keďže sa ale v praxi toto pole ako ani možnosť dotazovať sa na viacero domén nepoužíva, rozhodol som sa na dotazovaní sa v každom pakete iba na jedno doménové meno.



Obrázek 2.1: Vizualizácia štruktúry DNS paketu so zvýraznenými poliami, s ktorými pracujem v jednotlivých častiach zadania.

2.2 Kódovanie dát

Na to, aby bol DNS paket valídny musí mať správnu štruktúru, správne kódovanie a dotazovaná doména musí byť tzv. Fully Qualified Domain Name(FQDN). Doménové meno spĺňujúce tento štandard musí pozostávať iba z alfanumerický znakov a pomlčky, maximálny počet znakov medzi oddeľovačom(bodkami) je 63 a jej celková dĺžka nesme presiahnuť 255 znakov.

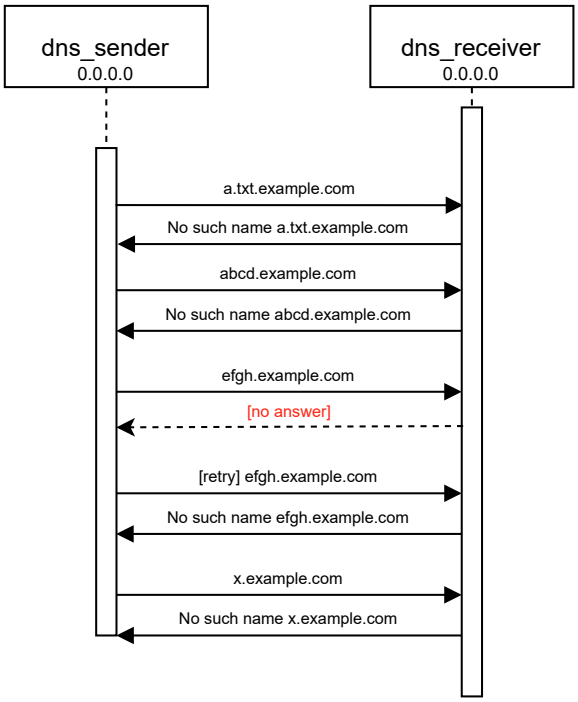
Jedným z povinných parametrov oboch programov je `base_host`, ktorý predstavuje koreňovú doménu pre všetky dotazované domény. To znamená, že ak by moje dáta na odoslanie boli reťazec `aaa` a zadaný `base_host` by bol `example.com`, dotazovaná doména by bola `aaa.example.com`. Parameter `base_host` musí byť na oboch stranách rovnaký pre správne oddelenie dát od koreňovej domény. V mojom projekte je celková dĺžka parametru `BASE_HOST` obmedzená na strane klienta na 64 znakov. Na rovnakú dĺžku(64 znakov) je obmedzený aj parameter `DST_FILEPATH`.

Pre kódovanie dát som zvolil vlastnú implementáciu kódovania `base16`, ktorá rozdelí každý binárny znak na dva a po posune do alfanumerického rozsahu sa premietne do znakov `a` až `p`. Toto kódovanie som zvolil kvôli jednoduchosti implementácie ale bolo potrebné ošetriť prípad keby sa do dátovej časti zmestil iba nepárny počet znakov. V tom prípade by na strane odosieľateľa nebolo možné správu dekódovať, keďze by mu chýbali 4b posledného znaku. Preto v každej správe odosielam párny počet zakódovaných znakov.

Pred odoslaním dát musia byť prevedené do špeciálneho DNS formátu, v ktorom je doménové meno pozmené spôsobom, že každý oddeľovač(bodka) je nahradený počtom znakov, ktoré na ním následuje. Tento počet, nazývaný tiež `length octet` má veľkosť presne jedného bajtu.

2.3 Potvrdenie príjmu dát

Na potvrdenie príjmu dát odosielam zo strany príjemcu využívam pole DNS hlavičky `RCODE` určeného na kód odpovede. Ako odpoveď som zvolil kód `4`, teda že vyhľadávanie takéhoto DNS dotazu nie je implementované. Keďže v odosielaných paketoch vypínam rekurziu vyhľadávania, dotazovaný DNS server sa nebude ďalej dotazovať v stromovej štruktúre DNS serverov. Takiež v potvrdzovacích paketoch mením pre zachovanie validity prenosu hodnotu v poli `QR` na `1`, čo značí odpoveď DNS serveru. Odosieľateľ po prijatí potvrdzovacej odpovede pokračuje odoslaním ďalšieho paketu.



Obrázek 2.2: Príklad komunikácie pre prenos zakódovaného reťazca "abcdefgh". Pre demonštráciu sú odoslané 2 malé pakety, zároveň na jeden paket nedostal odosieľateľ odpoveď.

2.4 Odosielanie dát

Komunikáciu začína vždy klient(`dns_sender`). Pri nepotvrdení príjmu paketu zo serverovej časti sa opakuje odoslanie a príjem potvrdenia. Klientská aj serverová časť majú každá vlasntý počet sekúnd na vypršanie požiadavku. Pre `dns_receiver` som nastavil 20 s a pre `dns_sender` sú nastavené 3 s. Tým pádom ak po zahájení prenosu nedostane server 20 s ďalší paket, prenos sa ukončí a čaká na ďalšie zahájenie prenosu. Pre klientskú časť vypršanie limitu nastáva ak dané 3 s nedostane potvrdzujúcu odpoveď, potom následujú ešte dva pokusy(každý opäť s 3 s limitom). Ak ani potom nedostane potvrdenie, program sa ukončí.

2.5 Klientská časť aplikácie

Program na strane klienta na začiatku vykonáva niekoľko kontrol, ako napríklad kontrolu dĺžky parametra `base_host` a tiež či obsahuje iba povolené alfanumerické znaky, prípadne pomlčku a či nezačína číselným znakom(FQDN nesmie začínať číslom). Program pracuje stream-ovo, teda žiadne dáta okrem aktuálne odosielaného paketu neukladá do pamäti. To umožňuje rýchlejší beh programu ako aj úsporu operačnej pamäte.

Odosielanie paketov prebieha iteratívne, teda `dns_sender` odosieľa ďalší paket až po potvrdení prijatia od príjemcu. V prvom pakete z klientskej časti sa nachádza meno cieľového súboru do ktorého sa má obsah na strane príjemcu uložiť. Prenos je úspešne ukončený posledným paketom, ktorých v dátovej časti odošle jediný znak `x` následovaný ešte reťazcom `base_host`.

2.6 Serverová časť aplikácie

Serverová časť aplikácie, program `dns_receiver` beží v nekonečnom cykle a čaká na prvý paket od odosieľateľa. Po príjme overí či sa sufix zhoduje s parametrom `base_host` zadaného pri spustení programu, aby príjemnca nezachytil a neukladal pakety iné než od našeho odosielateľa. Po kontrole následuje dekódovanie a keďže v každom pakete sa nachádza iba párny počet znakov, je možné dáta hneď dekódovať do pôvodnej podoby a uložiť do otvoreného súboru. Vyhneme sa tak ukladaniu dát do pamäti, čo zrýchľuje beh programu.

Pri spustení `dns_receiver` sa zadáva povinný parameter `DST_DIRPATH`, ktorý špecifikuje priečinok, do ktorého sa budú všetky prijaté dáta ukladať pod cestou, ktorá sa špecifikuje pri odosielaní v parametri `dns_sender`. Ak priečinok `DST_DIRPATH` neexistuje, tak sa vytvorí a nastavia sa preň správne práva. Ak existuje, overí možnosť zápisu. V prípade ak cesta k danému súboru zadaná na strane klienta v parametri `DST_FILEPATH` neexistuje, `dns_receiver` ju vytvorí. Ak už existuje súbor s rovnakým menom pod danou cestou, súbor sa prepíše.

2.7 Testovanie

Pre testovanie som implementoval ďalšiu aplikáciu(`dns_tester`), ktorá pri správnej konfigurácii klientskej aj serverovej časti slúži ako middleman. Prijaté pakety náhodne zahadzuje, čím testujem správne fungovanie opakovaného posielania pri absencii potvrdenia. Ďalej som testoval zmenu identifikácie potvrdzovacieho paketu, čo simuluje oneskorenie odoslania potvrdenia predošlého paketu. Program má jediný voliteľný pozičný parameter `UPSTREAM_DNS_IP`, ktorý špecifikuje IP adresu prijímateľa(`dns_receiver`). Pri nezadaní sa budú pakety preposielať na adresu `127.0.0.1`.

```
[LOG]      Socket for communication with receiver created
[LOG]      Listening to sender on 0.0.0.0:1645
[RECEIVE]  Data received from 127.0.0.1, port 60513
[DROP]     Dropping packet
[RECEIVE]  Data received from 127.0.0.1, port 60513
[SENT]     Sent receiver and confirmed back to sender
[RECEIVE]  Data received from 127.0.0.1, port 60513
[CHANGE]   Changing packet id
[RECEIVE]  Data received from 127.0.0.1, port 60513
[SENT]     Sent receiver and confirmed back to sender
[RECEIVE]  Data received from 127.0.0.1, port 60513
[CHANGE]   Changing packet id
```

Obrázek 2.3: Ukážka výstupu testovacieho programu `dns_tester`, na ktorej môžeme vidieť zahodenie(`DROP`), zmenu ID(`CHANGE`) ako aj úspešné preposlanie paketu(`SENT`).

Závěr

Pri vypracovávaní tohto projektu som sa bližšie zoznámil so systémom DNS, jeho špecifikáciou a pravidlami a tiež s implementáciou klientskej aj serverovej časti komunikujúcich protokolom UDP. Najväčšou výzvou bol výber správneho kódovania pre zachovanie validity paketov a ošetrenie všetkých detailov formátu a štruktúry valídneho DNS paketu.

Literatura

[1] Mockapetris, P. *RFC 1035 Domain Names - Implementation and Specification.* Internet Engineering Task Force, November 1987. Dostupné z: http://tools.ietf.org/html/rfc1035.

[2] Pearson, O. *DNS tunnel - through bastion hosts.* Apr 1998. Dostupné z: https://seclists.org/bugtraq/1998/Apr/79.