VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta informačných technológií

Síťové aplikace a správa sítí

2022/2023

Projekt ISA

Tunelování datových přenosů přes DNS dotazy

Králik Dalibor (xkrali20) Brno 14.11.2022

Obsah

1. Problematika

DNS (Domain name system) je decentralizovaný systém doménových mien, ktorý je realizovaný DNS servermi a protokolom DNS. Jeho hlavnou úlohou je preklad doménových mien na IP adresy a naopak.

DNS tunneling je metóda hackerského útoku, kedy útočník môže pomocou infikovaného počítača v internej sieti vynášať data a prijímať ich pomocou svojho servera. Hacker potrebuje ovládať doménu a server, ktorý bude pôsobiť ako autoritatívny DNS server, aby mohol spustiť programy tunelovania. Vzhľadom na to, že DNS dotazy sú vždy povolené prejsť skrz firewall, infikovaný počítač je schopný posielať DNS dotazy na DNS resolver, ktorý tieto dotazy pošle na útočníkov server a prijme ich. Jedná sa teda o tunelovanie dát – získavania a extraktovanie informácií. [1]

2. Návrh aplikácie

2.1. DNS sender

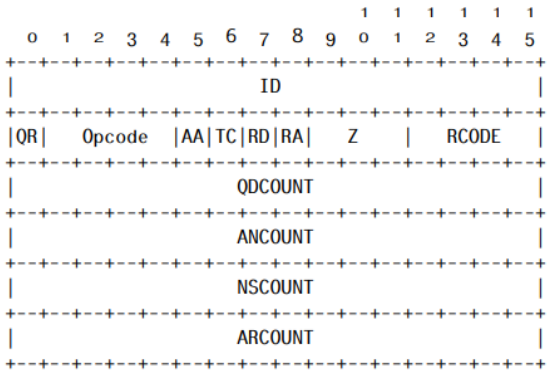
Hlavnou úlohou DNS senderu je poslať UDP packet, ktorý obsahuje DNS packet s dátami pomocou DNS servera, ktorý bol zadaní ako parameter programu dns_sender, alebo DNS servera zo súboru resolv.conf. Následne sa tento UDP packet pošle na DNS port - port číslo 53.

Program musel vytvoriť korektne UDP packet viz [2] s DNS packetom a poslať ho.

DNS packet sa skladá z viacerých častí:

1. DNS header
2. DNS question
3. DNS answer
4. Authority
5. Additional

2.1.1. DNS header

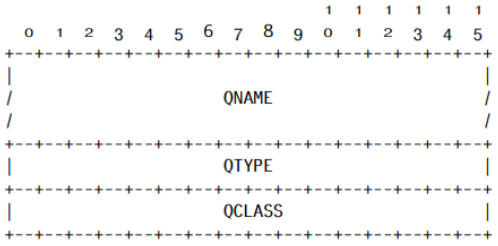


Obrázok 1: DNS Header

DNS Header popisuje typ packetu a to, čo daný DNS packet obsahuje víz [3].

Pre DNS Header som v projekte vytvoril dátovú štruktúru, ktorú som potom postupne napĺňal dátami.

2.1.2. DNS question



Obrázok 2. DNS question

DNS question obsahuje 3 polia pre dáta (QName, QType, QClass).

Pole QType špecifikoval typ DNS požiadavku. V tomto projekte som využil hodnotu htons(1), ktorá reprezentovala A záznam.

Pole QClass špecifikovalo triedu DNS požiadavku. V tomto projekte som použil hodnotu htons(1), ktorá reprezentovala *Internetové adresy*.

QName je pole, ktoré ma maximálnu dĺžku 255 bytov a obsahuje dáta s bázovou doménou, na ktorú sa má DNS packet posielať. V QName som posielal zašifrované dáta spoločne s bázovou doménou. Avšak pre správne formátovanie dát v QName bolo potrebné previesť tieto dáta to tzv. DNS formátu. Tento formát vyzerá tak, že dáta musia byť v substringoch dlhých maximálne 63 znakov a pred týmto substringom sa musí nachádzať číslo počtu znakov v substringu prevedené do hexadecimálneho formátu. Jedine v takomto prípade majú dáta v QName správny formát a software ako Wireshark dokáže QName rozložiť a vyčítať z neho dáta.

Dáta v QName som šifroval pomocou šifry BASE32, ktorej knižnicu pre C napísal Markus Gutschke, ktorá podlieha *the Apache License, Version 2.0.* a bola voľne dostupná na stiahnutie.

Vzhľadom na to, že DNS sender posielal len tzv. DNS Question, nebolo nutné sa v tomto prípade starať o DNS Answer.

Po odoslaní UDP packetu DNS sender čakal, kým sa mu vráti UDP packet od DNS receivru s DNS answer. DNS sender mal nastavený tzv. timeout, po ktorého uplynutí sa predpokladalo, že UDP packet nedorazil k DNS receivru a DNS sender zopakoval poslanie packetu s datami. Tento timeout má DNS sender nastavený na 3 sekundy. Timeout som nastavil pomocou funkcie setsockopt().

2.2. DNS receiver

Cieľom DNS receivru je prijať UDP packety na porte 53 z ľubovolnej zdrojovej adresy, vytiahnuť dáta z DNS QNAME, dešifrovať ich pomocou knižnice base32.c, dáta uložiť do súboru a poslať validnú odpoveď -> UDP packet s DNS packetom a DNS Answer.

DNS receiver beží v nekonečnom while cykle, ktorý prijíma UDP packety pomocou funkcie recvfrom(). Po prijatí UDP packetu tento packet rozloží a vytiahne z neho potrebné dáta. DNS receiver skontroluje, či prijal packet so správnou bázovou doménou. Ak táto bázová doména bola správna, začal spracovávať dáta, ktoré dešifruje a uloží do súboru. Následne vytvorí nový UDP packet, ktorý obsahuje DNS odpoveď. Náplní buffer štruktúrou DNS Header, DNS Question a následne DNS Answer.

Po naplnení bufferu tento UDP packet pošle na adresu, z ktorej prijal UDP packet s DNS Question.

Vzhľadom na to, že beží v nekonečnej slučke, tak celý tento proces opakuje, až kým nie je jeho program ukončený pomocou *CTRL+C*.

2.3. Komunikačný protokol

DNS sender neposielal len dátové packety. Pred poslaním prvého dátového packetu poslal DNS sender tzv. inicializačný packet, ktorý slúžil na inicializáciu spojenia a poslanie inicializačných dát (FILE_PATH) DNS receivru. DNS receiver si z QNAME inicializačného packetu zobral FILE_PATH (cestu k súboru, do ktorého sa majú ukladať dešifrované dáta). Dátový obsah init packet bol nasledovný:

INITPATH[*FILE_PATH*]

kde za FILE_PATH program doplnil cestu k súboru, ktorá bola zadaná ako parameter programu.

Následne sa po INIT packete posielali dátové packety so zašifrovanými dátami v QNAME.

Po odoslaní všetkých packetov sa vytvoril a odoslal END packet, ktorý signalizoval, že DNS sender ukončil posielanie dát. Tento packet je veľmi dôležitý hlavne pre DNS receiver, pretože po prijatí tohto packetu korektne uzatvorí súbor, do ktorého sa zapisovali dáta. Dáta v END packete majú nasledujúci tvar:

[ENDPACKET]

3. Implementácia

3.1. DNS Sender

1. Prijatie a spracovanie parametrov program
2. Vytvorenie socketu (endpointu) a nastavenie socketu pomocou funkcie setsockopt()
3. Tvorba inicializačného packetu, tvorba a naplnenie DNS header a DNS question
4. Odoslanie inicializačného packetu
5. Čakanie a spracovanie answer packetu s DNS Answer, v prípade chýbajúcej odpovede sa program presunie na bod 3.
6. Príprava DNS Header pre dátový packet

7. Zašifrovanie časti načítaných dát a vloženie do QNAME v správnom formáte + pridanie BASE_HOST
8. Poslanie dátového packet
9. Čakanie a spracovanie answer packet s DNS Answer, v prípade chýbajúcej odpovede sa program presunie na bod 6.
10. Ak DNS sender prijme answer packet, zoberie ďalšie dáta a presunie sa na bod 6., kým nespracuje všetky dáta.
11. Po odoslaní všetkých dátových packetov program začne pripravovať DNS Header pre END packet.
12. Pridanie dát do QNAME.
13. Odoslanie END packet
14. Čakanie a spracovanie answer packetu s DNS Answer, v prípade chýbajúcej odpovede sa program presunie na bod 11.

3.2. DNS Receiver

1. Prijatie a spracovanie parametrov program
2. Vytvorenie socketu (endpointu) a nastavenie socketu pomocou funkcie setsockopt()
3. Prijímanie packetov v infinite while cykle
4. V prípade prijatia INIT packetu sa zistí z dát FILE_PATH, spojí sa s DIR_PATH, vytvorí sa a otvorí daný súbor
5. Odošle sa UDP packet DNS senderovi s DNS answer
6. V prípade dátového packetu sa dáta spracujú, dešifrujú a uložia do súboru
7. Odošle sa UDP packet DNS senderovi s DNS answer
8. V prípade END packet sa uzatvorí korektne otvorený súbor pomocou funkcie fclose()
9. Odošle sa UDP packet DNS senderovi s DNS answer

4. Návod na použitie

Po rozbalení balíčka .tar sa presuňte do hlavného priečinka projektu. Následne otvorte terminál, ktorý sa bude nachádzať v hlavnom priečinku projektu. Príkazom *$make* preložíte projekt. Tento príkaz by Vám mal vygenerovať spustiteľné súbory *dns_sender* a *dns_receiver*. Bližšie informácie o parametroch a spôsoboch spustenia programov *dns_sender* a *dns_receiver* sa dočítate v súbore README.md, ktorý sa nachádza v hlavnom priečinku projektu.

Názorný príklad spustenia:

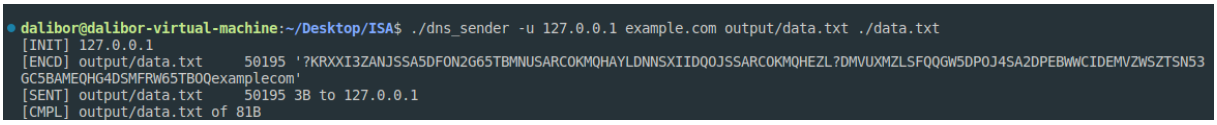
DNS receiver sme spustili príkazom:

$sudo ./dns_receiver example.com ./data

DNS sender sme spustili s príkazom:

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

Po spustení DNS senderu by mal DNS sender poslat UDP packety a jeho výstup by mal vyzerať nasledovne:



Obrázok 3.: Výstup z DNS sendera

DNS receiver by v tomto momente mal dané packety odchytiť a na výstup napísať hlášky o spracovaní, ktoré by mali vyzerať nasledovne:

Obrázok 4.: Výstup z DNS receivra

V tomto momente DNS receiver čaká na prijatie ďalších UDP packetov.

5. Testovanie

Aplikáciu som testoval pomocou softwaru Wireshark, v ktorom som zachytával DNS packety, ktoré poslal môj DNS sender. V týchto packetoch som kontroloval správny formát QNAME sekcie DNS packetu a dáta, ktoré som poslal.

Taktiež som pomocou Wiresharku zachytával aj DNS Answer packety, ktoré poslal DNS receiver, sledoval, či smerujú na správnu IP adresu a či obsahujú tie dáta, ktoré by DNS Answer mala mať.

Bibliografia

[1] paloaltonetworks, „paloaltonetworks,“ [Online]. Available: https://www.paloaltonetworks.com/cyberpedia/what-is-dns-tunneling.

[2] geeksforgeeks.org, „geeksforgeeks.org,“ 2022-05-04. [Online]. Available: https://www.geeksforgeeks.org/udp-server-client-implementation-c/.

[3] A. Mislove, „mislove.org,“ 2011-01-24. [Online]. Available: https://mislove.org/teaching/cs4700/spring11/. [Cit. 2011-01-24].