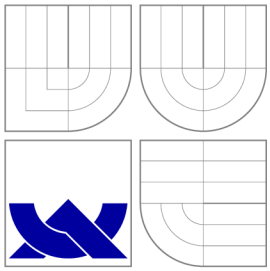
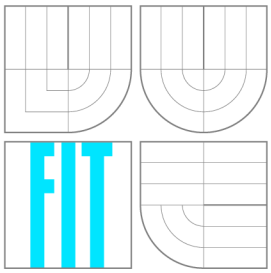


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

DETEKCE TUNELOVÁNÍ DNS NA ZÁKLADĚ ANALÝZY DAT Z APLIKAČNÍ VRSTVY

DNS TUNNELLING DETECTION BASED ON APPLICATION LAYER DATA ANALYSIS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — MICHAL KOUTENSKÝ
AUTHOR

VEDOUCÍ PRÁCE — Ing. MICHAL KOVÁČIK
SUPERVISOR

BRNO 2016

Zadání bakalářské práce/18639/2015/xkoute04

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačových systémů Akademický rok 2015/2016

Zadání bakalářské práce

Řešitel: **Koutenský Michal**
Obor: Informační technologie
Téma: **Detekce tunelování DNS na základě analýzy dat z aplikační vrstvy**
DNS Tunnelling Detection Based on Application Layer Data Analysis
Kategorie: Počítačové sítě

Pokyny:

1. Seznamte se se službou a protokolem DNS, nastudujte problematiku zneužití DNS pro tunelování síťového provozu.
2. Seznamte se s formáty IPFIX, pcap a nástroji pro vytváření DNS tunelů. Analyzujte průběh tunelování na zachycených datech a možnosti jeho detekce.
3. Navrhněte metody detekce tunelování DNS používající data z aplikační vrstvy. Můžete vycházet z již existujících metod.
4. Implementujte navržené metody detekce.
5. Otestujte implementované metody a jejich schopnost detekce tunelování na reálných datech.
6. Zhodnoťte výsledky a navrhněte možnosti rozšíření práce.

Literatura:

- Dle pokynů vedoucího.

Pro udělení zápočtu za první semestr je požadováno:

- Splnění bodů 1 a 2 zadání.

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Kováčik Michal, Ing.**, UPSY FIT VUT
Datum zadání: 1. listopadu 2015
Datum odevzdání: 18. května 2016

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačových systémů a sítí
612 66 Brno, Božetěchova 2

L.S.

doc. Ing. Zdeněk Kotásek, CSc.
vedoucí ústavu

Abstrakt

Cieľom práce je návrh detekčného algoritmu na detekciu DNS tunelovania s použitím dát z aplikačnej vrstvy. Samotnému návrhu prechádza prehľad a analýza dostupných tunelovacích nástrojov a ich spoločných znakov. Špecifická pozornosť je venovaná nástroju *iodine*, s ktorým sú uskutočnené komplexnejšie testy a benchmarky. V závere je implementovaný detekčný algoritmus testovaný na reálnych dátach a sú detailnejšie popísané jeho schopnosti a nedostatky.

Abstract

This bachelor's thesis deals with designing and implementing a detection algorithm for detecting DNS tunnelling using application layer data. The algorithm's design is preceded by overview and analysis of current tunneling tools and their shared characteristics. The tunnelling tool *iodine* is given extra attention and is used to carry out more complex tests and benchmarks. The thesis concludes by testing the implemented algorithm on real data and highlighting its strengths and shortcomings.

Kľúčové slová

DNS tunelovanie, detekcia tunelovania, analýza aplikačnej vrsvy, iodine

Keywords

DNS tunnelling, tunnelling detection, application layer analysis, iodine

Citácia

KOUTENSKÝ, Michal. *Detekce tunelování DNS na základě analýzy dat z aplikační vrstvy.* Brno, 2016. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Kováčik Michal.

Detekce tunelování DNS na základě analýzy dat z aplikační vrstvy

Prehlásenie

Prehlasujem, že som túto bakalálsku prácu vypracoval samostatne pod vedením pána Ing. Michala Kováčika.

.....
Michal Koutenský
17. mája 2016

Poďakovanie

Na tomto mieste by som chcel poďakovať svojmu vedúcemu Ing. Michalovi Kováčikovi za jeho čas, pripomienky a odbornú pomoc na konzultáciach k tejto bakalárskej práci. Ďalej by som rád poďakoval svojim rodičom za podporu pri písaní práce ako aj počas celého štúdia.

Obsah

Kapitola 1

Úvod

V dnešnej dobe je Internet neoddeliteľnou súčasťou života a množstvo ľudí, zariadení a služieb ktoré ho využívajú sa neustále rozrastá. S aktuálnymi trendmi ako *Internet of Things* môžme očakávať že sa toto číslo bude aj naďalej rapídne zvyšovať. Každý z nás má niekoľko zariadení ktoré Internet využívajú, a väčšina týchto zariadení využíva služieb DNS.

DNS je neoddeliteľnou súčasťou moderného Internetu, avšak aj napriek jeho dôležitosti a rozšírenosti sa na prevenciu zneužitia nekladie veľký dôraz. Spôsobov zneužitia je niekoľko, s rôznymi cieľmi — od zamedzenia prístupu k službám, cez podvrhovanie odpovedí a následné presmerovanie komunikácie na kompromitovaný server až po enkapsuláciu iných protokolov v DNS, inak známe aj ako DNS tunelovanie. DNS tak často môže byť slabým článkom inak dobre zabezpečenej firemnej siete a môže byť pomocou tunelovania zneužitý na napr. exfiltráciu dát, alebo naopak infiltráciu tretej osoby do internej siete.

Táto práca sa zaoberá analýzou DNS tunelovania, dostupných nástrojov na tunelovanie, návrhom algoritmu na jeho detekciu a implementáciou navrhnutého algoritmu ako nástroja.

V nasledujúcej kapitole sa budem najprv detailnejšie venovať systému DNS, jeho dôležitosti, využitiu a spôsobu fungovania a neskôr nasleduje popis DNS tunelovania, typických použití a popis bežného princípu na ktorom DNS tunelovanie funguje. V ďalšej kapitole 3 nasleduje popis najbežnejšie používaných formátov na zachytávanie sieťovej komunikácie, konkrétne *pcap* a *NetFlow*, resp. *IPFIX*. Kapitola číslo 4 sa zaoberá prehľadom dostupných nástrojov na tunelovanie, rozborom a analýzou ich komunikácie a na koniec nasleduje benchmark jedného z nástrojov, *iodine*. Kapitola 5 obsahuje návrh algoritmu na základe poznatkov a pozorovaní získaných v predošlých kapitolách. Následujúca kapitola popisuje implementáciu algoritmu ako nástroja. Posledná kapitola obsahuje zhodnotenie implementovaného algoritmu, jeho testovanie na reálnych dátach a možnosti rozšírenia.

Kapitola 2

DNS a DNS tunelovanie

2.1 DNS

Domain Name System [21, 22] je jednou z najvýznamnejších technológií moderného Internetu. Jeho hlavnou úlohou je prekladať ľudsky zapamätateľné doménové mená na IP adresy. Túto funkcionalitu využívajú takmer všetci užívatelia Internetu, zvyčajne však bez ich vedomia. Doménu v zadanej URL *resolver* dotazom na DNS server preloží na IP adresu ktorá je následne použitá pre sieťovú komunikáciu. Tento proces je pre koncového užívateľa neviditeľný. Dotazovať sa DNS serveru na informácie je však možné aj priamo, pomocou utilít ako napr. *nslookup* alebo *dig*.

DNS je systém hierarchický a žiaden server nemá kompletné informácie o všetkých doménach. Dotazy sú preto rekurzívne alebo iteratívne. Pri rekurzívnom dotaze sa dotazovaný server sám dotazuje ďalších serverov na informácie ktoré nevie, pri iteratívnom tieto dotazy vykonáva resolver a server mu poskytne iba adresu nasledujúceho serveru.

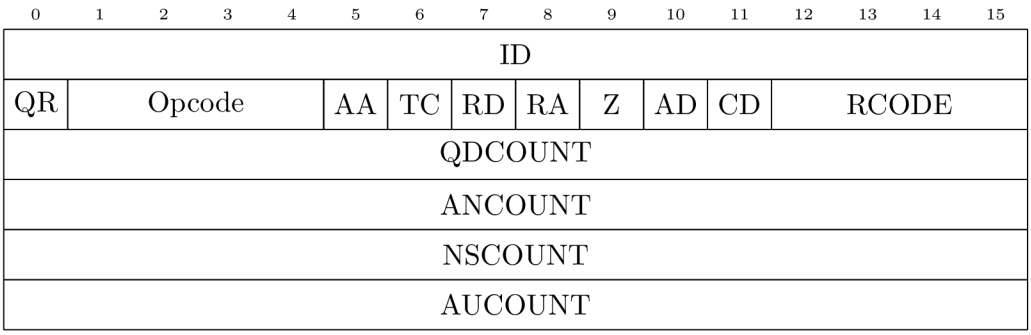
Komunikácia prebieha formou dotaz — odpoveď, kedy klient zašle serveru niekoľko dotazov a server mu na ne odpovie dotazovanými informáciami, resp. odpovie že nevie odpovedať. Komunikácia prebieha zvyčajne na porte 53 pomocou protokolu UDP, avšak v niektorých situáciách, ako je napr. zónový prenos sa využíva aj protokol TCP [20].

Pre každú doménu existuje aspoň jeden autoritatívny server ktorý má korektné informácie o danej doméne. Existujú však aj *caching* servery, ktoré si na určitú dobu uchovávajú informácie na ktoré boli už dotázané pre zvýšenie rýchlosti odpovede, tieto informácie však nemusia byť vždy aktuálne.

Autoritatívny server môže byť buď primárny alebo sekundárny. Primárny uchováva originálne záznamy o zóne, zatiaľ čo sekundárny uchováva identickú kópiu získanú synchronizáciou s primárnym pomocou zónového prenosu.

DNS hlavička pozostáva z nasledujúcich častí [14]:

- *ID* — identifikátor pre daný pár dotaz – odpoveď
- *QR* — bit udávajúci či je daný paket dotaz alebo odpoveď
- *Opcode* — typ dotazu
- *AA* — odpoveď pochádza od autoritatívneho serveru
- *TC* — správa bola skrátená kvôli príliš veľkej dĺžke
- *RD* — žiadosť o rekurzívne spracovanie dotazu



Obr. 2.1: DNS hlavička

- *RA* — server podporuje rekurzívne spracovanie dotazu
- *Z* — rezervované, musí byť 0
- *AD* — všetky dáta v odpovedi boli autentifikované serverom
- *CD* — požiadavok na neoverovanie autentifikácie odpovedí serverom
- *QDCOUNT* — počet dotazov
- *ANCOUNT* — počet odpovedí
- *NSCOUNT* — počet záznamov odkazujúcich na autoritatívny server
- *AUCOUNT* — počet ďalších záznamov

Okrem prekladu doménových mien na adresy poskytuje DNS aj iné informácie, ako napr. adresy mailových serverov pre danú doménu.

Typ záznamu	Použitie
A	Mapovanie doménového mena na IPv4 adresu
AAAA	Mapovanie doménového mena na IPv6 adresu
CNAME	Alias pre doménové meno
MX	Adresa mailového serveru
NS	Adresa DNS serveru
PTR	Ukazateľ na kanonické meno
SOA	Udáva autoritatívne údaje o zóne
SRV	Adresa serveru poskytujúceho nejakú službu
TXT	Text s ľubovolnými informáciami

Obr. 2.2: Najpoužívanejšie typy DNS záznamov

Hierarchická štruktúra znamená, že autoritatívny server pre danú doménu pozná adresy serverov pre jeho subdomény. Na vrchole hierarchie sú tzv. koreňové servery [13], ktoré majú informácie o serveroch pre top-level domény. Adries koreňových serverov je 13, pomenované abecedne A – M. Fyzických serverov je však výrazne viac, a to 504, rozložených vhodne po celom svete. 10 z 13 adries má servery na viacerých kontinentoch.

Dostupnosť koreňových serverov je extrémne dôležitá pre správne fungovanie Internetu, keďže každá rezolúcia adresy začína u nich. Útoky na tieto servery boli zaznamenané v rokoch 2002 [25], 2007 [1] a 2015 [3], vďaka distribuovanosti serverov sa však útočníkom nepodarilo výrazne ovplyvniť dostupnosť DNS služieb.

Hierarchická štruktúra sa využíva aj na overenie autenticity záznamov a zabráneniu podvrhnutiu falošných záznamov útočníkom. Autoritatívny server podpíše svoje záznamy použitím asymetrického šifrovania a svoj verejný kľúč uverejní. Server v hierarchii nad ním podpíše jeho verejný kľúč svojim kľúčom. Týmto sa vytvorí reťaz dôvery ktorá vedie od daného serveru až ku koreňovému.

2.2 DNS tunelovanie

DNS má obrovský význam pre správnu funkčnosť Internetu a internetových služieb a DNS komunikácia prebieha takmer v každej sieti. Aj napriek tomuto rozšíreniu sa však na bezpečnosť DNS zvyčajne nekladie veľký dôraz. DNS je teda možné veľmi jednoducho zneužiť na nežiadúce účely. Jedným z týchto spôsobov je DNS tunelovanie.

Tunelovanie všeobecne je prenos dát nepodporovaného protokolu pomocou iného, podporovaného protokolu. V prípade DNS tunelovania ide zvyčajne o bežný IP prenos, ktorý by bol inak blokovaný. Toto umožňuje útočníkovi napr. využívať captive portal Wi-Fi siete bez potreby poznať prihlasovacie údaje alebo pristupovať na Internet spôsobom ktorý je v rozpore s firemnou politikou. DNS tunelovanie môže byť taktiež využívané malwarom, ako napr. *Morto* alebo *Feederbot* [15, 19].

Tunelované dáta sú zakódované do doménového mena a sú zasielané ako bežný dotaz/odpoveď. Spôsob kódovania ani použitý typ záznamu nie sú pevne nie sú pevne dané, každý nástroj využíva kombináciu jemu vhodnú.

Keďže maximálna povolená dĺžka domény je 255 znakov [21], prenosová rýchlosť DNS tunelovania je veľmi malá. Niektoré nástroje sa snažia tento limit obísť používaním experimentálnych typov záznamov (napr. NULL) a rôznych rozšírení ako EDNS [24]. Ďalšou možnosťou na zvýšenie rýchlosti je využitie kompresie na zmenšenie objemu prenášaných dát.

Nástrojov na tunelovanie pomocou DNS je viacero, napísaných v rôznych jazykoch od C až po jazyky ako Python a Perl. Nástroje sú často multiplatformové a schopné bežať aj na architektúrach ako ARM, čoho dôsledkom je ich dostupnosť napr. aj pre mobilné zariadenia so systémom Android.

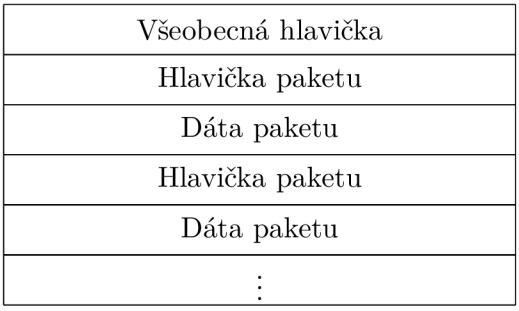
DNS tunelovanie je možné využívať podobne ako VPN, existujú dokonca komerčné služby ktoré túto funkcionalitu poskytujú, často pod názvom VPN-over-DNS. Oproti tradičnej VPN je očividnou nevýhodou rýchlosť, na bežné prehliadanie Internetu je však zvyčajne postačujúca.

Kapitola 3

Formáty

3.1 pcap

Pcap [2] je formát knižnice *libpcap* používanej rôznymi aplikáciami na zachytávanie sieťovej komunikácie ako napr. *tcpdump* alebo *Wireshark*. Štruktúra formátu je veľmi jednoduchá — za hlavičkou obsahujúcou informácie ako je verzia formátu, maximálna dĺžka uloženého paketu a technológiu použitú na linkovej vrstve nasledujú dvojice hlavičky paketu obsahujúcej časové údaje o príchode paketu, uloženú a pôvodnú dĺžku a za ňou samotné dáta paketu.



Obr. 3.1: Formát *pcap*

Pakety uložené do formátu *pcap* obsahujú údaje o všetkých vrstvách, od linkovej po aplikačnú a formát je preto vhodný na ukladanie dát pre neskoršiu analýzu komunikácie, či už na IP vrstve (kto s kým komunikoval), alebo na vrstvách vyšších, kde je možnosť nahliadnuť na konkrétne dáta prenášané v aplikačnej vrstve. Pomocou formátu *pcap* je možné analyzovať spôsob komunikácie konkrétnych aplikačných protokolov (rôzne handshaky, využitie konkrétnych správ daného protokolu...) ako aj použité kódovanie a iné charakteristiky protokolu. Táto hĺbka dostupných informácií znamená že *pcap* nachádza široké využitie v praktických situáciách — debugovanie sieťových problémov, analýza charakteristík sieťovej komunikácie malwaru alebo aj ako pomôcka na výukové účely.

Aplikácie využívajúce knižnicu *libpcap* zachytávajú všetky prichádzajúce aj odchádzajúce pakety na rozhraní, často s využitím promiskuitného módu karty. Keďže pristupujú k rozhraniu priamo, zvyčajne potrebujú administrátorské práva.

3.2 NetFlow

NetFlow [16] je proprietárny formát vytvorený spoločnosťou Cisco na zachytávanie sieťových tokov. Tok je definovaný ako sekvencia paketov s rovnakými hodnotami nasledujúcich 7 parametrov:

1. Ingress interface
2. Zdrojová IP adresa
3. Cieľová IP adresa
4. IP protokol
5. Zdrojový TCP/UDP port
6. Cieľový TCP/UDP port, typ a kód ICMP
7. IP Type of service

NetFlow obsahuje okrem už spomenutých informácií o toku aj ďalšie informácie ako je čas alebo počet bytov a paketov. Do toku sa nemusí zarátavať každý paket, keďže je to výpočetne náročné, ale môže sa brať ako vzorka každý n-tý paket, kde n môže byť konštanta alebo náhodné číslo.

NetFlow využíva tzv. *sondy* ktoré zbierajú dáta, zvyčajne na aktívnych prvkoch v sieti ako sú routre. Tieto dáta sú zasielané *kolektoru* ktorý ich zhromažďuje a uschováva. Výhodou protokolu *NetFlow* je výrazne menšia veľkosť dát, čo umožňuje uschovávať *NetFlow* dáta výrazne dlhšie a za dlhšie časové obdobie.

NetFlow je vhodný na analýzu celkovej komunikácie a tvorbu štatistík, napr. pri správe sietí — prehľad vyťaženia siete počas dňa, objem prenesených dát a podiel aplikačných protokolov na celkovej komunikácii. Absencia konkrétnych dát však neumožňuje detailnú analýzu aplikačných protokolov ako tomu je u formátu *pcap*.

Aj napriek tomu že protokol *NetFlow* je proprietárny, na svojich zariadeniach ho podporujú aj iný výrobcovia sieťových zariadení ako Cisco. *NetFlow* je možné využívať aj na koncových staniciach s populárnymi operačnými systémami podobným Unixu ako Linux a BSD varianty, pomocou aplikácií ako *ipt-netflow* a *softflowd*.

3.3 IPFIX

IPFIX [18, 17] je štandard skupiny IETF slúžiaci ako univerzálny formát pre zber tokových dát na sieti. Základom formátu *IPFIX* je proprietárny formát *NetFlow* spoločnosti Cisco, konkrétne verzia 9.

IPFIX umožňuje rozšíriť zaznamenávané položky o tokové dáta z aplikačnej vrstvy alebo iné informácie zaznamenávané tradične protokolmi *SNMP* a *syslog*. Toto umožňuje vyššiu flexibilitu a širšie možnosti využitia zbieraných dát na analýzu. *IPFIX* obsahuje aj ďalšie vylepšenia oproti proprietárnemu *NetFlow*, ako napr. možnosť položiek s premennou šírkou.

Aj keď je formát *IPFIX* relatívne nový, na svojich zariadeniach ho podporuje množstvo výrobcov, vrátane Cisca. Aplikácie pre koncové stanice ako *ipt-netflow* taktiež podporujú tento nový formát.

Kapitola 4

Analýza dostupných tunelovacích nástrojov

4.1 DeNiSe

DeNiSe [6] je skupina Python skriptov na tunelovanie TCP cez DNS vytvorených Maximillianom Dornseifom. Na prenos dát využíva TXT záznamy a base64 kódovanie.

4.2 dns2tcp

dns2tcp [9] je nástroj na DNS tunelovanie napísaný v jazyku C. Autormi sú Olivier Dembour a Nicolas Collignon. Nástroj využíva CNAME, KEY a TXT záznamy a dáta kóduje base64 kódovaním.

```
0000   3f 7a 45 4f 43 45 52 30 72 6b 41 41 41 41 77 49  ?zEOCER0rkAAAAwI
0010   37 63 66 6d 54 53 54 34 2f 48 6f 41 66 75 37 31  7cfmTST4/HoAfu71
0020   42 71 6d 51 4d 62 56 5a 79 42 6b 58 75 41 6d 6c  BqmQMbVZyBkXuAml
0030   57 66 52 44 6a 43 67 53 51 77 62 63 79 79 48 6d  WfRDjCgSQwbcyyHm
0040   3f 64 37 50 51 73 47 63 67 49 58 45 4a 36 5a 62  ?d7PQsGcgIXEJ6Zb
0050   36 65 76 43 73 6a 59 47 58 64 45 41 41 41 41 67  6evCsjYGXdEAAAAg
0060   45 55 45 4f 45 67 6e 63 6e 66 4b 79 74 57 75 30  EUEOEgncnfKytWu0
0070   65 63 51 7a 54 71 4e 4e 41 53 4e 47 39 4f 5a 59  ecQzTqNNASNG9OZY
0080   18 4d 42 6d 4c 31 51 72 48 66 45 59 44 34 73 75  .MBmL1QrHfEYD4su
0090   6f 62 31 57 4b 4a 77 3d 3d 01 32 01 62 01 33 05  ob1WKJw==.2.b.3.
00a0   31 32 35 38 30 03 64 6e 73 06 74 75 6e 6e 65 6c  12580.dns.tunnel
00b0   00                                               .
```

Obr. 4.1: Dáta prenášané nástrojom *dns2tcp*

4.3 DNScapy

DNScapy [7] je nástroj napísaný v Pythone na vytvorenie SSH tunelu cez DNS. Autormi sú Pierre Bienaimé a Pascal Mazon. Využíva CNAME a TXT záznamy a base64 kódovanie.

```
0000   3f 41 41 41 41 67 46 35 45 4b 2f 69 30 75 4c 2f  ?AAAAgF5EK/i0uL/
0010   7a 67 36 55 68 7a 49 59 7a 68 7a 6e 42 66 32 67  zg6UhzIYzhznBf2g
0020   50 4f 5a 36 76 7a 34 62 45 5a 64 56 7a 4f 55 63  POZ6vz4bEZdVzOUc
0030   55 5a 79 77 37 4b 68 76 42 34 5a 6a 73 4f 42 6e  UZyw7KhvB4ZjsOBn
0040   3f 44 51 61 70 75 68 70 32 6f 43 38 55 32 39 50  ?DQapuhp2oC8U29P
0050   54 56 50 44 53 59 35 76 7a 4d 53 56 59 65 57 76  TVPDSY5vzMSVYeWv
0060   6f 41 71 38 64 74 42 2f 76 4d 5a 53 7a 7a 7a 58  oAq8dtB/vMZSzzzX
0070   51 71 47 57 56 33 41 53 53 72 43 4c 68 75 30 54  QqGWV3ASSrCLhu0T
0080   3e 48 53 7a 5a 32 4a 77 44 5a 50 46 66 4a 4c 7a  >HSzZ2JwDZPFfJLz
0090   62 4b 41 44 4f 59 6d 5a 30 50 48 4b 50 32 6b 55  bKADOYmZ0PHKP2kU
00a0   44 46 63 31 72 4f 6a 77 38 6d 57 73 34 62 4c 71  DFc1rOjw8mWs4bLq
00b0   44 57 38 35 4c 52 33 53 52 76 34 59 54 38 3d 01  DW85LR3SRv4YT8=.
00c0   30 01 62 01 31 05 31 32 33 39 38 03 64 6e 73 06  0.b.1.12398.dns.
00d0   74 75 6e 6e 65 6c 00                             tunnel.
```

Obr. 4.2: Dáta prenášané nástrojom *DNScapy*

4.4 DNScat

DNScat [4] je nástroj napísaný v jazyku Java Tadeuszom Pietraszekom. Využíva A a CNAME záznamy s base64 kódovaním.

4.5 dnscat2

Dnscat2 [5] je nástroj na vytvorenie šifrovaného command-and-control kanálu pomocou DNS protokolu. Klient je napísaný v jazyku C a server v jazyku Ruby. Hlavným autorom je Ron Bowes. *dnscat2* je nástupcom nástroja *dnscat* (taktiež od Rona Bowesa), ktorý bol reimplementáciou *DNScat* od Tadeusza Pietraszeka v jazyku C. Nástroj využíva TXT, MX, CNAME, A a AAAA záznamy a hexadecimálne kódovanie.

```
0000   3c 39 39 61 31 30 33 65 66 62 61 30 30 30 30 30  <99a103efba00000
0010   30 30 30 63 64 63 35 64 63 38 32 65 64 64 33 66  000cdc5dc82edd3f
0020   62 34 35 36 39 65 33 36 35 35 34 36 39 64 32 34  b4569e3655469d24
0030   38 37 66 66 32 62 61 36 35 38 66 34 32 3c 35 37  87ff2ba658f42<57
0040   65 33 64 32 36 61 35 66 36 30 33 65 37 61 34 32  e3d26a5f603e7a42
0050   38 62 61 39 63 33 65 39 66 35 30 64 36 64 33 34  8ba9c3e9f50d6d34
0060   34 30 37 63 39 61 36 35 38 33 64 63 35 36 62 65  407c9a6583dc56be
0070   31 32 64 39 34 31 39 66 36 35 1a 61 62 66 34 37  12d9419f65.abf47
0080   33 63 61 63 33 35 61 35 63 66 35 34 38 35 39 32  3cac35a5cf548592
0090   63 35 30 38 63 02 74 31 03 64 6e 73 06 74 75 6e  c508c.t1.dns.tun
00a0   6e 65 6c 00                                      nel.
```

Obr. 4.3: Dáta prenášané nástrojom *Dnscat2*

4.6 Heyoka

Heyoka [8] je nástroj na DNS tunelovanie s cieľom predísť detekcii a dosiahnuť vyššej prenosovej rýchlosti ako iné dostupné nástroje. Nástroj je napísaný v jazyku C a autormi sú Alberto Revelli a Nico Leidecker. Využíva binárne kódovanie ako aj base32/64 a TXT a NULL záznamy. Umožňuje taktiež rozložiť komunikáciu medzi viacero serverov a podvrhovanie IP a MAC adries iných zariadení v lokálnej sieti pre rozloženie DNS komunikácie rovnomerne po celej sieti.

```
0000   3f 6c 01 13 00 87 00 16 03 01 03 30 02 00 00 46  ?l.........0...F
0010   03 01 56 cd ed d3 a4 a8 20 d8 0b 7f 3a 7b db 2b  ..V..... ...:{.+
0020   5e 12 cf 0b df f8 8d a7 9d 32 a2 25 a4 ab 1c 70  ^........2.%...p
0030   c4 c4 20 c4 2b 00 00 d3 7b a6 e5 30 4f 9c e1 e6  .. .+...{..0O...
0040   3f f2 89 44 dc 7b d7 95 52 fb 98 a9 3e 4a ca d3  ?..D.{..R...>J..
0050   38 c5 f6 a3 00 2f 00 0b 00 02 de 00 02 db 00 02  8..../..........
0060   d8 30 82 02 d4 30 82 01 bc a0 03 02 01 02 02 10  .0...0..........
0070   4d 0b 24 2c 40 02 6d 8a 41 19 3d 78 77 76 3a ea  M.$,@.m.A.=xwv:.
0080   3f 30 0d 06 09 2a 86 48 86 f7 0d 01 01 05 05 00  ?0...*.H........
0090   30 13 31 11 30 0f 06 03 55 04 03 13 08 77 6f 6a  0.1.0...U....woj
00a0   61 6b 2d 50 43 30 1e 17 0d 31 36 30 32 32 33 31  ak-PC0...1602231
00b0   37 34 30 35 39 5a 17 0d 31 36 30 38 32 34 31 37  74059Z..16082417
00c0   2f 34 30 35 39 5a 30 13 31 11 30 0f 06 03 55 04  /4059Z0.1.0...U.
00d0   03 13 08 77 6f 6a 61 6b 2d 50 43 30 82 01 22 30  ...wojak-PC0.."0
00e0   0d 06 09 2a 86 48 86 f7 0d 01 01 01 05 00 03 82  ...*.H..........
00f0   02 74 32 03 64 6e 73 06 74 75 6e 6e 65 6c 00     .t2.dns.tunnel.
```

Obr. 4.4: Dáta prenášané nástrojom *Heyoka*

4.7 iodine

Iodine [10] je multiplatformový nástroj napísaný v jazyku C. Autormi sú Erik Ekman, Bjorn Andersson a Anne Bezemer. Nástroj využíva kódovania base32, base64 a base128 ako aj binárne kódovanie — podporované kódovanie je autodetekované na začiatku komunikácie a záleží aj na použitom type záznamu. Používané typy záznamov sú NULL, PRIVATE, TXT, SRV, MX, CNAME a A — vhodný typ záznamu je taktiež autodetekovaný. Prenášané dáta sú komprimované do *gzip* formátu.

```
0000   3e 31 6d 65 62 77 38 32 ca 32 68 62 be ee 6b d6  >1mebw82.2hb..k.
0010   67 67 cd bf ec de 34 79 6a 33 73 de c5 be de 64  gg....4yj3s....d
0020   79 c6 df ca 45 57 c2 4c 54 fd cf f9 73 c4 43 45  y...EW.LT...s.CE
0030   54 34 79 64 68 c7 f8 cc bf be e2 59 79 43 55 11  T4ydh......YyCU.
0040   64 45 4e c6 43 57 75 df 39 be 57 64 be d6 6f 52  dEN.CWu.9.Wd..oR
0050   be 03 64 6e 73 06 74 75 6e 6e 65 6c 00           ..dns.tunnel.
```

Obr. 4.5: Dáta prenášané nástrojom *iodine*

4.8 OzymanDNS

OzymanDNS [11] je skupina Perl skriptov napísaná Danom Kaminskym na vytvorenie SSH cez DNS tunelu. Využíva base32 a base64 kódovanie a TXT záznamy.

4.9 tcp-over-dns

tcp-over-dns [12] je nástroj napísaný v jazyku Java Timom Valenzuelom. Využíva base63, base16 a hexhack37 kódovania a CNAME a TXT záznamy. Dáta sú komprimované algoritmom *LZMA*.

```
0000   3f 62 6a 69 4b 59 68 77 43 69 70 35 55 31 6d 75  ?bjiKYhwCip5U1mu
0010   6e 67 35 53 77 72 79 69 4d 43 32 62 56 58 6d 42  ng5SwryiMC2bVXmB
0020   48 68 68 48 43 6f 46 49 54 68 6c 38 72 76 74 74  HhhHCoFIThl8rvtt
0030   31 4c 4c 6e 7a 62 36 7a 68 62 68 31 30 6a 61 38  1LLnzb6zhbh10ja8
0040   3f 35 6d 4e 45 69 52 79 66 61 61 61 61 64 69 56  ?5mNEiRyfaaaadiV
0050   66 77 4c 61 32 66 31 6b 54 74 56 6c 34 38 61 44  fwLa2f1kTtVl48aD
0060   71 74 79 64 43 63 46 35 52 54 6c 49 64 69 44 66  qtydCcF5RTlIdiDf
0070   6a 78 6f 6e 4b 56 5a 36 32 4e 71 72 76 36 42 76  jxonKVZ62Nqrv6Bv
0080   3f 4d 69 6d 39 48 58 31 54 67 31 54 35 4f 54 71  ?Mim9HX1Tg1T5OTq
0090   6e 61 6f 52 7a 6c 73 67 47 70 62 50 56 72 61 61  naoRzlsgGpbPVraa
00a0   61 61 62 64 51 6a 75 45 78 61 41 55 63 57 77 4d  aabdQjuExaAUcWwM
00b0   39 57 30 66 75 36 69 6b 37 34 61 53 52 33 38 47  9W0fu6ik74aSR38G
00c0   08 67 68 50 52 63 4d 6f 48 02 74 31 03 64 6e 73  .ghPRcMoH.t1.dns
00d0   06 74 75 6e 6e 65 6c 00                          .tunnel.
```

Obr. 4.6: Dáta prenášané nástrojom *tcp-over-dns*

4.10 Rozbor paketov a komunikácie

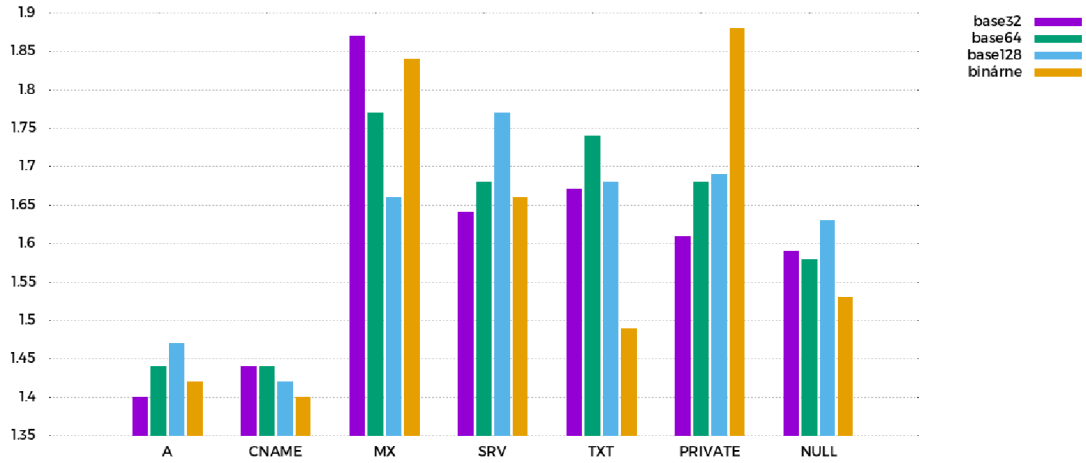
Dĺžka doménového mena je obmedzená na 255 znakov, väčšina nástrojov sa snaží tento priestor využiť čo najviac, z čoho môžme pokladať dotazy na extrémne dlhé doménové mená za podozrivé. Ďalším podozrivým faktorom je samotný obsah — dáta sú kódované, zvyčajne kódovaním base64, ktorého výstupom sú náhodne vyzerajúce zhluky znakov. Môžme teda hľadať skupiny znakov ktoré sa v bežných doménových menách nevyskytujú, ako napr. vysoký počet spoluhlások alebo samohlások za sebou. Nezobraziteľné znaky využívané binárnym kódovaním u nástrojov ako *iodine* sú taktiež veľmi podozrivé.

Typy záznamov používané nástrojmi často patria medzi tie menej používané až experimentálne — vysoký počet napr. NULL záznamov pre tú istú doménu znamená vysokú pravdepodobnosť že sa jedná o DNS tunelovanie a nie bežný DNS prenos. Všeobecne vysoký počet dotazov na jednu a tú istú doménu sa dá považovať za anomáliu a nie súčasť bežnej DNS komunikácie a môže indikovať existenciu DNS tunelu.

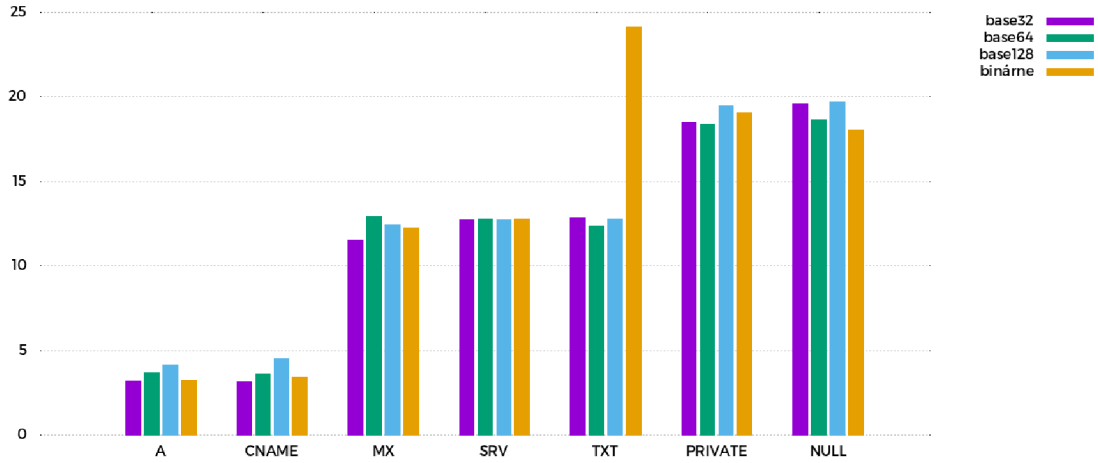
Na základe týchto pozorovaní môžme vytvoriť zoznam pravidiel ktoré použijeme pri detekcii.

4.10.1 Benchmark a analýza nástroja iodine

Nástroj *iodine* bol zvolený pre komplexnejšiu analýzu z dôvodu že zo skúmaných nástrojov poskytuje najväčšiu konfigurovateľnosť čo sa týka parametrov tunelovania. Väčšina skúmaných nástrojov sa dá principiálne považovat za podmnožinu nástroja *iodine* čo sa týka funkcionality. Výnimkami sú nástroje *dnscat2*, ktorý používa hexadecimálne kódovanie, a *Heyoka*, ktorý poskytuje pokročilejšiu funkcionalitu na zabránenie detekcii v porovnaní s ostatnými nástrojmi, ktoré túto problematiku prevažne neriešia.

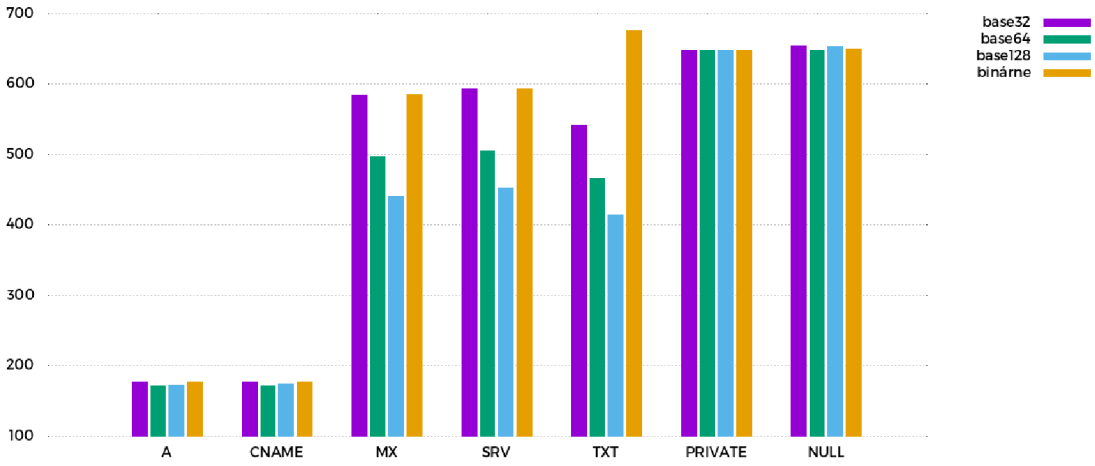


Obr. 4.7: Primerná veľkosť zhluku samohlások pri jednolivých konfiguráciach nástroja



Obr. 4.8: Primerné percento nezobraziteľných znakov pri jednolivých konfiguráciach nástroja

Z nameraných dát si môžme všimnúť, že hodnoty zhlukov spoluhlások sú si veľmi podobné (testovacia doména obsahovala v názve *dns*, čo zdvihlo minimum u všetkých prípadov na hodnotu 3, tento fakt treba brať do úvahy), zhluky spoluhlások sa preto nejavia ako vhodné na detekciu. Zvyšné tri merané informácie sú oveľa zaujímavejšie. Priemerná dĺžka zhluku samohlások u nástroja *iodine* sa pohybuje medzi 1.4 – 1.88, u bežných DNS dotazov je nameraná hodnota 0.47, čo by aj v ideálnom prípade kedy každý zhluk má minimálnu



Obr. 4.9: Primerná veľkosť dotazov pri jednolivých konfiguráciach nástroja

možnú dĺžku 2 znamenalo že ¾ doménových mien zhluk neobsahujú. Hľadanie zhlukov s dĺžkou 2 a viac bude preto pri detekcii veľmi užitočné. Bežné DNS dotazy neobsahujú nezobraziteľné znaky, pakety nástroja *iodine* áno, v rôznom počte. Pri testovaní bolo použité upstream kódovanie base128, ďalšie možnosti sú base64 a base32 — toto nastavenie je autodetekované. Pri použití kódovaní base64 a base32 by tieto hodnoty boli taktiež nulové. Veľkosť dát je pri použití záznamov A a CNAME podobná ako u bežných dotazov, pri ostatných typoch je v priemere 2 až 3-krát väčšia.

Kapitola 5

Návrh detekčného algoritmu

Cieľom prace je vytvorenie nástroja na detekciu DNS tunelovania na základe analýzy dát z aplikačnej vrstvy. Vstupom budú teda dáta obsahujúce zachytenú DNS komunikáciu obsahujúce aspoň názov dotazovanej domény. Okrem dát z aplikačnej vrstvy sú na detekciu potrebné aj metadáta ako veľkosť aplikačných dát a čas kedy bol dotaz zachytený. Algoritmus teda pracuje so sadou časovo usporiadaných dotazov, resp. odpovedí na ne.

V predchádzajúcej kapitole sme sa zamerali na pozorovanie a analýzu tunelovanej komunikácie, jej spoločné znaky a rozdiely oproti bežnej komunikácii. Asi najvýznamnejším znakom je neprirodzene veľký objem prenášaných dát. Dotazy v bežnej komuniácii sú zvyčajne malé, pretože dlhé doménové mená sú pre užívateľov ťažko zapamätateľné. Výnimkou sú domény využívané rôznymi službami alebo skriptmi ku ktorým užívateľ nepristupuje priamo a preto môžu byť dlhé, často obsahujú aj časť ktorá sa užívateľovi javí ako náhodný text. Komunikácia DNS tunelov sa naopak snaží možností DNS protokolu využiť čo najviac aby bola schopná preniesť čo najväčší objem dát v čo najmenšom čase. Dáta nie sú prenášané len zakódovaním do domény ale často sú využívané aj ďalšie polia pre konkrétny typ záznamu, ktoré za určitých podmienok umožňujú preniesť niekoľkonásobne viac dát ako by tomu bolo možné pri použití len doménových mien. Z tohto dôvodu nie je postačujúce sledovať dĺžku doménového mena ale je potrebné sa zamerať na celkovú veľkosť dát aplikačnej vrstvy.

Okrem veľkosti dát sú ďalším spoločným znakom náhodne vyzerajúce doménové mená. Tieto doménové mená náhodné nie sú, keďže obsahujú prenášané dáta zakódované vhodným kódovaním (zvyčajne jedným z base* kódovaní), z pohľadu užívateľa a hlavne z pohľadu prirodzeného jazyka sa však takto tvária. Tohoto faktu je možné využiť a zamerať sa na kombinácie znakov ktoré sa v prirodzenom jazyku (a teda aj doménových menách) bežne nevyskytujú. Príkladom môžu byť zhluky samohlások, spoluhlások alebo v prípade binárneho kódovania nezobraziteľných znakov.

Ďalšou z možností ako tento fakt využiť je napr. frekvenčná analýza znakov v doménovom mene. V dnešnej dobe globalizácie však užívatelia pristupujú úplne bežne k doménam vo viacerých a odlišných jazykoch ako je ich rodný jazyk a tieto jazyky môžu mať odlišné frekvenčné charakteristiky. Preto by bolo pri tomto prístupe vhodné, možno až potrebné, vedieť slová akého jazyka daná doména obsahuje. Toto sa dá čiastočne odvodiť z národných domén najvyššej úrovne, tento spôsob však nie je úplne spoľahlivý a u (neustále sa zvyšujúceho[1]) množstva generických top-level domén prakticky nemožný. Z týchto dôvodov sa frekvenčnej analýze pri detekcii venovať nebudeme.

[1] https://newgtlds.icann.org/en/program-status/delegated-strings

Nástroje na DNS tunelovanie využívajú rozsiahle množstvo dostupných typov záznamov, množstvo ktorých sa v bežnej komunikácii vyskytuje veľmi zriedkavo. Hľadanie týchto vzácnejších typov záznamov je tiež jeden z možných spôsobov ako sa snažiť detekovať tunel. Rada nástrojov však podporuje aj bežnejšie používané typy záznamov ako CNAME alebo A a preto sa v tejto práci zameriame na znaky ktoré sú spoločné pre čo najväčšiu množinu používaných konfigurácií.

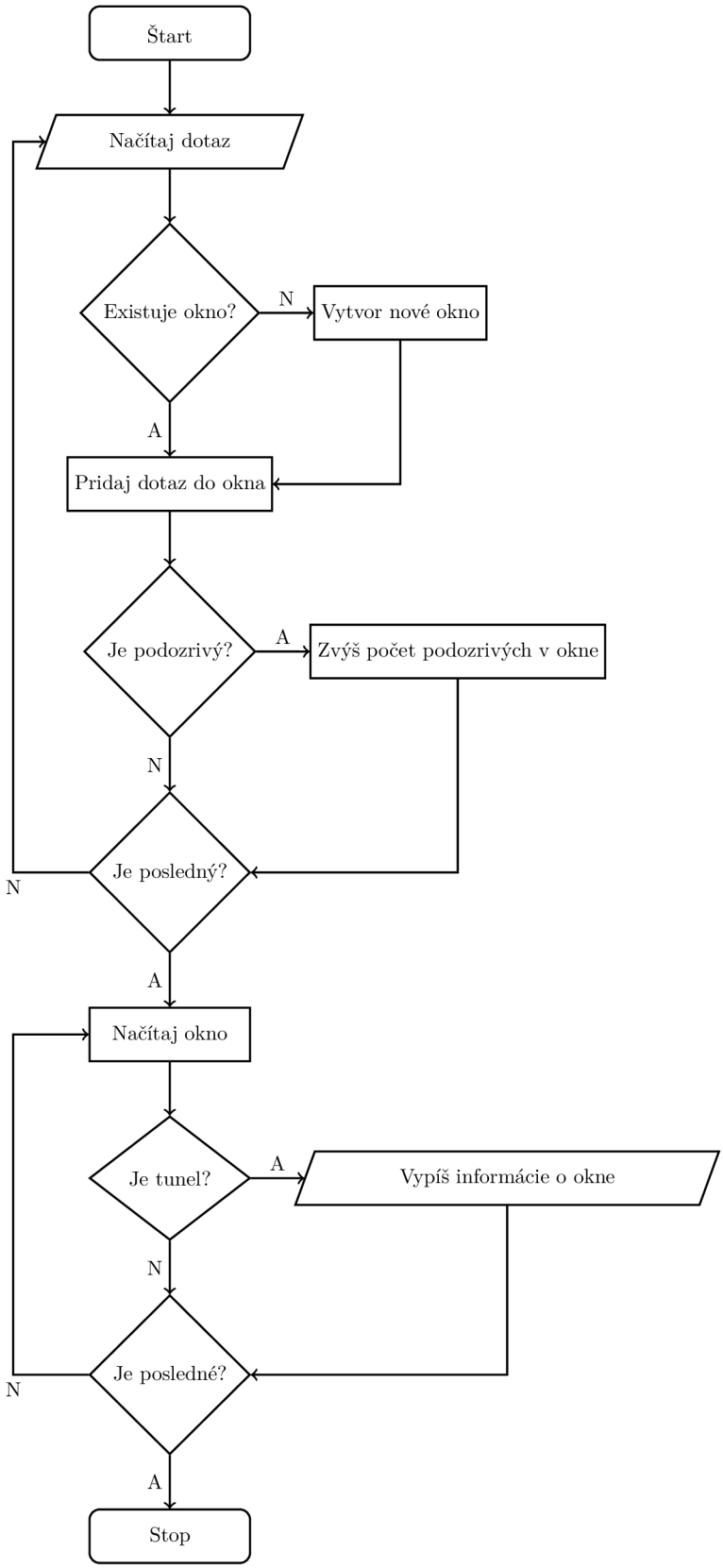
Posledným významným spoločným znakom je frekvencia dotazov na danú doménu. Pri bežnom prehliadaní Internetu sa užívateľ zvyčajne dotáže niekoľkokrát za sekundu, pri tunelovanej komunikácii je toto číslo výrazne vyššie, a to až niekoľkotisíckrát za sekundu.

Na obrázku 5.1 môžeme vidieť základný princíp detekčného algoritmu. Nástroj postupne spracováva dotazy na vstupe a radí ich do časových okien zoskupených podľa "základnej" domény, tj. domény prvého a druhého stupňa. Okno obsahuje doménu do ktorej patrí, počet paketov, počet podozrivých pektov a čas prvého a posledného dotazu v okne. Dotaz patrí do okna pokiaľ je od posledného paketu v okne vzdialený najviac o časový interval t. Následne je kontrolovaný na podozrivé znaky. Podľa zistení popísaných vyššie môžeme zostaviť pravidlá na detekciu. Budeme hľadať dotazy ktoré:

1. obsahujú zhluk 2 a viac samohlások
2. majú veľkosť dát väčšiu ako 400 B
3. obsahujú nezobraziteľné znaky

Zhluky samohlások budeme hľadať len v doménach tretej úrovne a úrovní nižších — týmto zabránime množstvu falošných pozitív pre populárne domény ktoré obsahujú dve samohlásky za sebou v názve, ako napr. *google.com* alebo *facebook.com*. Keďže top-level domény sú spravované rôznymi organizáciami, použitie top-level domény na tunelovanie je veľmi nepravdepodobné a táto zmena nijak neovplyvní efektívnosť detekcie.

Ak dotaz splní aspoň jednu z podmienok, považujeme ho za podozrivý a túto informáciu zapíšeme aj do okna do ktorého patrí. Po spracovaní všetkých dotazov nasleduje vyhodnocovanie okien. Vypočíta sa percento podozrivých paketov v okne – táto hodnota musí byť vyššia ako parameter s. Ten prakticky udáva striktnosť detekcie — nižšie hodnoty detekujú viac tunelov, no zvyšuje sa šanca že sú to len falošné pozitíva. Následne sa ešte kontroluje, či je počet paketov v okne väčší ako dĺžka okna násobená hodnotou parametra p — ten je teda možné vnímať ako minimálny priemerný počet paketov za sekundu v tunelovanej komunikácii. Kontrolovaním množstva komunikácie v okne zabránime falošným pozitívam pre okná ktoré síce spĺňajú podmienku percenta podozrivých paketov, obsahujú však len malé množstvo paketov. Príkladom môže byť okno s jedným podozrivým paketom — percentuálne je množstvo podozrivých 100%, jeden osamelý dotaz však neznamená prítomnosť tunelu. Ak okno spĺňa obidve podmienky, je komunikácia považovaná za tunel a informácie o okne sú vypísané užívateľovi.



Obr. 5.1: Graf detekčného algoritmu

Kapitola 6

Implementácia

Nástroj na detekciu je implementovaný v jazyku C pre operačný systém GNU/Linux. Spôsob použitia je popísaný na obrázku 6.1.

```
Uage:
  detude [-h] [-p <pps>] [-w <seconds>] [-s <percent>] file
Arguments:
  -h: Print this help
  -p: Packets per second (default: 700)
  -w: Window size in seconds (default: 2)
  -s: Necessary percentage of suspicious packets (default: 60)
```

Obr. 6.1: Spôsob spustenia detekčného nástroja

Nástroj je rozdelený na 3 hlavné časti — `detude.c` rieši hlavnú aplikačnú logiku a samotný detekčný algoritmus, `btree.c` obsahuje potrebné dátové štruktúry a operácie nad nimi a `csv.c` sa venuje načítavaniu vstupných dát vo formáte *csv*.

Nástroj pracuje so vstupnými dátami vo formáte *csv*[1]. Formát *csv* bol zvolený pre svoju jednoduchosť, flexibilnosť a malú priestorovú náročnosť v porovnaní s inými dostupnými formátmi ako je napr. *pcap*. Nástroje na zachytávanie sieťovej komunikácie ako napr. populárny Wireshark často poskytujú možnosť exportu zachytených dát do formátu *csv* — je teda možné nepriamo spracovávať aj vstupy poskytnuté v iných formátoch.

Vstupné súbory musia obsahovať na prvom riadku hlavičku popisujúcu obsah jednotlivých stĺpcov. Na korektné spracovanie vstupu musí hlavička obsahovať aspoň nasledujúce tri hodnoty: `uint64 BYTES`, `time TIME_FIRST` a `string DNS_NAME`. Na poradí stĺpcov nezáleží, ako ani na konkrétnej pozícii. Súbor môže obsahovať aj ďalšie stĺpce okrem týchto troch, na spracovávanie vstupných dát nemajú vplyv a ich hodnoty sú ignorované. Pre správne fungovanie algoritmu a triedenie do okien je nevyhnutné, aby vstupné dáta boli zoradené podľa hodnôt v stĺpci `TIME_FIRST` vzostupne. Nástroj si toto zoradenie kontroluje a v prípade že narazí na dotaz ktorého čas je menší ako čas toho čo bol spracovávaný pred ním informuje užívateľa o tomto fakte a svoju činnosť ukončí. Hodnoty v stĺpci `TIME_FIRST` sú očakávané vo formáte `%Y-%m-%dT%T`[2]. Okrem doménových mien nie sú hodnoty ohraničené dvojitými úvodzovkami (").

Nástroj postupne číta dáta zo vstupu a zaraďuje ich do okien do ktorých patria. Okná sú implementované ako jednosmerný zoznam štruktúr zobrazených na obrázku 6.2. Každé okno

[1] Comma Separated Values

[2] `man strptime`

obsahuje čas začiatku a konca okna, počet paketov v okne a počet podozrivých paketov. Dotazy na domény sa zhlukujú do okien na základe času. Dotaz je súčasťou okna ak je od posledného paketu v okne vzdialený najviac o časový interval t, kde t je parameter konfigurovateľný pri spustení. Prednastavená hodnota t sú 2 sekundy. Pokiaľ dotaz dorazil neskôr, stáva sa začiatkom nového okna. Následne sa kontrolujú podozrivé znaky popísané v minulej kapitole — dĺžka, zhluky samohlások a nezobraziteľné znaky. Pokiaľ má dotaz aspoň jeden z týchto znakov, je započítaný ako podozrivý.

```
typedef struct dList {
  time_t first;       // time of first dns request
  time_t last;        // time of last dns request
  struct dList *next; // pointer to next window
  int count;          // number of requests in window
  int susp;           // number of suspicious requests in window
} *dListPtr;
```

Obr. 6.2: Štruktúra pre jednosmerný zoznam okien strom

Každá doména má svoj zoznam okien a tieto zoznamy sú uložené v binárnom strome ktorý používa doménu prvého a druhého stupňa ako kľúč. Štruktúra pre tento strom je popísaná na obrázku 6.3.

```
typedef struct bTree {
  struct bTree *left;  // pointer to the left leaf node
  struct bTree *right; // pointer to the right leaf node
  char *key;           // node key (domain name)
  dListPtr first;      // pointer to the first window (beginning of list)
  dListPtr last;       // pointer to the latest window (end of list)
} *bTreePtr;
```

Obr. 6.3: Štruktúra pre binárny strom

Po spracovaní všetkých paketov sa prechádza stromom po jednotlivých oknách. Kontroluje sa, či je počet paketov väčší ako priemerná hodnota paketov za sekundu p násobená dĺžkou okna. Parameter p je taktiež konfigurovateľný pri spustení, prednastavená hodnota je 700. Týmto sa vyradia falošné pozitíva okien, kde je napr. jeden paket s podozrivým znakom. Následne sa zisťuje percento podozrivých paketov — ak je vyššie ako hodnota s, vypíšu sa informácie o okne spolu s doménou do ktorej okno patrí. Prednastavená hodnota s je 60 a je taktiež konfigurovateľná pri spustení.

Keďže je detekčný algoritmus oddelený od spracovávania vstupu, je veľmi jednoduché pridať podporu pre ďalšie vstupné formáty dát. Vo funkcii `main` sa namiesto funkcie `csv_detection` zavolá funkcia spracúvajúca iný dátový formát, ktorá po načítaní každého dotazu zavolá funkciu `add_packet` s potrebnými parametrami. Táto funkcia následne analyzuje dotaz a uloží ho do správneho okna. Funkcia využíva pomocné funkcie ako `find_basedomain`, ktorá nájde v danom doménovom mene začiatok domény druhej úrovne, alebo `check_name`, ktorá s použitím ďalších funkcií (`is_vowel`, `is_unprintable`) rozhodne, či je dané doménové meno podozrivé. Funkcia ďalej skontroluje veľkosť a čas a dotaz zaradí do príslušného okna. Po úspešnom ukončení funkcie spracovávajúcej vstupné dáta je volaná funkcia `bTreePrint` ktorá prechádza stromom metódou *inorder* ktorá zaručuje že užívateľ dostane podozrivé domény zoradené v abecednom poradí.

Kapitola 7

Zhodnotenie výsledkov

7.1 Testovanie

Testovanie prebiehalo priebežne počas celej doby implementácie a zároveň s testovaním prebiehalo aj ladenie ako detekčného algoritmu tak aj detekčného nástroja. Na testovanie boli použité zachytené dáta nástroja *iodine* ako aj poskytnuté dáta reálnej DNS komunikácie.

Výstup nástroja pre detekovaný tunel je zobrazený na obrázku 7.1. Z poskytnutých informácií môžme vyčítať, že nástroj detekoval tunelovanú komunikáciu ktorá využívala doménu `dns.tunnel`, 100% dotazov v okne nástroj označil za podozrivé, okno obsahuje 584490 dotazov, okno (tj. súvislá komunikácia) trvala 748 sekúnd a priemerný počet paketov za sekundu je 781. Tento tunel bol vytvorený nástrojom *iodine* a využíval NULL záznamy a binárne kódovanie. Táto kombinácia konfiguračných parametrov je z hľadiska detekčného algoritmu veľmi priaznivá a zapríčinila tak pomerne vysokú hodnotu podozrivých dotazov.

```
DOMAIN: dns.tunnel SUSPICIOUS: 100.00% COUNT: 584490 LENGTH: 748s PPS: 781
```

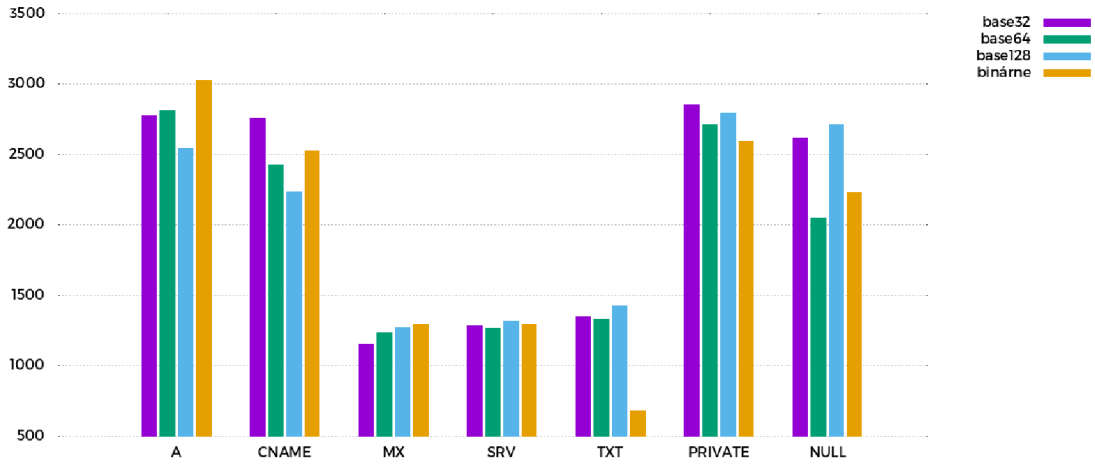
Obr. 7.1: Výpis detekčného nástroja (*iodine*, NULL záznamy, binárne kódovanie)

Na obrázku 7.2 je zobrazený výstup pre tunel nástroja iodine s využitím A záznamov a base32 kódovania. Z vypísaných informácií môžeme vidieť že táto konfigurácia je výrazne menej efektívna čo sa týka prenosovej rýchlosti dát. Dôležitejší je však fakt, že len necelých 64% dotazov bolo označených ako podozrivých. Táto kombinácia konfiguračných parametrov je naopak veľmi nepriaznivá a dá sa považovať za najhoršiu možnú možnosť. Aj napriek tomu bol nástroj schopný tunel detekovať s viac ako polovičnou úspešnosťou. Testovanie na iných dátach s podobnou konfiguráciou dosahovalo podobné výsledky a z tohto dôvodu bola predvolená hodnota hranice podozrivosti stanovená na 60%.

```
DOMAIN: dns.tunnel SUSPICIOUS: 63.98% COUNT: 779794 LENGTH: 1109s PPS: 703
```

Obr. 7.2: Výpis detekčného nástroja (*iodine*, A záznamy, base32 kódovanie)

Obrázok 7.4 obsahuje výstup nástroja po spustení s reálnymi dátami a prednastavenými hodnotami parametrov detekčného algoritmu. V porovnaní s dátami z nástroja *iodine* si môžme všimnúť výrazne kratšie dĺžky okien ako aj niekoľkonásobne vyšší počet dotazov za sekundu. Množstvo dotazov za sekundu je samozrejme výrazne závislé na bode v ktorom sú dáta zachytávané a popularite dotazovanej domény. Je teda na užívateľovi aby túto hodnotu správne nastavil vzhľadom na analyzované dáta. Hodnoty pri benchmarku nástroja



Obr. 7.3: Počet paketov za sekundu pri jednotlivých konfiguráciách nástroja *iodine*

iodine sa pohybovali v rozmedzí 1000 – 3000 dotazov za sekundu, ako môžeme vidieť na grafe 7.3. Tieto hodnoty sú však veľmi závislé na rôznych faktoroch prostredia a pri ďalších zachytených tuneloch občas dosahujú aj hodnoty okolo 500 až 800 dotazov za sekundu, ako môžme vidieť aj na obrázkoch 7.1 a 7.2. Pri zachytávaní komunikácie vo frekventovaných uzloch môže byť frekvencia komunikácie tunelu niekoľkonásobne nižšia ako dotazy na populárne domény, odporúčaná hodnota je preto niekoľko stoviek a slúži naozaj len na odstránenie falošných pozitív obsahujúcich malé množstvo jednotiek až desiatok paketov.

```
DOMAIN: ac.at SUSPICIOUS: 70.67% COUNT: 2216 LENGTH: 1s PPS: 2216
DOMAIN: access-board.gov SUSPICIOUS: 68.22% COUNT: 7178 LENGTH: 1s PPS: 7178
DOMAIN: adform.net SUSPICIOUS: 64.85% COUNT: 2979 LENGTH: 1s PPS: 2979
DOMAIN: alicdn.com SUSPICIOUS: 78.73% COUNT: 832 LENGTH: 1s PPS: 832
DOMAIN: amazonaws.com SUSPICIOUS: 77.00% COUNT: 27203 LENGTH: 1s PPS: 27203
DOMAIN: apple-dns.net SUSPICIOUS: 100.00% COUNT: 3498 LENGTH: 1s PPS: 3498
DOMAIN: avg.com SUSPICIOUS: 96.17% COUNT: 2191 LENGTH: 1s PPS: 2191
DOMAIN: cpsc.gov SUSPICIOUS: 82.14% COUNT: 257755 LENGTH: 31s PPS: 8314
DOMAIN: e5.sk SUSPICIOUS: 94.54% COUNT: 5546 LENGTH: 1s PPS: 5546
DOMAIN: elasticbeanstalk.com SUSPICIOUS: 67.14% COUNT: 767 LENGTH: 1s PPS: 767
DOMAIN: hpecorp.net SUSPICIOUS: 82.40% COUNT: 3449 LENGTH: 1s PPS: 3449
DOMAIN: isc.org SUSPICIOUS: 60.56% COUNT: 890 LENGTH: 1s PPS: 890
DOMAIN: online-metrix.net SUSPICIOUS: 89.58% COUNT: 931 LENGTH: 1s PPS: 931
DOMAIN: reuters.com SUSPICIOUS: 93.17% COUNT: 3573 LENGTH: 4s PPS: 893
DOMAIN: revsci.net SUSPICIOUS: 78.97% COUNT: 894 LENGTH: 1s PPS: 894
```

Obr. 7.4: Výpis detekčného nástroja (reálne dáta)

Prenosy zobrazené na obrázkoch 7.1 a 7.2 zobrazujú prenos 50MB dát a sú preto pomerne dlhé a súvislé — v porovnaní s týmito hodnotami sú okná z reálnych dát relatívne krátke. Pri dnešných rýchlostiach internetových pripojení je však možne preniesť značné množstvo dát za sekundu. Aj jednosekundové okno tak môže znamenať prítomnosť tunelu a preto výpis tejto hodnoty je skôr informatívneho charakteru a slúži ako dodatočný poznatok pre užívateľa. Každé okno sa však spracováva a vyhodnocuje samostatne — označených

môže byť niekoľko okien tej istej domény. Pri reálnej tunelovanej komunikácii sa dá predpokladať, že ak by bola komunikácia rozdelená do viacerých krátkych okien, bolo by detekované viac ako jedno. Tunelovacie nástroje často začínajú handshakom kde sa autodetekujú podporované možnosti komunikácie a volí sa optimálna konfigurácia s ohľadom na rýchlosť. V závislosti na dĺžke analyzovaného časového intervalu môžeme predbežne považovať výskyty jedného jednosekundového okná pre doménu za falošné pozitíva. Zameriame sa preto najprv na okná ktoré majú časovú dĺžku väčšiu ako je 1s.

Nástroj detekoval dve takéto okná — jedno pre doménu `reuters.com` a jedno pre `cpsc.gov`. Doména `reuters.com` patrí známej medzinárodnej spravodajskej agentúre Reuters a pravdepodobnosť že sa jedná o tunel je preto relatívne nízka. Po nahliadnutí do zdrojových dát nachádzame množstvo dotazov na poddomény domény `reutersmedia.reuters.com`. Slovo *reutersmedia* v sebe obsahuje hneď dva zhluky samohlások — *eu* a *ia*. Použitie poddomény *media* a jej variant pre hosťovanie multimediálneho obsahu pre veľké portály je relatívne časté a vytvára falošné pozitíva s ktorými si však detekčný algoritmus nevie poradiť. Niektoré možnosti riešenia tohto problému sú načrtnuté v sekcii 7.2.

Ďalšou detekovanou doménou je doména `cpsc.gov`. Tentokrát sa jedná o doménu vlády Spojených štátov amerických a pravdepodobnosť že ide o tunel je preto zase pomerne nízka. Nahliadnutie do dát ukazuje dotazy o veľkosti od 447 B (prvá hodnota nad detekčný limit 400 B) do neuveriteľných 66940047 B (približne 67 MB). Keďže praktický limit pre dáta prenášané protokolom UDP je 65 KB [23], jedná sa pravdepodobne o chybu pri spracovaní dát. V prípade že by hodnoty boli nezvyčajne veľké ale v povolených medziach (ako časť z nich aj je), určite by sa jednalo o podozrivú komunikáciu a je dobré že ju nástroj detekoval.

```
DOMAIN: amazonaws.com SUSPICIOUS: 76.11% COUNT: 1930 LENGTH: 1s PPS: 1930
DOMAIN: btrll.com SUSPICIOUS: 80.48% COUNT: 1245 LENGTH: 1s PPS: 1245
DOMAIN: cloudapp.net SUSPICIOUS: 60.71% COUNT: 1303 LENGTH: 1s PPS: 1303
DOMAIN: e5.sk SUSPICIOUS: 85.73% COUNT: 3846 LENGTH: 1s PPS: 3846
DOMAIN: hpecorp.net SUSPICIOUS: 81.26% COUNT: 8000 LENGTH: 3s PPS: 2666
DOMAIN: reuters.com SUSPICIOUS: 91.17% COUNT: 1416 LENGTH: 1s PPS: 1416
```

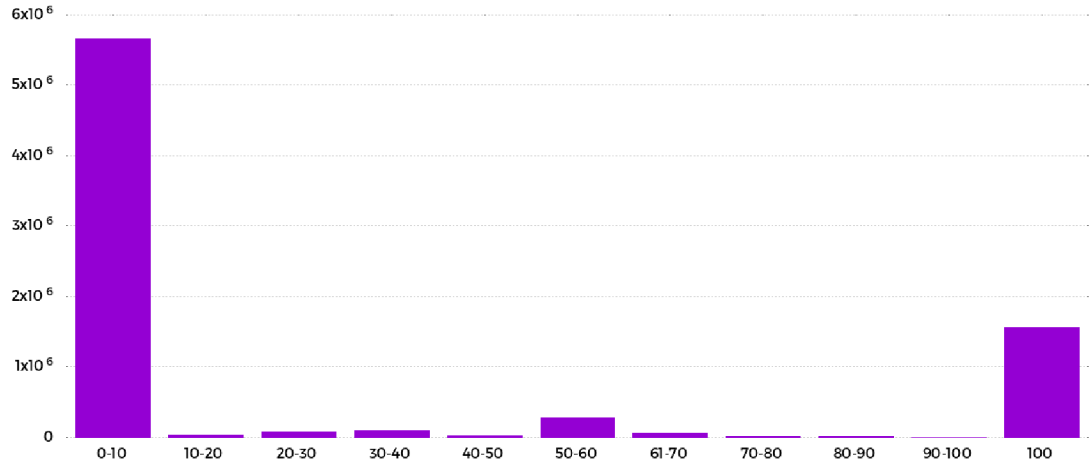
Obr. 7.5: Výpis detekčného nástroja (reálne dáta)

Výstup detekcie ďalších poskytnutých dát je na obrázku 7.5. Môžme si všimnúť, že detekované domény sú z veľkej časti zhodné s doménami na obrázku 7.4. Okná s časovou dĺžkou 1s budeme opäť pre začiatok ignorovať a zameriame sa na doménu `hpecorp.net`. Po nahliadnutí do zdrojových dát zisťujeme že dotazy smerujú len na 5 rozdielnych domén:

- `autocache.hpecorp.net`
- `epo9.austin.hpecorp.net`
- `radiacm.glb.itcs.hpecorp.net`
- `s42w0188g.bbn.hpecorp.net`
- `wssecmgr9.austin.hpecorp.net`

Štyri z týchto domén obsahujú zhluk samohlások, čo približne zodpovedá podozrivým percentám pre okno. Pri počte 8000 dotazov je počet 5 unikátnych domén príliš malý na to, aby sa jednalo o tunel. Ide tak znovu o falošné pozitívum, z podobných dôvodov ako vyššie spomínaný `reuters.com`. Spôsoby riešenia tohto problému sú opäť načrtnuté v sekcii 7.2.

Zo zvyšných, jednosekundových okien sa môžme pozrieť na tri najpodozrivejšie z obrázku 7.4. Prvým z nich je okno s doménou `apple-dns.net`. Zdrojové dáta ukazujú, že sa opäť jedná o podobný prípad ako už spomínaný `reuters.com`, tento krát ide o poddoménu *icloud*. Tomuto oknu sa preto viac venovať nebudeme, keďže nám neposkytuje žiadne nové informácie. Zameriame sa teda na ďalšie podozrivé okno, s doménou `avg.com`. Už z názvu domény môžeme tušiť že sa jedná o doménu antivírusovej spoločnosti AVG. Nahliadnutie do zdrojových dát odhaľuje množstvo dotazov na poddomény obsahujúce slová *cloud* a *download*. Opäť nezaujímavý "prípad Reuters". Posledné okno patrí k doméne `e5.sk`. V zdrojových dátach nachádzame veľké množstvo dotazov na náhodne vyzerajúce poddomény. Doména `e5.sk` patrí spoločnosti Eset a je využívaná ich antivírusovým softwarom na kontrolu prístupu k internetovým doménam pri použití rodičovskej kontroly[1]. Jedná sa teda o falošné pozitívum ktoré je však výzorovo veľmi podobné reálnemu tunelu. Rozhodnúť, či sa jedná o tunel alebo nie vyžaduje externé znalosti o doméne a jej použití ktoré sú výrazne nad rámec detekčného algoritmu. Výstup nástroja by teda mal byť overený človekom dostatočne znalým v problematike. Nástroj nevie s istotou určiť či je daná komunikácia tunel alebo nie, je však schopný zúžiť počet kandidátov na ľudsky spracovateľné množstvo.

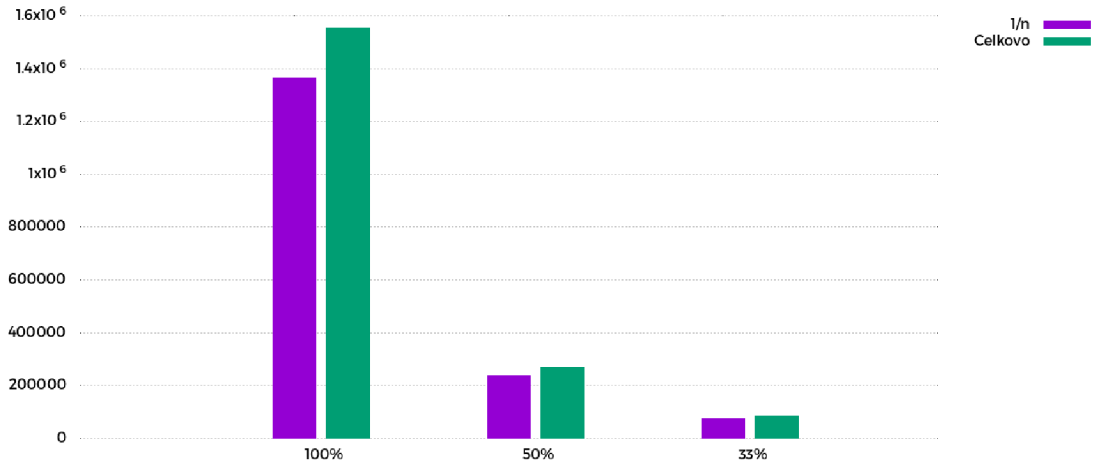


Obr. 7.6: Rozloženie okien podľa percenta podozrivosti

Obrázok 7.6 zobrazuje rozdelenie okien pre dáta z obrázku 7.4 do kategórií na základe percenta podozrivosti. Väčšina okien sa nachádza v kategórií $0-10\%$, tj. veľmi malá pravdepodobnosť tunelu, a tento stav je pre reálne dáta očakávaný. Pomerne veľký počet okien však obsahuje aj kategória 100%, čo signalizuje prítomnosť falošných pozitív. Zvyšné kategórie obsahujú relatívne málo okien v porovnaní s celkovým počtom, s menším nárastom v kategórii s hodnotou 50% a ešte menším nárastom v kategórii s hodnotou 33%. Dôvod je jednoduchý a spomínaný v práci niekoľkokrát. U okien s malým počtom dotazov stačí aby pár z nich bolo podozrivých a dosiahnu vysokú celkovú úroveň podozrivosti. Vysoký počet 100% podozrivých okien je zapríčinený jednodotazovými oknami, zvýšený počet 50% a 33% kategórií dvoj, resp. trojdotazovými oknami, kde len jeden dotaz je podozrivý. Pomer týchto (v grafe zaznačené ako *1/n*) okien a celkového počtu je zobrazený v grafe 7.7. Môžme si všimnúť, že u všetkých troch kategórií tvoria tieto okná viac ako 85% celkového počtu.

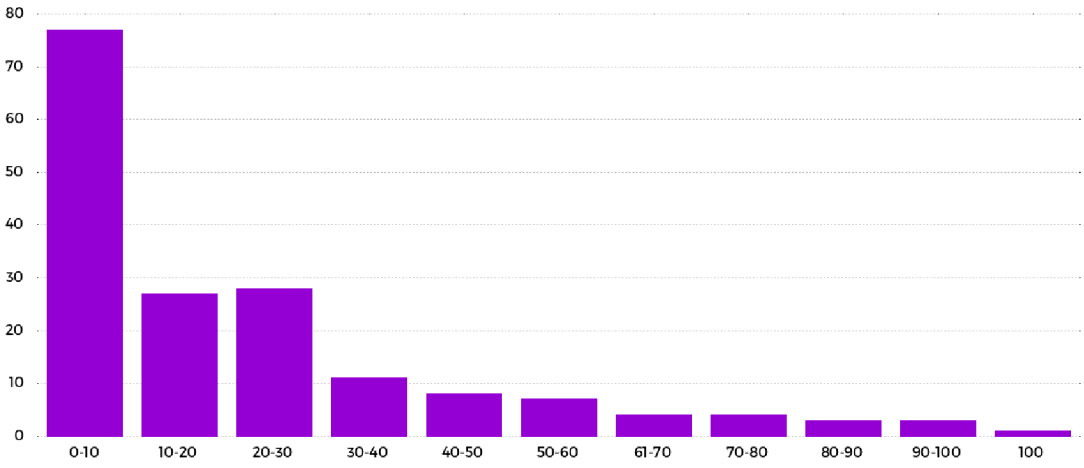
Obrázok 7.8 zobrazuje rovnaké dáta ako graf na obrázku 7.6, tentokrát však len pre okná s počtom dotazov za sekundu väčším ako 500. Už na prvý pohľad je jasné že sa rozlo-

[1] https://forum.eset.com/topic/2037-is-this-malware/



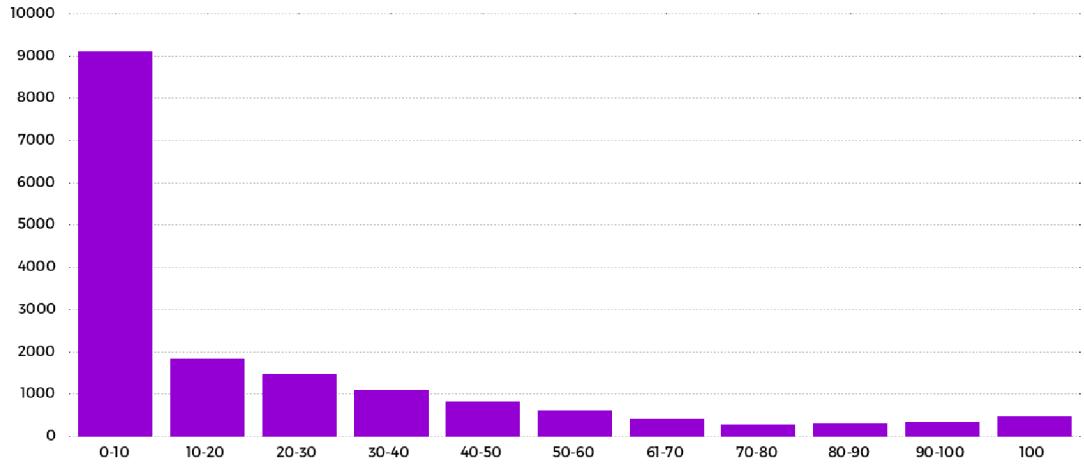
Obr. 7.7: Pomer extrémne malých okien u falošných pozitív

ženie drasticky zmenilo, ako aj klesol celkový počet okien o niekoľko rádov. Toto rozloženie má charakteristiky očakávané od reálnych dát — najviac okien patrí do kategórie 0 – 10% a až 75% okien leží v rozsahu 0 – 30%. Počet okien v kategóriách postupne klesá a najmenej (v tomto prípade 1) okien obsahuje kategória so 100% podozrivosťou. Aj jednoduché filtrovanie na základe množstva komunikácie výrazne zlepšilo kvalitu výsledkov.



Obr. 7.8: Rozloženie okien podľa percenta podozrivosti (dotaz/s > 500)

Na obrázku 7.9 sa pre porovnanie nachádzajú znova tie isté dáta, tentokrát pre okná s počtom dotazov za sekundu väčším ako 10. Počet okien v kategóriách opäť klesá, avšak v kategórii 70 – 80% sa otáča a následne stúpa. To značí že sa nepodarilo úspešne odstrániť všetky falošné pozitíva malých okien (ale väčších ako 10) a takéto filtrovanie je nedostatočné. Počet okien je taktiež 50 – 100× väčší ako v predchádzajúcom grafe a stále príliš veľký na dôkladnejšiu analýzu.



Obr. 7.9: Rozloženie okien podľa percenta podozrivosti (dotaz/s > 10)

7.2 Možnosti rozšírenia práce

Nástroj aj algoritmus sú relatívne jednoduché a možností ako ich rozšíriť sa naskytuje hneď niekoľko.

Jednou z možností je rozšíriť nástroj o podporu ďalších vstupných formátov ako napr. *pcap* alebo *IPFIX*. Nástroj má aplikačnú logiku oddelenú od čítania vstupu a pridať podporu pre nový formát je preto veľmi jednoduché. Stačí aby bola po každom načítanom pakete volaná funkcia `add_packet` s potrebnými parametrami.

Ďalšou možnosťou je vytvoriť komplexnejší detekčný algoritmus. Jednotlivým podozrivým znakom je možné pridať váhové koeficienty — napr. nezobraziteľné znaky by mali váhu veľkú, veľkosť dotazu váhu strednú a zhluky samohlások malú. Algoritmus je taktiež možné rozšíriť o ďalšie znaky a informácie ktoré si zaznamenáva. Príkladom sú dotazované typy DNS záznamov v komunikácii, či sa pre dané okno menia alebo sú konštantné, ich percentuálne rozdelenie alebo aj počet dotazujúcich sa zariadení. Je taktiež možné uchovávať si aj počet unikátnych domén — pre tunelovanú komunikáciu by mal byť počet domén optimálne polovicou počtu dotazov/odpovedí, pri dotazoch na sadu domén ktoré však majú podozrivé znaky by sa tento pomer naopak blížil k nule.

Kapitola 8

Záver

Úvod práce sa venoval systému DNS, jeho dôležitosti pre moderný Internet, spôsobu fungovania a možných využití. Uvádza taktiež problematiku zneužitia DNS a obsahuje detailnejší popis princípov a využitia DNS tunelovania.

Nasledoval popis najpopulárnejších formátov na zachytávanie sieťovej komunikácie, ich výhody, nevýhody a typické prípady použitia. Kapitola sa venuje formátom *pcap*, *NetFlow* a *IPFIX*.

Práca pokračuje prehľadom dostupných nástrojov na DNS tunelovanie s analýzou a rozborom ich komunikácie. Sú popísané spoločné črty nástrojov, ako aj netradičné charakteristiky v porovnaní s bežnou DNS komunikáciou. Detailnejšie sa práca venuje nástroju *iodine*, s ktorým boli uskutočnené benchmarky a výsledky týchto meraní a pozorovaní sú následne využité na návrh detekčného algoritmu. Nasleduje stručný popis implementácie, kde sa popisuje základný princíp, využité dátové štruktúry, konfigurovateľné parametre nástroja ako aj nutné predpoklady pre vstupné súbory.

Práca je zakončená zhodnotením schopnosti detekcie nástroja na reálnych dátach, jeho limitáciám, vyskytnutých problémoch a navrhnutím možných spôsobov rozšírenia práce. Zachytené dáta nástroja *iodine* je nástroj schopný detekovať s veľkou mierou úspešnosti. Na príkladoch je ukázaná detekcia najlepšej a najhoršej konfigurácie nástroja z pohľadu detekčného algoritmu. Veľká časť kapitoly je venovaná ukážkam detekcie na reálnych dátach, odhaľovaniu falošných pozitív a podrobnejšej analýze prečo boli detekované a navrhnutím možných úprav algoritmu na ich odstránenie. Aj keď implementovaný nástroj nie je schopný so 100% presnosťou označiť komunikáciu za tunel, s relatívne vysokou mierou spoľahlivosti dokáže znížiť množstvo komunikácie na podrobnejšiu analýzu, často z niekoľkých miliónov na jednotky až desiatky. Nástroj teda nie je pripravený na automatizované použitie, dokáže však byť užitočnou pomôckou pri manuálnej analýze komunikácie.

Literatúra

[1] Factsheet - Root server attack on 6 February 2007 [online]. 1.03.2007 [cit. 28.1.2016].
URL https://www.icann.org/en/system/files/files/factsheet-dns-attack-08mar07-en.pdf

[2] Development/LibpcapFileFormat - The Wireshark Wiki [online]. 23.8.2015 [cit. 1.2.2016].
URL https://wiki.wireshark.org/Development/LibpcapFileFormat

[3] Events of 2015-11-30 [online]. 4.12.2015 [cit. 28.1.2016].
URL http://root-servers.org/news/events-of-20151130.txt

[4] DNScat [online]. [cit. 20.2.2016].
URL http://tadek.pietraszek.org/projects/DNScat/

[5] GitHub - iagox86/dnscat2 [online]. [cit. 20.2.2016].
URL https://github.com/iagox86/dnscat2

[6] GitHub - mdornseif/DeNiSe: DeNiSe is a proof of concept for tunneling TCP over DNS in Python [online]. [cit. 20.2.2016].
URL https://github.com/mdornseif/DeNiSe

[7] Google Code Archive - Long-term storage for Google Code Project Hosting. [online]. [cit. 20.2.2016].
URL https://code.google.com/archive/p/dnscapy/

[8] Heyoka: your fast&spoofed DNS tunnel [online]. [cit. 20.2.2016].
URL http://heyoka.sourceforge.net/

[9] HSC - Tools - Dns2tcp [online]. [cit. 20.2.2016].
URL http://www.hsc.fr/ressources/outils/dns2tcp/

[10] kryo.se: iodine (IP-over-DNS, IPv4 over DNS tunnel) [online]. [cit. 20.2.2016].
URL http://code.kryo.se/iodine/

[11] Release! | Dan Kaminsky's Blog [online]. [cit. 20.2.2016].
URL https://dankaminsky.com/2004/07/29/51/

[12] tcp-over-dns [online]. [cit. 20.2.2016].
URL http://analogbit.com/software/tcp-over-dns/

[13] Root Server Technical Operations Assn [online]. [cit. 28.1.2016].
URL http://www.root-servers.org/

[14] Arends, R.; Austein, R.; Larson, M.; aj.: Protocol Modifications for the DNS Security Extensions. RFC 4035, 2005.

[15] Binsalleeh, H.; Kara, A. M.; Youssef, A.; aj.: Characterization of Covert Channels in DNS. In *2014 6th International Conference on New Technologies, Mobility and Security (NTMS)*, March 2014, ISSN 2157-4952, s. 1–5, doi:10.1109/NTMS.2014.6814008.

[16] Claise, B.: Cisco Systems NetFlow Services Export Version 9. RFC 3954, 2004.

[17] Claise, B.; Trammell, B.: Information Model for IP Flow Information Export (IPFIX). RFC 7012, 2013.

[18] Claise, B.; Trammell, B.; Aitken, P.: Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information. RFC 7011, 2013.

[19] Kara, A. M.; Binsalleeh, H.; Mannan, M.; aj.: Detection of malicious payload distribution channels in DNS. In *2014 IEEE International Conference on Communications (ICC)*, June 2014, ISSN 1550-3607, s. 853–858, doi:10.1109/ICC.2014.6883426.

[20] Lewis, E.; Hoenes, A.: DNS Zone Transfer Protocol (AXFR). RFC 5936, 2010.

[21] Mockapetris, P.: Domain Names - Concepts and Facilities. RFC 1034, 1987.

[22] Mockapetris, P.: Domain Names - Implementation and Specification. RFC 1035, 1987.

[23] Postel, J.: User Datagram Protocol. RFC 768, 1980.

[24] Vixie, P.: Extension Mechanisms for DNS (EDNS0). RFC 2671, 1999.

[25] Vixie, P.; Sneeringer, G.; Schleifer, M.: Events of 21-Oct-2002 [online]. 27.11.2002 [cit. 28.1.2016].
URL http://c.root-servers.org/october21.txt

Prílohy

Zoznam príloh

Príloha A

Tabuľky nameraných štatistík nástroja iodine

Typ záznamu	Kódovanie	Minimum	Maximum	Priemer
NULL	binárne	46 B	1470 B	650 B
NULL	base128	48 B	1470 B	654 B
NULL	base64	48 B	1470 B	649 B
NULL	base32	48 B	1470 B	655 B
PRIVATE	binárne	37 B	1470 B	648 B
PRIVATE	base128	37 B	1470 B	648 B
PRIVATE	base64	37 B	1470 B	648 B
PRIVATE	base32	35 B	1470 B	649 B
TXT	binárne	48 B	1464 B	676 B
TXT	base128	48 B	1468 B	415 B
TXT	base64	46 B	1472 B	466 B
TXT	base32	46 B	1460 B	541 B
SRV	binárne	48 B	1464 B	593 B
SRV	base128	48 B	1465 B	452 B
SRV	base64	46 B	1463 B	505 B
SRV	base32	48 B	1464 B	593 B
MX	binárne	48 B	1463 B	585 B
MX	base128	48 B	1459 B	441 B
MX	base64	48 B	1459 B	498 B
MX	base32	46 B	1463 B	584 B
CNAME	binárne	48 B	537 B	177 B
CNAME	base128	46 B	537 B	174 B
CNAME	base64	46 B	536 B	172 B
CNAME	base32	46 B	537 B	177 B
A	binárne	46 B	537 B	177 B
A	base128	48 B	537 B	173 B
A	base64	48 B	536 B	172 B
A	base32	48 B	537 B	177 B

Tabuľka A.1: Veľkosti paketov nástroja iodine

Typ záznamu	Kódovanie	Minimum	Maximum	Priemer
NULL	binárne	0	7	1.53
NULL	base128	0	7	1.63
NULL	base64	0	7	1.58
NULL	base32	0	7	1.59
PRIVATE	binárne	0	7	1.88
PRIVATE	base128	0	7	1.69
PRIVATE	base64	0	7	1.68
PRIVATE	base32	0	7	1.61
TXT	binárne	0	7	1.49
TXT	base128	0	7	1.68
TXT	base64	0	7	1.74
TXT	base32	0	7	1.67
SRV	binárne	0	7	1.66
SRV	base128	0	7	1.77
SRV	base64	0	7	1.68
SRV	base32	0	7	1.64
MX	binárne	0	7	1.84
MX	base128	0	7	1.66
MX	base64	0	7	1.77
MX	base32	0	7	1.87
CNAME	binárne	0	7	1.40
CNAME	base128	0	7	1.42
CNAME	base64	0	7	1.44
CNAME	base32	0	7	1.44
A	binárne	0	7	1.42
A	base128	0	7	1.47
A	base64	0	7	1.44
A	base32	0	7	1.40

Tabuľka A.2: Dĺžky zhlukov samohlások nástroja iodine

Typ záznamu	Kódovanie	Minimum	Maximum	Priemer
NULL	binárne	3	10	3.38
NULL	base128	3	11	3.40
NULL	base64	3	10	3.36
NULL	base32	3	11	3.42
PRIVATE	binárne	3	13	3.36
PRIVATE	base128	3	11	3.42
PRIVATE	base64	3	12	3.41
PRIVATE	base32	3	10	3.43
TXT	binárne	3	13	3.54
TXT	base128	3	10	3.33
TXT	base64	3	13	3.37
TXT	base32	3	13	3.34
SRV	binárne	3	10	3.31
SRV	base128	3	10	3.35
SRV	base64	3	12	3.33
SRV	base32	3	10	3.31
MX	binárne	3	11	3.33
MX	base128	3	10	3.35
MX	base64	3	10	3.38
MX	base32	3	13	3.36
CNAME	binárne	3	13	3.41
CNAME	base128	3	11	3.39
CNAME	base64	3	10	3.38
CNAME	base32	3	13	3.38
A	binárne	3	10	3.38
A	base128	3	12	3.39
A	base64	3	10	3.39
A	base32	3	10	3.39

Tabuľka A.3: Dĺžky zhlukov spoluhlások nástroja iodine

Typ záznamu	Kódovanie	Minimum	Maximum	Priemer
NULL	binárne	0.00%	79.84%	18.04%
NULL	base128	0.00%	91.70%	19.72%
NULL	base64	0.00%	91.30%	18.65%
NULL	base32	0.00%	80.24%	19.62%
PRIVATE	binárne	0.00%	91.30%	19.06%
PRIVATE	base128	0.00%	80.24%	19.51%
PRIVATE	base64	0.00%	80.24%	18.42%
PRIVATE	base32	0.00%	71.88%	18.50%
TXT	binárne	0.00%	80.24%	24.17%
TXT	base128	0.00%	52.96%	12.80%
TXT	base64	0.00%	79.84%	12.36%
TXT	base32	0.00%	91.70%	12.86%
SRV	binárne	0.00%	91.70%	12.80%
SRV	base128	0.00%	79.84%	12.73%
SRV	base64	0.00%	71.88%	12.78%
SRV	base32	0.00%	91.70%	12.75%
MX	binárne	0.00%	68.77%	12.29%
MX	base128	0.00%	91.70%	12.43%
MX	base64	0.00%	80.24%	12.93%
MX	base32	0.00%	91.70%	11.54%
CNAME	binárne	0.00%	53.60%	3.43%
CNAME	base128	0.00%	71.88%	4.53%
CNAME	base64	0.00%	71.88%	3.61%
CNAME	base32	0.00%	71.88%	3.13%
A	binárne	0.00%	91.70%	3.23%
A	base128	0.00%	91.30%	4.12%
A	base64	0.00%	54.50%	3.68%
A	base32	0.00%	56.92%	3.22%

Tabuľka A.4: Percentá nezobraziteľných znakov nástroja iodine

Štatistika	Minimum	Maximum	Priemer
Veľkosť	20 B	1208 B	195 B
Samohlásky	0	5	0.47
Spoluhlásky	2	10	2.54
Nezobraziteľné znaky	0.00%	0.00%	0.00%

Tabuľka A.5: Štatistiky bežnej DNS komunikácie