UNIVERZITA PAVLA JOZEFA ŠAFÁRIKA V KOŠICIACH
PRÍRODOVEDECKÁ FAKULTA

DETEKCIA MALVÉRU POMOCOU DNS ÚDAJOV

2018 **Martina PIVARNÍKOVÁ**

UNIVERZITA PAVLA JOZEFA ŠAFÁRIKA V KOŠICIACH
PRÍRODOVEDECKÁ FAKULTA

DETEKCIA MALVÉRU POMOCOU DNS ÚDAJOV

BAKALÁRSKA PRÁCA

Študijný program:	informatika
Pracovisko (katedra/ústav):	Ústav informatiky
Vedúci bakalárskej práce:	RNDr. JUDr. Pavol Sokol, PhD .
Konzultant bakalárskej práce:	Mgr. Tomáš Bajtoš

Košice 2018 **Martina PIVARNÍKOVÁ**

Univerzita P. J. Šafárika v Košiciach
Prírodovedecká fakulta

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Martina Pivarníková
Študijný program: Informatika (Jednoodborové štúdium, bakalársky I. st., denná forma)
Študijný odbor: 9.2.1. informatika
Typ záverečnej práce: Bakalárska práca
Jazyk záverečnej práce: slovenský
Sekundárny jazyk: anglický

Názov: Detekcia malvéru pomocou DNS údajov

Názov EN: Detection of malware using DNS data

Cieľ: 1) Analyzovať a spracovať možnosti detekcie škodlivého kódu a botnetov
2) Porovnať metódy používané pri detekcii bezpečnostných incidentov pomocou DNS údajov
3) Navrhnúť a implementovať systém pre detekciu malvéru. Systém vyhodnotiť

Literatúra:
[1] VU HONG, Linh. DNS Traffic Analysis for Network-based Malware Detection. 2012.
[2] SILVA, Sérgio SC, et al. Botnets: A survey. Computer Networks, 2013, 57.2: 378-403.
[3] AITCHISON, Ron. Pro Dns and BIND 10. Apress, 2011.
[4] ALIEYAN, Kamal, et al. A survey of botnet detection based on DNS. Neural Computing and Applications, 2015, 1-18.

Vedúci: RNDr. JUDr. Pavol Sokol, PhD.
Konzultant: Mgr. Tomáš Bajtoš
Ústav : ÚINF - Ústav informatiky
Riaditeľ ústavu: prof. RNDr. Viliam Geffert, DrSc.

Dátum schválenia: 09.05.2018

Poďakovanie

Týmto sa chcem poďakovať vedúcemu svojej práce RNDr. JUDr. Pavlovi Sokolovi, PhD. a konzultantovi Mgr. Tomášovi Bajtošovi za odborné vedenie, cenné rady a veľkú pomoc počas tvorby práce.

Abstrakt v štátnom jazyku

Jednou z top 15 hrozieb internetu podľa ENISA Threat Landscape sú boti a botnety. Táto práca je zameraná na detekciu malvéru pomocou DNS záznamov. Hlavným cieľom práce je navrhnúť a aplikovať metódy, ktoré budú účinné na odhalenie prítomnosti malvéru na hostiteľskom zariadení v počítačovej sieti. Zameriava sa hlavne na botnety, ktoré využívajú algoritmy generovania domén. Na odfiltrovanie legitímnej DNS prevádzky sa využíva zoznam povolených doménových mien (tzv. whitelist). Ďalšou filtráciou je zoznam škodlivých doménových mien (tzv. blacklist), podľa ktorého sa jednoznačne určí, ktorá doména je škodlivá. Vygenerované doménové mená často nemajú žiadnu štruktúru a väčšinou sú zložené z náhodných znakov. Z tohto dôvodu je tiež analyzovaná skladba doménového mena. Práce sa sústredí aj na sledovanie DNS odpovedí, pomocou ktorých sa dá indikovať, či je zariadenie v počítačovej sieti infikované.

Kľúčové slová: bot, botnet, DNS, malvér, detekčný systém

Abstrakt v cudzom jazyku

One of the top fifteen threats by ENISA Threat Landscape are bots and botnets. This work is focused on detecting malware using DNS records. The main goal is to design and apply methods that will be effective in detecting the presence of malware on the host in the network. Focus is on botnets that use domain generating algorithms. To filter legitimate DNS traffic, a list of legitimate domain names (so-called whitelist) is used. Another filtering is a list of malicious domain names (so-called blacklist) that determine which domain is harmful. Generated domain names often have no structure and are mostly composed of random characters. For this reason, domain name structure is also analysed. This work is likewise focused on tracking DNS responses to help determine whether the device in a computer network was infected.

Keywords: bot, botnet, DNS, malware, detection system

Obsah

Zoznam ilustrácií

Úvod

V súčasnosti má internet veľa využití v rôznych aspektoch každodenného života ľudí. Je súčasťou skoro každej organizácie, ale taktiež aj domácností. Zariadenia pripojené k internetu sú často nedostatočne zabezpečené. Jednou z hlavných služieb internetu je systém DNS. Bez neho by dnešný internet nemohol fungovať tak, ako ho poznáme, pretože by sme namiesto doménových mien boli donútení používať IP adresy. A vďaka rozšírenosti jeho služieb sa dá využiť na množstvo iných vecí, napríklad aj na vykonávanie škodlivej aktivity. Bezpečnostných hrozieb je pomerne veľa a v poslednom čase sa niektoré z nich rozvíjajú obrovskou rýchlosťou. Jednou z nich sú aj botnety, ktorými sa budeme venovať v tejto práci. Útočníci distribuujú druhy malvéru, ktoré premenia hostiteľa na bot. Infikovaný hostiteľ môže byť využívaný na automatizované úlohy na internete, bez toho aby o tom používateľ zariadenia vedel. Botnet je teda tvorený sieťou zariadení, ktoré boli infikované nejakým druhom malvéru.

Keďže sa riešenia na odhalenie botnetov stále vyvíjajú, botmastri taktiež stále vylepšujú svoje botnety tak, aby sa dali ťažšie odhaliť. Vyvíjajú rôzne riešenia, vďaka ktorým je detekcia botnetov a obrana voči nim stále väčšou výzvou. Preto je dôležité, aby sa vyvíjali a vylepšovali aj riešenia na ich monitorovanie. V rámci vývoja začali botnety využívať DNS techniky, aby sa vyhli detekčným mechanizmom. Je to najmä z dôvodu, že DNS prevádzka je veľmi veľká. DNS využívajú na to, aby boti lokalizovali Command & Control servery a pripojili sa k nim, alebo dokonca na komunikáciu a predávanie si informácií a príkazov.

V práci sa zameriavame najmä na botnety, ktoré využívajú algoritmy generovania domén. Využívame skutočnosť, že domény generované takýmito algoritmami sa často líšia skladbou a vlastnosťami doménového mena. Na to, aby sa dali odhaliť, je potrebné analyzovať údaje, ktoré sa získavajú zo sieťovej prevádzky. Úlohou tejto práce je detailná analýza týchto údajov s cieľom detegovať anomálie v sieti, ktoré by mohli byť indikátorom prítomnosti infikovaných hostiteľov.

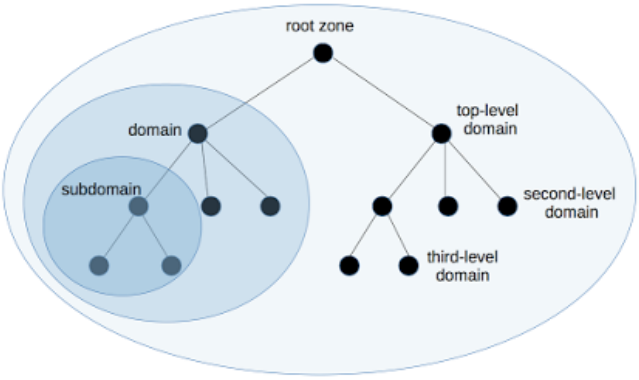
V prvej kapitole popisujeme funkcionalitu DNS, jeho architektúru a typy záznamov. Druhá kapitola obsahuje definície bota a botnetu, spôsob komunikácie a rôzne DNS metódy, ktoré využívajú. Následne porovnávame existujúce metódy na detekciu botnetov. Na základe toho potom navrhneme vlastný systém využívajúce rôzne metódy na detekciu malvéru a botnetov.

1 DNS

Všetky zariadenia pripojené k internetu využívajú IP adresy na komunikáciu. Ak by sme chceli pristupovať k zdroju dostupnému v počítačovej sieti, museli by sme poznať jeho fyzickú adresu. S existujúcimi miliónmi hostiteľmi a webstránkami je taká úloha nemožná. Pre uľahčenie práce s internetom bol preto vyvinutý systém doménových mien (Domain Name System - DNS). Tento protokol prekladá IP adresy do ľudského jazyka, teda do textových reťazcov, ktoré sú pre ľudí ľahšie zapamätateľné. Jeho hlavnou úlohou je mapovanie IP adries na doménové mená. Okrem toho poskytuje ďalšie preklady a informácie o entitách. Ponúka reverzný preklad IP adries na doménové mená, určenie autoritatívneho serveru danej domény alebo určenie mailového serveru priradeného k danej doméne.

1.1 Architektúra

Doménové mená sú usporiadané do hierarchickej stromovej štruktúry. Táto štruktúra je znázornená na obrázku č.1.

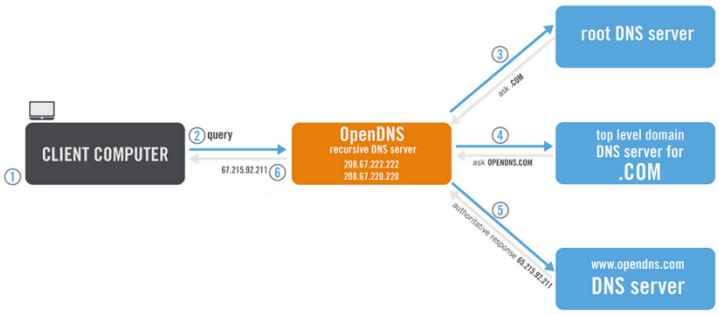


Obr. 1 Hierarchia DNS [1]

Na vrchole hierarchie sa nachádza 13 koreňových DNS serverov. Tie sú nasledované top-level domain (TLD) servermi. Každý z nich je zodpovedný za jednu z koncoviek domény, napr. com, net, org, sk a podobne. Spravujú záznamy o DNS serveroch prvej úrovne. V hierarchii sa pod nimi následne nachádzajú druhostupňové domény (second-level domain), treťostupňové domény (third-level domain) a tak ďalej.

1.2 DNS servery

Zvyčajne sa prevádzkujú lokálne (rekurzívne) DNS servery, ktoré však nespravujú žiadne domény. Slúžia ako predvolené servery koncových zariadení nachádzajúcich sa v počítačovej sieti. **Rekurzívne DNS servery** sú zodpovedné za poskytnutie IP adresy príslušného doménového mena dopytujúcemu hostiteľovi. Všetky zistené záznamy si ukladajú do svojho lokálneho úložiska. Často používané záznamy sa preto nemusia stále dopytovať na externé DNS servery, čím sa odľahčuje prevádzka. Hostiteľ žiada rekurzívny server DNS, aby našiel IP adresu priradenú k doméne. Ak server nemá záznam v cache, začne rekurzívny proces získania IP adresy [2]. Postupne dopytuje root server, TLD server a nakoniec získa adresu autoritatívneho servera. **Autoritatívne DNS servery** sú zodpovedné za poskytovanie odpovedí na rekurzívne DNS servery. Tieto odpovede obsahú príslušné IP adresy a ďalšie potrebné údaje o doméne [2]. Existujú dva typy autoritatívnych DNS serverov – primárne a sekundárne. Primárne obsahujú originálne informácie o všetkých zónových záznamoch. Sekundárne sú presnými kópiami primárnych. Využívajú sa, ak primárny server zlyhá, prípadne na rozloženie záťaže.



Obr. 2 Proces zisťovania IP adresy [2]

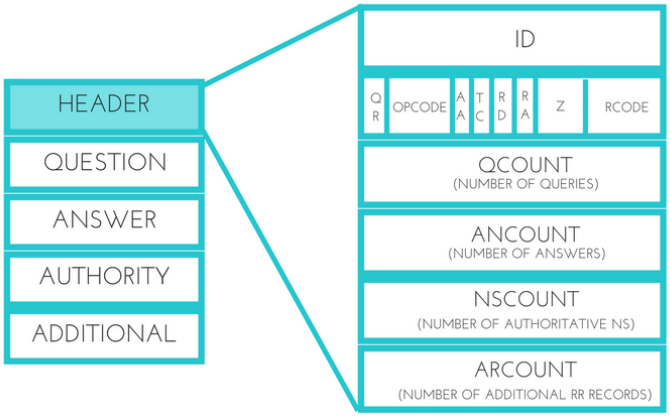
1.3 Typy DNS záznamov

Celý tento systém spravuje obrovskú databázu záznamov rôznych typov. V tejto práci sledujeme nasledujúce typy záznamov [3]:

- **A** - tento záznam je štandardizovaný v RFC 1035 [4] a definuje IPv4 adresu priradenú k doméne,
- **AAAA** - analógia záznamu A pre IPv6 definovaný v RFC 3596 [5],
- **CNAME** - alias pre existujúceho hostiteľa definovaného A záznamom,
- **PTR** - záznam pre reverzný preklad IP adries na doménové mená.

1.4 DNS paket

DNS požiadavky a odpovede majú rovnaký formát paketu. Na obrázku č.3 môžeme vidieť jeho štruktúru. Hlavička obsahuje identifikátor, rôzne príznaky (napr. príznak požiadavky alebo odpovede, príznak autoritatívnej odpovede alebo príznak požadovanej rekurzie) a počty záznamov, ktoré obsahuje paket. Ten ďalej obsahuje sekciu otázky, v ktorej je dopytované doménové meno. V sekcii odpovede sa nachádzajú vrátené IP adresy pre danú doménu a ďalšie informácie o nich. Autoritatívna sekcia obsahuje informáciu o autoritatívnom serveri [4]. Každý DNS server, ktorý obsahuje kompletnú kópiu zónového súboru domény je považovaný za autoritatívny pre danú doménu.



Obr. 3 Štruktúra DNS paketu

2 Bot a botnet

V súčasnosti bojuje internetová bezpečnosť s prudkým vývojom bezpečnostných hrozieb. Útočníci používajú pokročilejšie techniky, najmä v prípade útokov zameraných na nadnárodné organizácie. V mnohých prípadoch útokov sa botnet využíva na zvýšenie počtu hostiteľov zapojených do útoku. Takéto útoky sú dnes jednou z najväčších hrozieb pre bezpečnosť internetu.

2.1 Definícia bota a botnetu

Bot je aplikácia, ktorá môže vykonávať a opakovať konkrétnu úlohu [6]. Keď sa veľké množstvo botov rozšíri na hostiteľov a prepoja sa cez internet, sformujú skupinu nazývanú botnet, čiže sieť botov [6]. Botnet sa skladá z troch hlavných elementov [7]:

- botov,
- riadiacich serverov - command and control serverov (C&C) a
- botmastra.

Botnet je kontrolovaný útočníkom, čiže **botmastrom** prostredníctvom **riadiacich serverov**. Takáto infraštruktúra zvyčajne slúži ako jediný spôsob, ako efektívne kontrolovať botov. Typický botnet je vytvorený a udržiavaný v piatich fázach: prvotné nakazenie, druhotné nakazenie, pripojenie, škodlivý príkaz, aktualizácia a údržba [8]. Infraštruktúra botnetu umožňuje vykonávať rôzne typy útokov.

2.2 Použitie

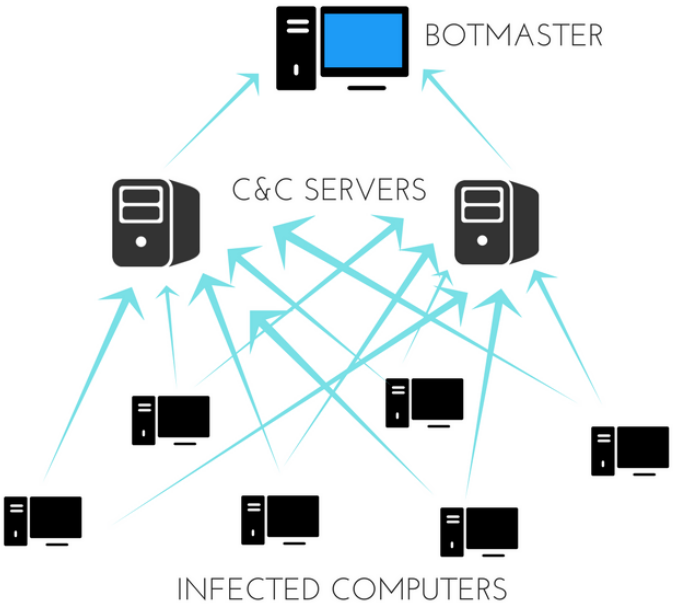
Existuje široká škála nekalých aktivít, na ktoré sa dajú botnety využiť. Obeťou môže byť napríklad samotný hostiteľ, ktorý bol infikovaný botom. V tomto prípade môže byť cieľom krádeže údajov – hesiel, údajov o kreditných kartách a rôznych iných osobných dát, ktoré môže útočník zužitkovať.

Táto sieť dá taktiež využiť na sprostredkovanie útoku na iný cieľ. Existuje veľa druhov takéhoto typu útoku, napríklad šírenie nevyžiadanej pošty (spamu) posielanej botnetmi. Taktiež sa veľmi často využíva na distribuované útoky odopretia služby (distributed denial of service, DDoS). V neposlednom rade sa v súčasnosti rozbieha predaj a prenájom botnetov, pričom cieľom útočníka je vytvoriť botnet a neskôr ho sprostredkovať na rôzne škodlivé aktivity.

2.3 Architektúra

Existujú dva modely botnetu - C&C model a peer-to-peer. **Peer-to-peer botnet** je decentralizovaná sieť botov. Server, ktorý poskytuje príkazy sa tvári ako bot. Tieto botnety sú flexibilnejšie, avšak nie sú dostatočne vyspelé a majú mnoho slabých stránok. Tieto robustné siete sa síce snažia ukryť odosielateľa správy, avšak väčšinou sa nesnažia skryť všetky uzly botnetu [9]. V tejto práci sa zaoberáme **centralizovaným modelom botnetu**, ktorý využíva C&C servery na sprostredkovanie komunikácie. Obrázok č.4 ukazuje architektúru tohto modelu. Hlavnou myšlienkou tohto modelu je, že po infikovaní hostiteľa škodlivým kódom, sa bot pripojí k C&C serveru a ďalej bude prostredníctvom neho dostávať príkazy.

Pre pripojenie bota k serveru je využívaný DNS protokol. Ku každému C&C serveru je pridelené minimálne jedno doménové meno, na ktoré pošle bot DNS požiadavku na preklad na IP adresu servera a takto ju získa. Bot môže mať pridelený zoznam doménových mien, kde sa môže riadiaci server nachádzať. To by ale mohlo viesť k ľahšiemu odhaleniu celého botnetu a jeho vyradeniu z prevádzky.



Obr. 4 Centralizovaný model botnetu

2.4 Fast flux a Domain flux

Botnety využívajú rôzne prístupy založené na DNS protokole, aby sa vyhli detekčným mechanizmom. Je to najmä z dôvodu, že DNS prevádzka je veľmi veľká. Botnety najčastejšie používajú dva prístupy na účinnejšie ukrytie svojej aktivity – Fast flux a Domain flux.

DNS fast flux sa vyskytuje, ak sa doména mapuje na viacero IP adries typu A, z ktorých každá ma veľmi krátku hodnotu TTL. To znamená, že doména sa prekladá na rôzne IP adresy v krátkych časových úsekoch. Fast flux sa zvyčajne používa v spojení s proxy servermi. Tie obvykle bežia na kompromitovaných hostiteľoch. Tieto proxy smerujú požiadavky na skutočného klienta. Pretože IP adresy sa menia veľmi rýchlo, je detekcia reálneho klienta sťažená. Táto metóda zriedka využíva aj AAAA záznamy [10].

Avšak fast flux používa iba jedno doménové meno, ktoré predstavuje jediný bod zlyhania. V rámci vývoja preto botnety začali využívať iný prístup na lokalizovanie C&C serverov, ktorý sa nazýva **domain flux**. Každý bot dynamicky generuje rozsiahly zoznam doménových mien. Tento zoznam je generovaný nezávisle každým botom. Na skutočné používanie C&C servermi sa využíva iba malá podmnožina týchto domén. Preto, ak sa chce bot pripojiť na C&C server, posiela požiadavky na DNS server na tieto vygenerované domény. Ak je nejaká z nich neexistujúca, bot sa presúva na nasledujúcu doménu v zozname kým nedostane odpoveď s IP adresou [11]. Väčšina odpovedí, ktoré prídu hostiteľovi budú NXDOMAIN – neexistujúca doména. Preto ak na nejakého hostiteľa príde veľký počet NXDOMAIN response, je možné, že bol infikovaný.

2.5 DGA algoritmus

Botnety založené na domain flux prístupe súčasne využívajú **DGA algoritmus** (**Domain generation algorithm**). Je to najmä z dôvodu sťaženia ich detekcie. Útočníci sa snažia predísť zaradeniu C&C domén do blacklistov. DGA je takisto ťažšie rozpoznateľné v porovnaní s tým, ak by boli IP adresy alebo domény priamo prítomné v kóde bota. Tento algoritmus umožňuje generovať pseudonáhodné doménové mená. Tie sa generujú z nejakého základu, napríklad z dátumu alebo času. Obe strany poznajú a používajú tento algoritmus. Preto sa zhodnú v nejakej malej podmnožine doménových mien. Tieto doménové mená vyzerajú často náhodne a málo sa podobajú prirodzenému

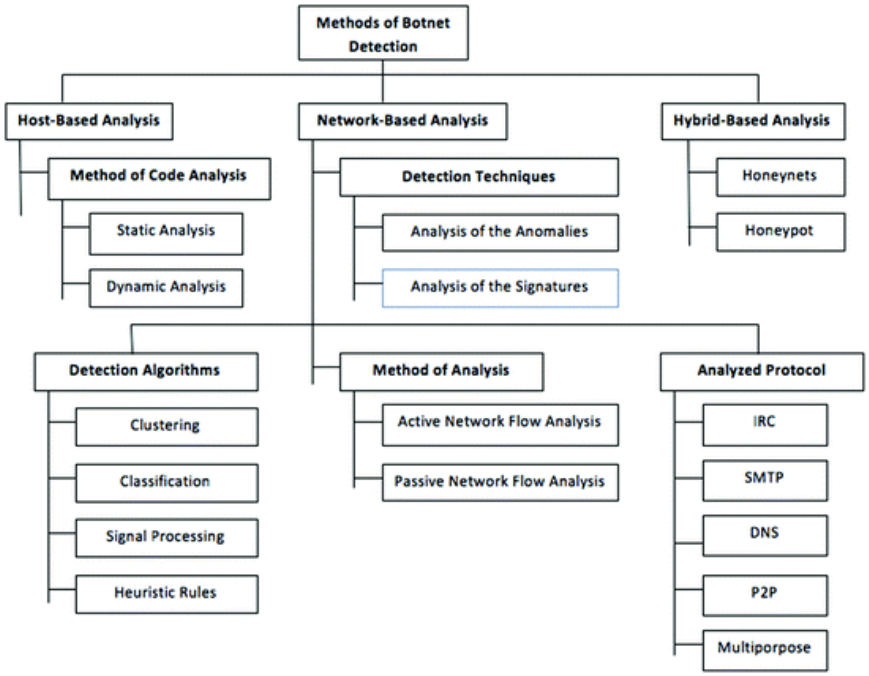
jazyku, preto sa dajú v DNS prevádzke sledovať anomálie, ktoré by objavili tieto vygenerované doménové mená. Na obrázku č.5 môžeme vidieť ukážku takýchto doménových mien.

Obr. 5 DGA doménové mená [12]

Botnety využívajúce takéto algoritmy sa môžu líšiť v spôsobe použitia algoritmov, ale aj vstupu, ktorý tieto algoritmy využívajú a vo frekvencii generovania. Napríklad, Conficker-A boti generujú 250 domén každé tri hodiny využívaním aktuálneho času a dátumu ako základ. Tie boli získané poslaním prázdnej HTTP GET požiadavky na legitímne stánky [13]. Ďalšia verzia, Conficker-C už zvýšila počet botom generovaných domén na 50 tisíc [14]. Botnet s názvom Torpig využíval ako základ pre generovanie reťazcov najpopulárnejšie príspevky na Twitteri [11]. Veľmi sofistikovaný spôsob využíva Kraken, ktorý vytvára reťazce podobné anglickému jazyku. Tie sa navyše končia často používanými príponami [15].

2.6 Detekcia botnetov

Z dôvodu veľkého počtu botnetov a ich diverzite boli voči nim implementované rôzne spôsoby obrany. Po zohľadnení rôznych prístupov je možné navrhnúť klasifikáciu rôznych metód na detekciu botnetov [16]. Táto klasifikácia je zobrazená na obrázku č.6.



Obr. 6 Klasifikácia metód na detekciu botnetov [16]

Metódy detekcie založené na analýze počítača (tzv. Host-Based) môžu byť použité počas životného cyklu botnetu. Vyžadujú fyzický prístup k infikovanému zariadeniu. Po zabezpečení škodlivého softvéru je možné vykonať reverzné inžinierstvo a získať informácie o adrese riadiaceho servera, o funkciách a kľúčových indikátoroch botnetu. Tieto informácie môžu v budúcnosti pomôcť detegovať botnet automatickým spôsobom. Pri využívaní tejto metódy je kľúčovým aspektom lokalizácia infikovaného hostiteľa v počítačovej sieti.

Metódy detekcie založené na analýze počítačovej siete sa využívajú najčastejšie, pretože nepotrebujú fyzický prístup k napadnutému hostiteľovi. Umožňujú detekciu infikovaných zariadení na základe sieťovej prevádzky. Dokážu rozpoznať aj tie botnety, ktoré ešte neboli použité na žiadny útok a stále sú vo fáze tvorby. Potenciálne infikované zariadenie môže byť následne analyzované pomocou metód založených na analýze zariadenia.

Metóda založená na analýze sieťovej prevádzky pozostáva zo štyroch menších podskupín metód rozdelených na základe rôznych faktorov. Prvá zahŕňa metódy, ktoré sú zamerané na charakteristiku sieťovej prevádzky, najmä anomálie a signatúry generované botnetmi. Ďalšia podskupina rozdeľuje metódy vďaka ich spôsobu pôsobenia: aktívne a pasívne. Prvá z nich poukazuje na metódy, ktoré upravujú sieťovú

prevádzku na získanie informácií o potenciálne infikovaných počítačoch, ktoré sú súčasťou botnetu. V súčasnosti sú efektívnejšie metódy založené na pasívnej analýze sieťovej prevádzky. Umožňujú detegovať boty neznámych botnetov tým, že analyzujú iba prevádzku bez jej modifikácie.

Metóda detekcie botnetov založená na získavaní údajov (zhlukovanie, klasifikácia atď.) taktiež využíva pasívnu analýzu [17]. Zástupcom tejto skupiny je napríklad BotMiner, ktorý navrhli Gu a spol. [18]. Ich metóda využíva zhlukovanie komunikácie a aktivitu prevádzky.

Posledná podskupina rozdeľuje **metódy detekcie botnetov podľa protokolov**, ktoré sa používajú na internú komunikáciu medzi botmi a C&C servermi. Binkley a Singh [19] navrhli metódu založenú na IRC komunikácii. Manasrah a spol. [20] navrhli mechanizmus založený na DNS, ktorý je však obmedzený, pretože využíva MAC adresu ako identifikátor namiesto IP adresy. To robí túto metódu nepoužiteľnou mimo lokálnej počítačovej siete.

Metódy v skupine **hybridnej analýzy** sú založené na pasciach na útočníkoch (honeypotoch), resp. ich sieťach (honeynetoch), ktoré umožňujú zistiť veľký počet botnetov. Ide o kombináciu metód detekcie založených na hostiteľovi a založených na analýze siete. Ak je detegovaná podozrivá aktivita v sieťovej prevádzke alebo na virtuálnom počítači, napadnuté zariadenie sa analyzuje pomocou statickej alebo dynamickej analýzy host-based metódy.

V tejto práci sa venujeme pasívnej analýze sieťovej prevádzky. Hľadáme anomálie a signatúry generované botnetmi. Súčasne využívame heuristické metódy na detekciu botnetov a údaje z DNS protokolu. V nasledujúcej kapitole si bližšie priblížime detekciu botnetov pomocou DNS.

3 Prístupy k detekcii botnetov pomocou DNS

Vzhľadom na anomálie, ktoré botnety generujú sme sa rozhodli pre delenie prístupov k detekcii botnetov pomocou DNS na základe:

- DNS odpovedí,
- doménového mena,
- opakújúcich sa vzorcov a iných vlastností

3.1 Na základe DNS odpovedí

Mnoho botnetov využíva fast-flux DNS prístup na skrývanie riadiacich centier. Táto metóda dokáže meniť IP adresu domény každých pár sekúnd. Na jedno doménové meno môže byť taktiež mapovaných viacero IP adries, niekedy až stovky. Taktiež sa využíva domain-flux na časté menenie doménových mien priradených ku kontrolným serverom. Vďaka týmto technikám je detekcia C&C sťažená. Avšak existujú niektoré indikátory, ktoré by mohli pomôcť detegovať botov.

Brustoloni a spol. [21] sa sústredili na TTL domény. Na základe toho taktiež zvolili selekciu dát. Množiny obsahujú len DNS odpovede s hodnotami TTL najviac 60, 300, prípadne 600 sekúnd alebo odpovede s NXDOMAIN kódom. Rozdelili dáta do troch množín – CSAA, CS_NS a DDNS_ NS. Prvá množina CSAA obsahovala iba odpovede s NOERROR kódom. Druhá s názvom CS_NS obsahovala NXDOMAIN odpovede. Posledná množina nazývaná DDNS_NS obsahuje odpovede od známych poskytovateľov DDNS s návratovými kódmi NOERROR alebo NXDOMAIN. Pre každú doménu druhého stupňa v každej množine vypočítali Čebyševovu nerovnosť s parametrom k = 4,47. Takto hľadali anomálie v prevádzke. Nakoniec skúmali, či sú tieto domény skutočne škodlivé. Najlepšie výsledky dávala podmnožina CS_NS, kde skoro všetky domény, ktoré vykazovali anomálie boli skutočne označené aj ako škodlivé.

Detekciu fast-flux pomocou sledovania TTL odpovede navrhli **Stalmans a spol. [22]**. Poukázali na fakt, že TTL pre takéto domény je menší ako 600, pri legitímnych doménach väčší ako 1800. Ďalej pozorovali výskyt viacero IP adries v odpovedi. Ak sú v iných rozsahoch a každá má fyzickú lokalizáciu na inom mieste vo svete, je možné že takáto doména patrí C&C serveru. Predstavili dva klasifikátory.

Prvý z nich je **C5.0 rozhodovací strom**, ktorý bol zostrojený za použitia cvičných údajov pozostávajúcich z testovacích domén, ktoré boli ručne vyhodnotené a označené ako fast-flux alebo legitímne. Vytvára čitateľné a ľahko interpretovateľné rozhodnutia. Tie môžu byť preskúmané s cieľom identifikovať vlastnosti, ktoré najpravdepodobnejšie identifikujú fast-flux domény. Druhou metódou je použitie **Bayesovského klasifikátora**. Bayesovská inferencia je štatistická technika, ktorá je užitočná pri klasifikácii problémov, ktoré majú binárny výsledok. Použite takéhoto klasifikátora je na vypočítanie pravdepodobnosti, či je doména fast-flux alebo nie. Tento klasifikátor sa tiež učil z tréningových dát. Produkuje celkovú pravdepodobnosť, ktorá označuje fast-flux. Klasifikátor používa vzorec odvodený od Bayesovej vety. Pri identifikácii podozrivých domén sa sústreďovali na rozdelenie pravdepodobností alfanumerických znakov. Taktiež si všímali celkovú odchýlku vzdialeností rozdelení pravdepodobnosti legitímnych a generovaných domén.

Systém s názvom Pleiades, vyvinutý **Antonakakisom a spol. [12]** dokázal identifikovať nové DGA botnety, infikované stroje a taktiež nájsť a zablokovať aktívne C&C domény. Tento systém pozostáva z dvoch hlavných modulov - DGA Discovery modul a DGA Classification and C&C Detection modul. Pri DGA botnetoch prichádza na hostiteľa veľké množstvo NXDOMAIN odpovedí. Preto boli zhromažďované iba takéto sekvencie neexistujúcich domén, ktoré boli ďalej analyzované. Rozhodli sa využiť delenie do klastrov pomocou bipartitného grafu, ktorého vrcholy sú hostitelia a NXDOMAINs. Preto ak takéto domény majú viac spoločných hostiteľov, tak pravdepodobne na týchto hostiteľoch beží rovnaký DGA algoritmus. Ďalšími krokmi bolo vyradenie domén generovanými známymi DGA. Nakoniec sú klastre takýchto domén označené ako dosiaľ neznámy druh DGA botnetu. Tento systém dokázal objaviť 6 nových botnetov a taktiež niekoľko už existujúcich.

Schonewille a Van Helmond [23] predstavili prístup založený na abnormálne sa opakujúcich NXDOMAIN odpovediach. Využívajú algoritmy na klasifikovanie podobných dopytov. Podľa ich pozorovania, DNS odpovede s kódom NXDOMAIN často zodpovedajú aktivite infikovaného hostiteľa snažiaceho sa pripojiť na C&C server.

3.2 Na základe doménového mena

Tento prístup je založený na monitorovaní anomálií v DNS požiadavkách a odpovediach na ne. Sústredí sa na identifikáciu doménových mien, ktoré boli automaticky generované pomocou algoritmu. Skladba takýchto domén je iná ako u legitímnych doménových mien. V zozbieraných údajoch sa hľadajú rôzne odchýlky, ktoré naznačujú prítomnosť bota v sieti. Boti sa pripájajú na C&C server pomocou DNS, keďže poznajú iba jeho doménové meno. Aby boli ťažšie detekovateľné, tieto doménové mená sú často náhodne generované. Pri analýze DNS požiadaviek je pozornosť upriamená na skladbu doménových mien.

Bilge a spol. [24] navrhli systém s názvom **EXPOSURE**, ktorý používa pasívne metódy analýzy DNS na detekciu domén, ktoré sa podieľajú na škodlivých aktivitách. Používali 15 funkcií, ktoré im umožnili charakterizovať rôzne vlastnosti domén a spôsobov, akými sú dopytované. Sledovali pomer čísel k dĺžke doménového mena. Rovnako analyzovali pomer dĺžky najdlhšieho podreťazca, ktorý má význam (nachádza sa v slovníku) k dĺžke domény.

Doyle [25] predstavil vo svojej práci metódu na detekciu pseudonáhodných doménových mien pomocou frekvenčnej analýzy. Keďže frekvencie znakov v doménových menách sú iné od frekvencií v anglických textoch, vytvoril frekvenčnú tabuľku znakov získanú z analýzy legitímnych doménových mien. Túto porovnával s tabuľkou s frekvenčnými hodnotami znakov v škodlivých doménových menách. Konkrétne sa jednalo o doménové mená generované botnetom Conficker. Všetky znaky v takýchto doménach mali približne rovnakú frekvenciu, teda sa líšili od referenčných hodnôt. Ďalej navrhol analýzu na základe váženého skóre. Pre každé doménové meno bola vypočítaná hodnota na základe referenčných frekvencií. Ak táto hodnota bola nižšia ako nejaký prah, doména bola označená ako škodlivá. Táto metóda bola efektívna pri detekcii viac ako 80% škodlivých domén, avšak detegovala aj určité percento falošných pozitív.

Systém **Pleiades [12]** sa pri zoskupovaní sústredil aj na skladbu doménových mien. Autori zvažovali frekvenciu n-gramov, entropiu alebo štruktúru domény (napríklad jej dĺžku, level domény, podobnú distribúciu frekvencie znakov a pod.). Vychádzali z faktu, že hostitelia, ktorí sú infikovaní rovnakým DGA-based malvérom majú tendenciu generovať čiastočne sa prekrývajúce množiny doménových mien.

Yadav a spol. [15] sa zaoberali analýzou doménového mena s využitím pravdepodobnostných, množinových a iných matematických metód. Predstavili detekciu DGA generovaných domén na základe Kullback-Leibler divergencie unigramov a bigramov. Následne určovali podobnosť medzi bigramami využitím Jaccardovho indexu. Podobnosť domén porovnávali pomocou Edit distance metódy.

3.3 Opakujúce sa vzorce/Iné vlastnosti

Pri analýze mnohých požiadaviek na konkrétne domény v časovom intervale sa môžu objaviť vzory označujúce škodlivú aktivitu. Sledované môžu byť denné podobnosti alebo opakujúce sa vzorce v DNS prevádzke.

Krmíček [26] sa sústredil na fakt, že používatelia posielajú požiadavky na lokálny DNS server. Z tohto dôvodu je podozrivé, ak viacero požiadaviek z jedného zariadenia smeruje na iný server. Takéto správanie môže indikovať infikovaného hostiteľa. Krmíček súčasne analyzoval aj časové okno DNS požiadaviek. Je možné že C&C server vydal informáciu o jeho presune, ak nejaká skupina hostiteľov pošle skupinu DNS požiadaviek v krátkom časovom intervale. Dáta získaval pomocou technológie NetFlow. V tejto práci poukázal na nedostatok dôležitých informácií obsiahnutých v údajoch získaných touto technológiou.

Časové vlastnosti požiadaviek a odpovedí boli taktiež analyzované v systéme **EXPOSURE [24].** Všímali si životnosť domény a sledovali opakujúce sa vzory v dennej DNS prevádzke. Skúmali zmeny počtu žiadostí o doménu. Jedným z ich poznatkov je, že škodlivé domény často vykazujú náhly nárast, po ktorom nasleduje náhly pokles počtu žiadostí.

Detekcia botnetov na základe skupinovej aktivity bola navrhnutá aj v práci, ktorú predstavili **Choi a Lee [27]**. Navrhli metódu na detekciu botnetov na základe ich základných charakteristík, čiže skupinovej aktivity. Navrhovaný systém, nazývaný **BotGAD**, dokáže v reálnom čase detegovať botnety z rozsiahlej siete, aj keď využívajú šifrovanú komunikáciu. Využili skutočnosť, že botnety zvyčajne generujú periodickú aktivitu, pričom legitímna prevádzka je náhodná.

Li a spol. [28] sa zameriavali na opakujúce sa správanie pri príchode paketov, s cieľom hierarchickej charakterizácie príchodov paketov, detekčných metód a kvantitatívnych metrík. Za týmto účelom prezentovali štruktúrovanú charakterizáciu

príchodov paketov, ktorá odráža časovú štruktúru opakujúceho sa správania v rôznych mierkach.

	TTL	NXDOMAIN	IP adresy v odpovedi	DNS server	skladba doménového mena	časové vlastnosti
R. Villamarin-Salomon, J.C. Brustoloni	X	X				
E. Stalmans, B. Irwin	X		X		X	
M. Antonakakis a spol.		X			X	
A. Schonewille, D. Van Helmond		X				
L. Bilge a spol.	X				X	X
R. Doyle					X	
S. Yadav a spol.					X	
V. Krmíček				X		X
H. Choi, H. Lee						X
J. Li a spol.						X

Tab. 1 Porovnanie riešení

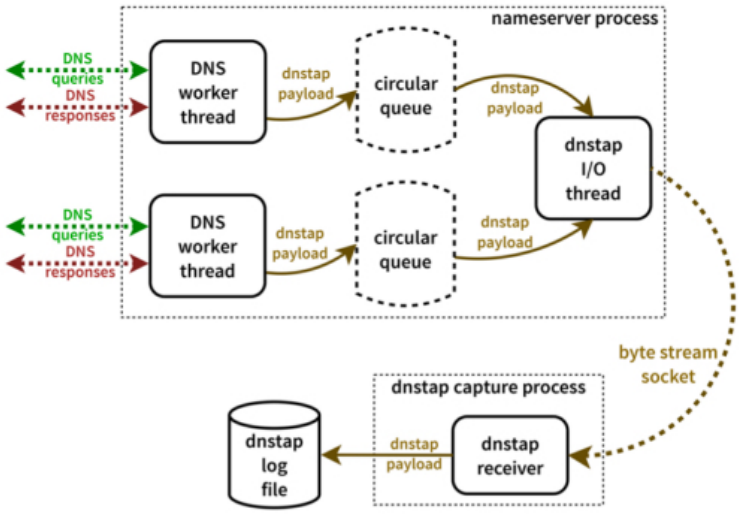
Na základe vyhodnotenia dostupných údajov a existujúcich metód sme sa rozhodli sústrediť na metódy najúčinnejšie na detekciu DGA botnetov. Z toho dôvodu sme využili hlavne metódy na analýzu doménových mien. Riešenie, ktoré predstavil Doyle [25], vykazovalo vysokú mieru úspešnosti. Priklonili sme sa k využitiu jeho metódy frekvenčnej analýzy a výpočtu váženého skóre domény. Predstavené riešenia ukázali, že identifikácia botnetov na základe NXDOMAIN odpovedí je užitočná. V rámci práce budeme sledovať a analyzovať aj tieto aspekty.

4 Systém na detekciu malvéru

V tejto práci sledujeme a analyzujeme anomálie DNS prevádzky pomocou údajov zo Študentských domov a jedální UPJŠ v Košiciach. Tieto údaje pochádzajú z rekurzívneho DNS servera. V tejto kapitole predstavíme ako vyzerajú vstupné údaje a uvedieme návrh systému. Súčasťou kapitoly je aj popis využívaných metód v navrhnutom systéme na detekciu malvéru.

4.1 Záznamy a predpríprava údajov

Na zber záznamov používame **DNSTAP** [29]. Je to rýchla a flexibilná metóda na zaznamenávanie a logovanie DNS prevádzky. Systém DNSTAP je súborový formát a súčasne aj program, ktorý vytvára súbory v danom formáte. Replikuje DNS záznamy na serveri posunie ich ďalej na spracovanie. Výstupom je bninárny súbor údajov. Schému priebehu môžeme vidieť na obrázku č.7.



Obr. 7 Schéma DNSTAP [29]

Na dekódovanie binárnych údajov sa využíva program **dnstap-read**. Výstupom z neho sú dva typy súborov. Prvý z nich je základný, ktorý obsahuje časovú pečiatku, druh záznamu, IP adresu dopytujúceho, protokol a doménové meno. Druhý ponúka rozšírené údaje o DNS zázname. Rozšírený záznam je zobrazený na obrázku č.8.

```
type: MESSAGE
identity: "sdaj_dns2"
version: "BIND 9.11.2-P1"
message:
  type: CLIENT_QUERY
  query_time: !!timestamp 2018-03-05 15:02:09.239964
  socket_family: INET
  socket_protocol: UDP
  query_address: 10.0.0.160
  response_address: 158.197.196.2
  query_port: 59810
  response_port: 53
  query_message: |
    ;; ->>HEADER<<- opcode: QUERY, rcode: NOERROR, id: 20071
    ;; flags: rd ; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 0

    ;; QUESTION SECTION:
    ;gmail.com.  IN    A

    ;; ANSWER SECTION:

    ;; AUTHORITY SECTION:

    ;; ADDITIONAL SECTION:

---
```

Obr. 8 Rozšírený DNSTAP záznam

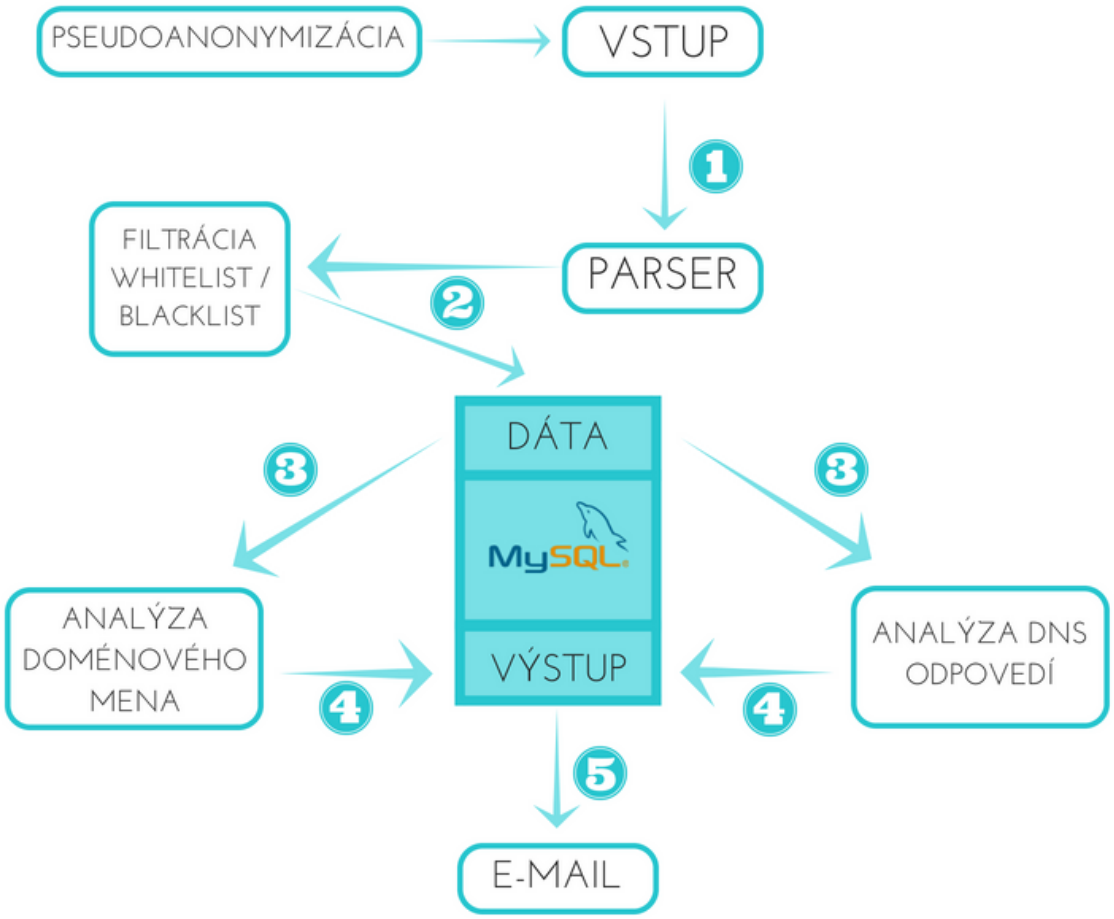
Táto ukážka zobrazuje klientsku DNS požiadavku na doménu gmail.com. V takomto formáte záznamu sa nachádza časová pečiatka, protokol, IP adresa dopytovateľa, IP adresa DNS servera, porty požiadavky a odpovede. V tele správy sa nachádza príznak požiadavky, návratový kód, identifikátor požiadavky a rôzne iné príznaky. Keďže je tento záznam DNS požiadavka, je vyplnená iba sekcia otázky, kde sa nachádza dopytované doménové meno. Záznam odpovede obsahuje v sekcii odpovede dopytované doménové meno a príslušnú IP adresu. Dôležitým aspektom pri predspracovaní údajov zohráva otázka **ochrany súkromia a osobných údajov**. Platný právny rámec podľa Nariadenie Európskeho parlamentu a Rady (EÚ) 2016/679 o ochrane fyzických osôb pri spracúvaní osobných údajov a o voľnom pohybe takýchto údajov, ktorým sa zrušuje smernica 95/46/ES (všeobecné nariadenie o ochrane údajov, GDPR) mení spracovanie osobných údajov v niektorých smeroch. Z nášho pohľadu dôležitou otázkou v tomto smere predstavuje spracovanie IP adries a doménových mien. Odpoveď na túto otázku nie je jednoduchá a závisí od rôznych okolností. V našom prípade ide ale spracúvanie údajov, ku ktorým univerzita vie doplniť ďalšiu údaje, pomocou ktorých by bolo možné identifikovať zariadenie a osobu, ktorej toto zariadenie patrí. Z tohto pohľadu IP adresy a doménové mená priradené zariadeniam v rámci

počítačovej siete predstavujú osobné údaje. Naopak, IP adresy a doménové mená mimo organizácie nebudú pre nás predstavovať osobné údaje, keďže k nim nie sme schopní priradiť ďalšie údaje, pomocou ktorých by fyzická osoba bola identifikovaná alebo identifikovateľná.

Keďže v našom prípade ide o spracovanie osobných údajov, je nutné tieto údaje anonymizovať alebo pseudoanonymizovať. **Anonymizáciu** môžeme charakterizovať ako proces, v ktorom sa z osobných údajov stanú údaje, pomocou ktorých nie je fyzická osoba identifikovaná alebo identifikovateľná (napr. pomocou IP adresy, doménového mena). Naopak **pseudoanonymizácia** predstavuje spracúvanie osobných údajov takým spôsobom, aby osobné údaje už nebolo možné priradiť konkrétnej dotknutej osobe bez použitia dodatočných informácií, pokiaľ sa takéto dodatočné informácie uchovávajú oddelene a vzťahujú sa na ne technické a organizačné opatrenia s cieľom zabezpečiť, aby osobné údaje neboli priradené identifikovanej alebo identifikovateľnej fyzickej osobe [30]. Z praktického pohľadu je vhodné ísť cestou pseudoanonymizácie, keďže po detekcii bota, administrátor potrebuje identifikovať infikované zariadenie a vykonať opatrenia na odstránenie tejto bezpečnostnej hrozby. V rámci práce pri predspracovaní údajov pseudoanonymizujeme IP adresy zariadení, ktoré predstavujú koncové zariadenie využívané študentom alebo zamestnancom univerzity.

4.2 Návrh riešenia

Navrhnutý systém je implementovaný v jazyku Python 3 a využíva MySQL databázu. Skladá sa zo štyroch hlavných častí, ktoré sú popísané v nasledujúcich podkapitolách. Na obrázku č.9. môžeme vidieť diagram popisovaného systému.



Obr. 9 Data-flow diagram systému

Na vstup systému prichádzajú údaje v DNSSEC rozšírenom formáte. Tieto údaje obsahujú záznamy typu CLIENT_QUERY a RESPONSE, RESOLVER_QUERY a RESPONSE. Tento systém analyzuje klientsku časť, preto sa v prvom kroku odfiltrujú RESOLVER záznamy. Z vyfiltrovaných údajov sa následne vyberajú atribúty jednotlivých záznamov. Sledujeme tieto údaje:

- časovú pečiatku,
- IP adresu odkiaľ prišila požiadavka,
- doménové meno,
- návratový kód,
- id záznamu a
- ak existuje, IP adresu z odpovede na požiadavku.

Druhým krokom je filtrácia. Ak sa pri kontrole záznamov pomocou **blacklistu** nájde škodlivé doménové meno, vloží sa do výstupnej databázy. Filtrácia pomocou **whitelistu** odfiltruje záznamy, ktoré sú legitímne a s ktorými nebudeme ďalej pracovať. Tie zvyšné sa uložia do databázy na ďalšie spracovanie.

V treťom kroku prebehnú nad údajmi v databáze dva druhy analýzy – **analýza skladby doménového mena** a **analýza DNS odpovedí**. Tieto metódy paralelne analyzujú vstupné údaje. Ich výstupom je databáza s potenciálne škodlivými záznamami. V poslednom, piatom kroku sa s určitou periodickosťou odosiela e-mail správcovi siete, ktorý obsahuje anomálne údaje z výstupnej databázy.

4.3 Filtrácia vstupných údajov

Keďže DNS prevádzka je veľmi veľká, potrebujeme nejakú prvotnú filtráciu. Na tento účel sme sa rozhodli využiť dva druhy filtrácie údajov – porovnávanie so zoznamom legitímnych doménových mien a porovnávanie so zoznamom škodlivých domén.

4.3.1 Whitelist

Možnosťou ako na začiatku vyradiť z analýzy veľké množstvo záznamov je porovnať ich doménové mená so zoznamom legitímnych doménových mien, tzv. „Whitelist". Takýto zoznam je napríklad **Alexa's top 1 000 000** [31], čiže evidencia milión najčastejšie navštevovaných stránok na webe. Je jedným z najrozšírenejších, preto je najlepšie porovnávať zachytené DNS záznamy práve s ním. Tým sa vyfiltruje obrovské množstvo záznamov s doménovými menami, ktoré by inak museli byť analyzované. Zoznam **Alexa** bol vytvorený na základe údajov o premávke poskytnutých používateľmi v globálnom paneli údajov spoločnosti Alexa počas trojmesačného intervalu. Poradie stránky je založené na počte jedinečných používateľov Alexy, ktorí navštívia stránku v daný deň. Rovnako sa berie do úvahy aj celkový počet žiadostí o URL adresu. Viac žiadostí o rovnakú doménu v ten istý deň od toho istého používateľa sa však počíta ako jedno zobrazenie stránky. Stránka s najvyššou kombináciou jedinečných návštevníkov a zobrazenia stránok sa nachádza na prvom mieste v zozname. Tieto zoznamy obsahujú iba top-level domény (napríklad domain.com) [32].

Poradie	Doména	Poradie	Doména
1	google.com	11	amazon.com
2	youtube.com	12	twitter.com
3	facebook.com	13	tmall.com
4	baidu.com	14	google.co.jp
5	wikipedia.org	15	live.com
6	yahoo.com	16	vk.com
7	google.co.in	17	sohu.com
8	reddit.com	18	instagram.com
9	qq.com	19	sina.com.cn
10	taobao.com	20	jd.com

Tab. 2 Alexa top 20 domén

Okrem toho, že Whitelist dokáže odfiltrovať najpoužívanejšie webstránky, filtruje aj domény generované operačným systémom. Výhodou je, že obsahuje aj domény využívané systémami na ochranu koncových zariadení (napr. antivírusovými programami), pretože tie využívajú DGA na generovanie vlastných domén tretieho rádu, ktoré by inak vykazovali anomálie v našom systéme. Napríklad Eset antivírus generuje náhodné znaky v treťom stupni domény e5.sk.

4.3.2 Blacklist

Jedným z prístupov, ako okamžite zistiť, či je doménové meno škodlivé, je porovnať ho s tzv. „Blacklistom“, teda zoznamom škodlivých doménových mien. V rámci práce sme využili zoznam **DNS Blackhole** [33], ktorý je dostupný online. Každý záznam obsahuje doménové meno a typ hrozby, ktorú predstavuje (napr. phishing, botnet, ransomvér, a podobne). Tento zoznam je udržiavaný a pravidelne aktualizovaný každé 2-3 dni. Vybraný blacklist združuje záznamy z rôznych iných stránok, ktoré sa zaoberajú analýzou a detegovaním škodlivých domén.

V tabuľke č.3 vidíme ukážku záznamov blacklistu. V prvom stĺpci sa nachádza škodlivá doména. V druhom stĺpci je vyznačené, ako sa táto doména využíva na škodlivé účely. Nachádzajú sa tam phishingové domény, domény generované botnetmi (napr. botnetom Necurs) alebo domény generované tróskym koňom s názvom Suppobox. Súčasne v ňom nájdeme aj tzv. „attack page“ ktoré slúžia napríklad na distribúciu malvéru. Následne je tam uvedený zdroj, teda označenie organizácie, ktorá detegovala škodlivú doménu (napr. spamhaus.org) a dátum objavenia domény.

Doména	Druh	Zdroj	Dátum
amazon.co.uk.security-check.ga	phishing	openphish.com	20171117
autosegurancabrasil.com	phishing	openphish.com	20171117
ksdiy.com	attackpage	safebrowsing.clients.google.com	20171117
vliiflsilgr.com	botnet	spamhaus.org	20171025
webhotell.enivest.no	malware	spamhaus.org	20171025
xyxbhig.net	botnet	spamhaus.org	20171025
alongarms.net	suppobox	private	20171025
alreadypartial.net	suppobox	private	20171025
cochvwr.net	necurs	private	20171025
dmanistravel.com	suspicious	spamhaus.org	20180417
kelderman-evers.nl	malware	spamhaus.org	20180417
kjkasdjaksdasdbe.com	malware	spamhaus.org	20180417
lastikpark.us	malware	spamhaus.org	20180417
perhapsstraight.net	botnet	spamhaus.org	20180417

Tab. 3 Blacklist domény

4.3.3 Spôsob vyhľadávania v zoznamoch

Whitelist a blacklist sú veľmi rozsiahle databázy. Základným spôsobom porovnávania reťazcov je porovnať každé doménové meno s každým záznamom v databáze. Avšak tento spôsob nie je efektívny a preto boli navrhnuté iné prístupy. Ich porovnanie môžeme vidieť v tabuľke č.4.

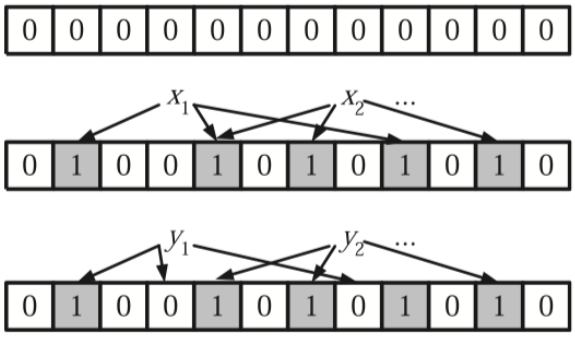
	Každý s každým	Písmenkový strom	Bloom filter	*n* - počet záznamov v databáze
Časová zložitosť	O(*n*)	O(*L*)	O(*k*)	*L* - dĺžka reťazca
Pamäťová zložitosť	O(*n*L*)	O(*dĺžka abecedy * L * n*)	O(*m*)	*k* - počet hashovacích funkcií

Tab. 4 Porovnanie algoritmov

Jednou z možností porovnávania reťazcov je **písmenkový strom**. Je to dátová štruktúra, ktorá sa využíva na uchovávanie množiny alebo asociatívneho poľa, kde kľúčmi sú zväčša reťazce. Je to strom, kde každý vrchol predstavuje jediné písmeno abecedy. Koreň tohto stromu predstavuje prázdny reťazec. Vrcholy, ktoré sú vo vzdialenosti *k* od koreňa predstavujú predponu dĺžky *k*. Vkladanie reťazca a jeho vyhľadávanie má časovú zložitosť O(*L*), kde *L* je dĺžka reťazca. Využitie klasickej

implementácie písmenkového stromu v našom systéme nie je možné. Je to z toho dôvodu, že písmenkové stromy hľadajú prefixy a preto nemôžeme využiť vyhľadávanie jednotlivých stupňov domény v strome. Na korektné porovnávanie by sme museli využiť knižnicu publicsuffixlist, ktorá dokáže z domény vyčleniť len druhý stupeň. Avšak využitie tejto knižnice nie je časovo efektívne.

Preto sme sa rozhodli využiť **Bloomov filter**. Je to pravdepodobnostná dátová štruktúra. Využíva sa na overenie, či prvok patrí do množiny alebo nie. Aj keď existuje nízka pravdepodobnosť chyby, efektivita tohto algoritmu prevažuje túto nevýhodu [34].



Obr. 10 Ukážka bloomovho filtra [34]

Na obrázku č.10. môžeme vidieť ukážku Bloomovho filtra.. Bloomov filter pre reprezentáciu množiny $\{x_1,x_2, \ldots ,x_n\}$ obsahujúcu n prvkov je pole m bitov. Na začiatku sú nastavené na 0. Ak chceme pridať prvok x_i do množiny, vypočítame hodnotu každej z k nezávislých hashovacích funkcií $h_1,..., h_k$ pre prvok x_i. Každý hash poskytuje bitové miesto, ktoré sa nastaví na 1. Ak chceme skontrolovať, či je v množine prvok y, hashujeme ho rovnako ako prvky x_i. Ak je na jednom z bitov 0, prvok sa nenachádza v množine [34]. Algoritmus vyhľadávania je na znázornený obrázku č.11.

```
Data: x is the object key to insert into the Bloom filter.
Function: insert(x)
for j : 1...k do
    /* Loop all hash functions k        */
    i ← h_j(x);
    if B_i == 0 then
        /* Bloom filter had zero bit at
        position i                       */
        B_i ← 1;
    end
end
```

Obr. 11 Vyhľadávanie v bloomovom filtri [35]

4.4 Analýza skladby doménového mena

Zraniteľnosťou DGA je vytvorenie doménového mena, ktoré sa vytvára pomocou nejakého algoritmu. Preto sa botnety využívajúce túto techniku dajú detegovať aj analýzou skladby doménového mena. Napríklad pri botnete s označením Conficker sa pomocou frekvenčnej analýzy ukázalo, že všetky písmená abecedy sú rovnomerne využívané. Avšak pri prirodzenom jazyku a taktiež aj pri legitímnych doménových menách sa niektoré písmená vyskytujú oveľa častejšie ako iné [6]. V rámci nášho riešenia budeme využívať niekoľko prístupov, ktorým sa podrobnejšie venujeme v nasledujúcich podkapitolách.

4.4.1 Shannonova entropia

Prvým prístupom je využitie Shannonovej entropie [36]. Entropia v našom prípade určuje mieru náhodnosti doménového mena. Keďže DGA domény sú často veľmi náhodné, tak hodnota entropie v danom doménovom mene bude vysoká. Entropiu vypočítame vzorcom (1), kde p_i je pravdepodobnosť i-teho znaku vyskytujúceho sa v reťazci.

$$H = -\sum_{i} p_i \log_2 p_i \tag{1}$$

Podľa výsledkov výpočtu entropie pre každé doménové meno zo vstupných údajov sa optimalizuje prah, ktorý by výsledok nemal prekročiť, ak sa jedná o legitímne doménové meno. Rozhodli sme sa využiť hodnotu, ktorá vychádza z dvoch výsledkov – priemernej entropie whitelistu a priemernej entropie reálnych DGA domén z datasetu. Ich priemerom je číslo 3,761, ktoré bolo nastavené ako prahová hodnota.

4.4.2 Analýza pomocou frekvencie znakov

Druhým prístupom je využitie analýzy pomocou frekvencie znakov. Ryan Doyle ukázal v článku [25], že táto metóda sa osvedčila v detegovaní pseudonáhodných domén. Vytvoril jednoduchý vzorec (2) na výpočet váhy doménového mena pomocou váženej frekvencie znakov, kde x_i je frekvencia znaku i a n je počet znakov v doméne. Výsledok je vynásobený číslom 1000 na uľahčenie práce s váhami. Ak takáto váha je nižšia ako nejaká daná hodnota, tak sa tam častejšie vyskytujú znaky, frekvencia

ktorých je nízka. Rovnako sme sa rozhodli zvoliť si prah hodnotu 46,994, pretože tá vykazovala v tejto metóde najlepšie výsledky.

$$w = \frac{\sum_{i=0}^{n} x_i}{n} \times 1000 \quad (2)$$

Frekvencie znakov v anglickom jazyku nezodpovedajú frekvenciám znakov v doménových menách. Ich porovnanie môžeme vidieť na obrázku č.12. V tomto prípade budeme využívať frekvencie, ktoré boli využité v [25] s kombináciou frekvencií, ktoré sme vypočítali pre čísla a iné znaky zo zoznamu Alexa.

	Ref.	Sample		Ref.	Sample
a	0.08167	0.08673	***n***	0.06749	0.06246
b	0.01492	0.02242	***o***	0.07507	0.07256
c	0.02782	0.04300	***p***	0.01929	0.02890
d	0.04253	0.03547	***q***	0.00095	0.00203
e	0.12702	0.10543	***r***	0.05987	0.06705
f	0.02228	0.01595	***s***	0.06327	0.07062
g	0.02015	0.02896	***t***	0.09056	0.06821
h	0.06094	0.02371	***u***	0.02758	0.02965
i	0.06966	0.07400	***v***	0.00978	0.01376
j	0.00153	0.00344	***w***	0.02360	0.01667
k	0.00772	0.01473	***x***	0.00150	0.00588
l	0.04025	0.04898	***y***	0.01974	0.01867
m	0.02406	0.03569	***z***	0.00074	0.00504

Obr. 12 Frekvencie znakov v anglickom jazyku a doménových menách [25]

4.4.3 Iné možnosti analýzy

V doménových menách sa dajú pozorovať aj iné vlastnosti. Príkladom môže byť analýza počtu čísel alebo pomlčiek a pomer týchto hodnôt k dĺžke domény. Ak sa tieto hodnoty odchyľujú od vlastností zvyčajných doménových mien, je takáto doména podozrivá. Rovnako sa dá sledovať aj samotná dĺžka domény, frekvencia n-gramov, podobnosť medzi bigramami v doménových menách alebo podobnosť domén v prevádzke [15]. V budúcnosti je možné sa venovať metódam založeným na týchto pozorovaniach.

4.5 Analýza DNS odpovedí

Vďaka DGA algoritmu sa generujú obrovské množstvá doménových mien. Z nich je reálne registrovaná iba veľmi malá podmnožina, ktorá sa využíva ako skutočné domény C&C serverov. Bot posiela požiadavky na veľké množstvo neexistujúcich (neregistrovaných) domén počas krátkeho časového intervalu. Z tohto dôvodu sa k hostiteľovi dostane väčšie množstvo DNS odpovedí (NXDOMAIN). Túto skutočnosť je možné využiť v rámci detekcie botnetov a sledovať takéto anomálie v DNS prevázke.

Po zoradení DNS záznamov podľa IP adresy a časovej pečiatky sledujeme CLIENT_RESPONSE záznamy. Sledujeme množstvo návratových kódov NXDOMAIN na jedného hostiteľa počas krátkeho časového okna.

Časové okná požiadaviek sa líšia u rôznych typov botnetov. Napríklad boty Conficker posielajú požiadavky na 20 doménových mien každých 10 sekúnd [37]. Iné boty posielajú požiadavky oveľa pomalšie (jedna požiadavka každých 5 minút). Aby sme sa vyhli strate výsledkov, rozhodli sme sa sledovať časové okno 10 minút medzi jednotlivými NXDOMAIN odpoveďami na jednu IP adresu. Aj keď botnety generujú tisícky domén za pár hodín, bot môže mať šťastie a získať reálnu doménu po malom počte požiadaviek. V zhľadom na vyššie uvedené budeme považovať požiadavky za podozrivé, ak sa v zvolených časových intervaloch vyskytne minimálne 10 takýchto odpovedí.

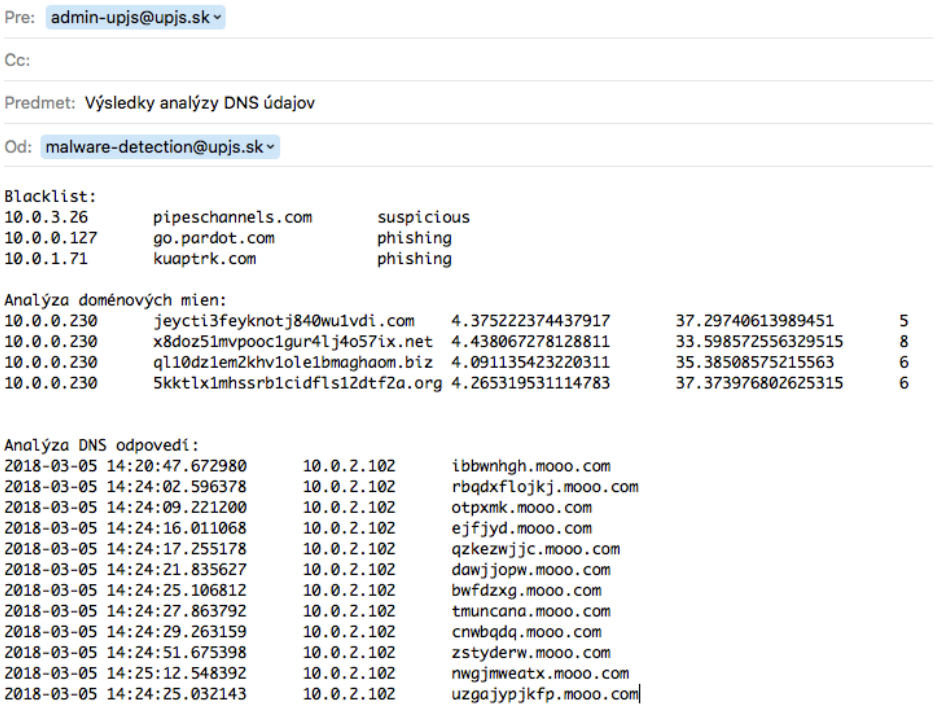
Time	Source	Port	Destination	Port	Info
17:49:38	192.168.204.150	52933	192.168.204.2	53	Standard query 0x5f43 A uknw9n17fo131nhr8rtxgbx9y.com
17:49:38	192.168.204.2	53	192.168.204.150	52933	Standard query response 0x5f43 No such name
17:49:39	192.168.204.150	58865	192.168.204.2	53	Standard query 0xa255 A 15psrx44hsh8zrry0aet3xnha.net
17:49:39	192.168.204.2	53	192.168.204.150	58865	Standard query response 0xa255 No such name
17:49:40	192.168.204.150	54468	192.168.204.2	53	Standard query 0x7ca0 A 9rg38gfs0t7d1mb10cu1a29aw9.org
17:49:40	192.168.204.2	53	192.168.204.150	54468	Standard query response 0x7ca0 No such name
17:49:41	192.168.204.150	49662	192.168.204.2	53	Standard query 0x64ad A 3pd31izm8jeed8zbe11vjkw7j.net
17:49:41	192.168.204.2	53	192.168.204.150	49662	Standard query response 0x64ad No such name
17:49:42	192.168.204.150	65479	192.168.204.2	53	Standard query 0x912a A 8u4bwl1o2d1gp6mhhmx7htbdy.com
17:49:42	192.168.204.2	53	192.168.204.150	65479	Standard query response 0x912a No such name
17:49:43	192.168.204.150	59399	192.168.204.2	53	Standard query 0xa3f2 A 1u9dii11mtm2ufrgwhjj17x45l7.org
17:49:43	192.168.204.2	53	192.168.204.150	59399	Standard query response 0xa3f2 No such name
17:49:44	192.168.204.150	52229	192.168.204.2	53	Standard query 0x64bc A 13mxszjxoit2d13c6iip1wljt5g.biz
17:49:44	192.168.204.2	53	192.168.204.150	52229	Standard query response 0x64bc No such name
17:49:45	192.168.204.150	61776	192.168.204.2	53	Standard query 0x382b A 1eud1wpbik59r1waf5k21qkruu4.net
17:49:45	192.168.204.2	53	192.168.204.150	61776	Standard query response 0x382b No such name
17:49:47	192.168.204.150	63096	192.168.204.2	53	Standard query 0x9007 A ogm2xj1nlnxzjgzsjvl12wp5gw.com
17:49:47	192.168.204.2	53	192.168.204.150	63096	Standard query response 0x9007 No such name

Obr. 13 Ukážka prevádzky infikovaného bota Zeus [38]

4.6 Informačný modul

Poslednou a súčasne dôležitou súčasťou systému je modul na oboznamovanie administrátora o výskyte zariadenia v počítačovej sieti, ktoré s istotou (výskyt na blackliste) alebo s určitou pravdepodobnosťou (prekročenie hodnôt stanovených v rámci analýzy skladby doménového mena) je infikované a súčasťou botnetu. Úlohou tohto modulu je poslať emailovú správu administrátorovi.

Odosielaný email obsahuje dáta z troch výstupných tabuliek obsiahnutých v databáze. Prvá je výstupná tabuľka pre údaje z blacklistu, ktorá obsahuje jednoznačne škodlivé domény a napadnutých hostiteľov. Druhá je výstupná tabuľka analýzy doménových mien, ktorá obsahuje anomalické dáta, čiže domény, ktorých hodnoty entropie a váženého skóre na základe frekvencie znakov prekročili dané prahy. Treťou tabuľkou je výstup analýzy DNS odpovedí, ktorá obsahuje IP adresu potenciálne napadnutého hostiteľa, časovú pečiatku a doménové mená. Na obrázku č.14 vidíte vzor takejto emailovej správy.



Pre: admin-upjs@upjs.sk

Cc:

Predmet: Výsledky analýzy DNS údajov

Od: malware-detection@upjs.sk

```
Blacklist:
10.0.3.26       pipeschannels.com       suspicious
10.0.0.127      go.pardot.com           phishing
10.0.1.71       kuaptrk.com             phishing

Analýza doménových mien:
10.0.0.230      jeycti3feyknotj840wu1vdi.com    4.375222374437917       37.29740613989451       5
10.0.0.230      x8doz51mvpooc1gur4lj4o57ix.net  4.438067278128811       33.598572556329515      8
10.0.0.230      ql10dz1em2khv1ole1bmaghaom.biz  4.091135423220311       35.38508575215563       6
10.0.0.230      5kktlx1mhssrb1cidfls12dtf2a.org 4.265319531114783       37.373976802625315      6


Analýza DNS odpovedí:
2018-03-05 14:20:47.672980      10.0.2.102      ibbwnhgh.mooo.com
2018-03-05 14:24:02.596378      10.0.2.102      rbqdxflojkj.mooo.com
2018-03-05 14:24:09.221200      10.0.2.102      otpxmk.mooo.com
2018-03-05 14:24:16.011068      10.0.2.102      ejfjyd.mooo.com
2018-03-05 14:24:17.255178      10.0.2.102      qzkezwjjc.mooo.com
2018-03-05 14:24:21.835627      10.0.2.102      dawjjopw.mooo.com
2018-03-05 14:24:25.106812      10.0.2.102      bwfdzxg.mooo.com
2018-03-05 14:24:27.863792      10.0.2.102      tmuncana.mooo.com
2018-03-05 14:24:29.263159      10.0.2.102      cnwbqdq.mooo.com
2018-03-05 14:24:51.675398      10.0.2.102      zstyderw.mooo.com
2018-03-05 14:25:12.548392      10.0.2.102      nwgjmweatx.mooo.com
2018-03-05 14:24:25.032143      10.0.2.102      uzgajypjkfp.mooo.com
```

Obr. 14 Emailová správa posielaná administrátorovi

5 Výsledky

Anonymizované záznamy z DNS prevádzky na Študentských domov a jedální UPJŠ sme podrobili analýze pomocou navrhnutého systému. Použitím Bloomovho filtra sa rapídne zvýšila efektivita filtrácie pomocou whitelistu a blacklistu, keďže výpočtová zložitosť porovnávania je *O(k),* kde *k* je počet hashovacích funkcií. Zvolený whitelist dokázal v našom datasete odfiltrovať až 92,79% záznamov. Po porovnaní domén so záznamami v blackliste sme v prevádzke objavili tri phishingové domény a jednu doménu označenú ako podozrivú, ktoré boli viackrát dopytované z rôznych hostiteľov. Môžeme ich vidieť v tabuľke č.5. Po analýze doménových mien a analýze DNS odpovedí sa avšak žiadne anomálie nenašli.

Doména	Druh
pipeschannels.com	suspicious
go.pardot.com	phishing
kuaptrk.com	phishing
ext-sq.squarespace.com	phishing

Tab. 5 Výsledky filtrácie cez blacklist

Analýzu doménových mien sme spustili nad datasetom doménovými menami, ktoré reálne dopytoval botnet s názvom **Gameover Zeus** [39]. Ukážku výsledkov môžeme vidieť v tabuľke č.6. V prvom stĺpci tabuľky sa nachádza analyzované doménové meno. V druhom je výsledok výpočtu Shannonovej entropie. Môžeme vidieť, že presahuje náš zvolený prah – hodnotu 3,761. V ďalšom stĺpci sa nachádza výpočet váhy na základe frekvencií znakov. Tieto hodnoty spadajú taktiež pod zvolenú prahovú hodnotu 46,994. Počty čísel v doméne sa rovnako vymykajú normálu.

Doménové meno	Entropia	Frekvencia	Čísla
1tnt2x21s7y3sv37jnqu1jtyaw2.biz	4.17923	27.07313	10
pldbim16n9wtt1cv1t3gbp85vj.com	4.32485	29.83525	8
drnhh61jugfqq16ki1tvgsd5h.net	4.18998	31.60745	6
d8sc2l1vv3gnq1jl37twkur2yf.biz	4.63161	25.65637	8
1kh5teop5r6ans5uyoa13yybgp.org	4.17146	38.14097	7
k5ehni1smlxapyepk1i13o26ns.com	4.28468	40.19971	7
qwx7ar1r6kcyg19p954512jfaed.com	4.53891	27.22827	11

Tab. 6 Výsledky analýzy doménového mena

Z výsledkov výpočtov môžeme vidieť, že systém na detekciu botnetov navrhnutý v tejto práci dokáže detegovať škodlivé domény pomocou blacklistu. Taktiež dokáže detegovať reálne existujúce botnety. Ako môžeme vidieť v tabuľke, hodnoty entropie a frekvencie sú v nami zvolenom intervale. To potvrdzuje, že hodnoty prahov v jednotlivých metódach boli správne navrhnuté a tento systém dokáže indikovať podozrivú doménu.

Záver

Keďže hrozba botnetov rapídne rastie, je potrebné sa týmto problémom zaoberať. Botnety sa využívajú na rôznu škálu škodlivých aktivít. Z tohto vyplýva nutnosť vyvíjať stále efektívnejšie riešenia na ich detekciu.

Prvým cieľom práce bolo analyzovať možnosti detekcie škodlivého kódu a botnetov. Na začiatku sme sa venovali fungovaniu systému doménových mien. To bolo dôležité pre pochopenie toho, ako ho botnety využívajú a ako zefektívniť ich detekciu. Súčasne sme predstavili samotné fungovanie botnetu a jeho spôsob komunikácie. Popísali sme DNS techniky, ktoré využíva na vyhnutie sa detekčným mechanizmom. Zistili sme, aké sieťové anomálie generujú botnety, čo nám pomohlo navrhnúť systém na ich detekciu. Porovnali sme rôzne typy mechanizmov, ktoré sa využívajú na detekciu a analýzu botnetov. Rozhodli sme sa pre využitie detekcie na základe pasívnej sieťovej prevádzky, konkrétne pre sledovanie DNS prevádzky.

Druhý cieľ práce je zameraný na porovnanie metód používaných pri detekcii bezpečnostných incidentov pomocou DNS údajov. Popísali sme už existujúce prístupy na detekciu botnetov, vďaka ktorým sme dokázali porovnať, aké metódy sú najefektívnejšie. Na základe tohto porovnania sme sa rozhodli využiť viacero prístupov. V práci využívame detekciu na základe analýzy doménového mena a na základe DNS odpovedí.

Tretím cieľom práce je návrh a implementácia systému na detekciu botnetov a škodlivých domén z DNS prevádzky. Dôležitosť tohto systému bola načrtnutá v úvodnej kapitole. V rámci návrhu systému popisujeme spracovanie a predprípravu údajov, ktoré sa analyzujú. Popísali sme systém DNSTAP, ktorý je pre našu infraštruktúru najvýhodnejší na zber DNS údajov. Tento systém neovplyvňuje činnosť siete, keďže nezaťažuje DNS server a neovplyvňuje jeho beh. Následne popisujeme metódy, ktoré sme si zvolili pre využitie v systéme. Prvým krokom je odfiltrovanie legitímnej DNS prevádzky pomocou whitelistu. Rozhodli sme sa pre využitie zoznamu Alexa, ktorý obsahuje milión najčastejšie využívaných legitímnych domén. Následne sme predstavili filtráciu pomocou blacklistu, ktorý jednoznačne určí, či je doména škodlivá. Keďže filtrácia pomocou whitelistu a blacklistu bola spočiatku neefektívna, museli sme sa rozhodnúť pre zefektívnenie týchto metód využitím Bloom filtra, ktorá bola tiež uvedená. Následne sme navrhli dva druhy analýz, ktoré sú efektívne

pri detekcii botnetov. Prvou z nich je analýza doménového mena pomocou Shannonovej entropie a frekvenčnej analýzy. Následne sme určili hodnoty, ktoré určujú, či je doména podozrivá. Druhou z nich je analýza DNS odpovedí, ktorá sa zameriava na časové okná domén s priznakom NXDOMAIN.

Napokon sme analyzovali DNS prevádzku zo Študentských domov a jedální UPJŠ pomocou navrhnutého systému. Avšak v tomto datasete sa nenašli žiadne anomálie, preto je nutné analyzovať ďalšiu prevádzku v počítačovej sieti vo väčšom časovom rozsahu. Na otestovanie správnosti analýzy doménového mena sme sa rozhodli previesť túto metódu nad reálnymi doménami generovanými botnetom s názvom Gameover Zeus. Tá nám potvrdila efektivitu danej metódy, keďže na danom datasete mala stopercentnú úspešnosť.

Systém, ktorý sme navrhli a implementovali, dokáže upozorniť správcu siete o anomáliách v prevádzke. To prispeje k bezpečnosti počítačovej siete na Univerzite Pavla Jozefa Šafárika. Pomocou vzniknutého systému sa dá skúmať vhodnosť zvolených metód. V budúcnosti je možné model systému rozširovať a môže byť základom pre komplexnejší systém na analýzu anomálií v počítačovej sieti.

Zoznam použitej literatúry

1. Google Domains Help. [online] Dostupné z: https://support.google.com/domains/answer/3251148?hl=en-GB.
2. Difference between Authoritative and Recursive DNS Nameservers. [online] Dostupné z: https://umbrella.cisco.com/blog/2014/07/16/difference-authoritative-recursive-dns-nameservers/.
3. AITCHISON, R., Pro Dns and BIND 10. 2011: Apress.
4. MOCKAPETRIS, P., RFC 1035—Domain names—implementation and specification, November 1987. [online] Dostupné z: http://www.ietf.org/rfc/rfc1035.txt, 2004.
5. THOMSON, S., et al., Rfc 3596: Dns extensions to support ip version 6. [online] Dostupné z: https://www.ietf.org/rfc/rfc3596.txt 2003.
6. ESLAHI, M., R. SALLEH, and N.B. ANUAR. Bots and botnets: An overview of characteristics, detection and challenges. IEEE.
7. Yin, C., et al., Botnet detection based on correlation of malicious behaviors. Int J Hybrid Inf Technol, 2013. **6**(6): p. 291-300.
8. FEILY, M., A. SHAHRESTANI, and S. RAMADASS. A survey of botnet and botnet detection. IEEE.
9. Wang, P., S. Sparks, and C.C. Zou, An advanced hybrid peer-to-peer botnet. IEEE Transactions on Dependable and Secure Computing, 2010. **7**(2): p. 113-127.
10. LIVINGOOD, J., N. MODY, and M. O'REIRDAN, Recommendations for the Remediation of Bots in ISP Networks (RFC 6561). Internet Eng. Task Force 2012.
11. STONE-GROSS, B., et al. Your botnet is my botnet: analysis of a botnet takeover. ACM.
12. ANTONAKAKIS, M., et al. From Throw-Away Traffic to Bots: Detecting the Rise of DGA-Based Malware.
13. PORRAS, P.A., H. SAÏDI, and V. YEGNESWARAN. A Foray into Conficker's Logic and Rendezvous Points.

14. PORRAS, P., H. SAÏDI, and V. YEGNESWARAN, Conficker C analysis. SRI International, 2009.

15. YADAV, S., et al. Detecting algorithmically generated malicious domain names. ACM.

16. Ostap, H. and R. Antkiewicz. A Concept of Clustering-Based Method for Botnet Detection. Springer.

17. Silva, S.S.C., et al., Botnets: A survey. Computer Networks, 2013. **57**(2): p. 378-403.

18. GU, G., et al. BotMiner: Clustering Analysis of Network Traffic for Protocol- and Structure-Independent Botnet Detection.

19. BINKLEY, J.R. and S. SINGH, An Algorithm for Anomaly-based Botnet Detection. SRUTI, 2006. **6**: p. 7-7.

20. MANASRAH, A.M., et al., Detecting botnet activities based on abnormal DNS traffic. arXiv preprint arXiv:0911.0487, 2009.

21. VILLAMARIN-SALOMON, R. and J.C. BRUSTOLONI. Identifying Botnets Using Anomaly Detection Techniques Applied to DNS Traffic. in 2008 5th IEEE Consumer Communications and Networking Conference. 2008.

22. STALMANS, E. and B. IRWIN. A framework for DNS based detection and mitigation of malware infections on a network. in 2011 Information Security for South Africa. 2011.

23. SCHONEWILLE, A. and D.-J. VAN HELMOND, The domain name service as an IDS. Research Project for the Master System-and Network Engineering at the University of Amsterdam, 2006.

24. BILGE, L., et al. EXPOSURE: Finding Malicious Domains Using Passive DNS Analysis.

25. DOYLE, R., Frequency analysis second level domains. 2010, [online] Dostupné z: http://ryandoyle.net.

26. KRMÍČEK, V., Inspecting DNS Flow Traffic for Purposes of Botnet Detection, in GEANT3 JRA2 T4 Internal Deliverable. 2011.

27. CHOI, H. and H. LEE, Identifying botnets by capturing group activities in DNS traffic. Computer Networks, 2012. **56**(1): p. 20-33.

28. LI, J., et al. Modeling repeating behaviors in packet arrivals: Detection and measurement. IEEE.

29. DNS query/response logging with dnstap. [online] Dostupné z: https://jpmens.net/2017/09/11/dns-query-response-logging-with-dnstap/.

30. REGULATION, General Data Protection. Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46. Official Journal of the European Union (OJ), 2016, 59: 1-88.

31. Alexa Top 1 000 000 Websites. [online] Dostupné z: http://www.seobook.com/download-alexa-top-1-000-000-websites-free.

32. Alexa Top Sites. [online] Dostupné z: https://aws.amazon.com/alexa-top-sites/.

33. DNS-BH – Malware Domain Blocklist by RiskAnalytics [online] Dostupné z: http://www.malwaredomains.com/

34. BRODER, A. and M. MITZENMACHER, Network applications of bloom filters: A survey. Internet mathematics, 2004. **1**(4): p. 485-509.

35. Tarkoma, S., C.E. Rothenberg, and E. Lagerspetz, Theory and practice of bloom filters for distributed systems. IEEE Communications Surveys & Tutorials, 2012. **14**(1): p. 131-155.

36. SHANNON, C.E., A mathematical theory of communication. Bell system technical journal, 1948. **27**(3): p. 379-423.

37. ZHANG, H., et al. Botdigger: Detecting dga bots in a single network.

38. Zeus-related DGA DNS requests and responses. [online] Dostupné z: https://www.malware-traffic-analysis.net/2014/09/01/index.html.

39. Gameover Zeus DGA sample 31000 DGA domains from Dec 2014 [online] Dostupné z: http://www.secrepo.com.

Prílohy

Príloha A: CD médium – bakalárska práca v elektronickej podobe, zdrojové kódy systému na detekciu malvéru

Príloha B: Filtrácia pomocou Whitelist a Blacklist

Príloha C: Vláknová analýza doménového mena

Príloha D: Vláknová analýza DNS odpovede

Príloha B: Filtrácia pomocou Whitelist a Blacklist

```
from pybloom import BloomFilter
import pymysql

f = BloomFilter(capacity=5000000, error_rate=0.00001)
db = pymysql.connect(host='localhost',
                     user='***', passwd='***', db="DNSData")
cursor = db.cursor()
cursor.execute("SELECT domain_name FROM Whitelist")
res = cursor.fetchall()
for r in res:
    f.add(r[0])
cursor.close()
# Whitelist comparision using Bloom filter
def check_whitelist(dn):
    domain_name = dn.split('.')
    pp = None
    for i in range(len(domain_name)):
        pp = ".".join((domain_name[-(i + 1)], pp) if pp is \
                 not None else (domain_name[-(i + 1)],))
        if pp in f:
            return True
    return False
fblack = BloomFilter(capacity=12000, error_rate=0.00001)
cursor_bl = db.cursor()
cursor_bl.execute("SELECT domain_name FROM Blacklist")
results_bl = cursor_bl.fetchall()
cursor_bl.close()
# Blacklist comparision using Bloom filter
def check_blacklist(dn):
    for r in results_bl:
        fblack.add(r[0])
    if dn in fblack:
        return True
    else:
        return False
```

Príloha C: Vláknová analýza doménového mena

```
import threading
from queue import *
import time
import pymysql
import math
from collections import Counter

BUF_SIZE = 10000
q = Queue(BUF_SIZE)


class ProducerThread(threading.Thread):
    def __init__(self, target=None, name=None):
        super(ProducerThread, self).__init__()
        self.target = target
        self.name = name

    def run(self):
        db = pymysql.connect(host='localhost',
                             user='***', passwd='***', db="DNSData")
        cursor = db.cursor()
        cursor.execute("SELECT DISTINCT domain_name,query_address
                        FROM Data3")
        while True:
            if not q.full():
                res = cursor.fetchone()
                if res is None:
                    cursor.close()
                    return
                q.put(res)
        cursor.close()
        return
class ConsumerThread(threading.Thread):
    def __init__(self, target=None, name=None):
        super(ConsumerThread, self).__init__()
        self.target = target
        self.name = name
```

```
        return
    def entropy(self,s):
        if s is not None:
            p, lns = Counter(s), float(len(s))
            return -sum(count / lns * math.log(count / lns, 2) \
                 for count in p.values())
        else:
            return None

    def frequency(self,dn):
        freq = eval(open("freq.txt").read())
        sum = 0
        for letter in dn:
            sum += freq[letter.lower()]
        w = (sum / len(dn)) * 1000
        return w

    def numbers(self,dn):
        sum = 0
        for letter in dn:
            if letter.isdigit():
                sum += 1
        return sum

    def dashes(self,dn):
        sum = 0
        for letter in dn:
            if letter == '-':
                sum += 1
        return sum

    def analyse(self,item):
        db = pymysql.connect(host='localhost',
                             user='***', passwd='***', db="DNSData")
        cursor = db.cursor()
        ent = self.entropy(item[0])
        freq = self.frequency(item[0])
        num = self.numbers(item[0])
```

```
        d = self.dashes(item[0])
        string = ' '
        if ent > 3.761 and freq < 46.994:
            cursor.execute("INSERT INTO DNResults( \
                 domain_name,entropy,frequency_an,numbers,dash,ip)\
                 VALUES (%s,%s,%s,%s,%s,%s)",
                 (item[0],ent,freq,num,d,item[1]))

            db.commit()
        cursor.close()

    def run(self):
        while True:
            if not q.empty():
                item = q.get()
                self.analyse(item)
            else: break
        return


if __name__ == '__main__':
    p = ProducerThread(name='producer')
    c = ConsumerThread(name='consumer')

    p.start()
    time.sleep(2)
    for i in range(4):
        c = ConsumerThread(name='consumer')
        c.start()
```

Príloha D: Vláknová analýza DNS odpovede

```
import threading
from datetime import timedelta
from datetime import datetime
from queue import *
import pymysql
import time

BUF_SIZE = 10000
q = Queue(BUF_SIZE)

class ProducerThread(threading.Thread):
    def __init__(self, target=None, name=None):
        super(ProducerThread, self).__init__()
        self.target = target
        self.name = name

    def run(self):
        db = pymysql.connect(host='localhost',
                             user='***', passwd='***', db="DNSData")
        cursor = db.cursor()
        cursor.execute('SELECT DISTINCT query_address FROM Data3 \
                        WHERE rcode = "NXDOMAIN"')
        while True:
            if not q.full():
                res = cursor.fetchone()
                if res is None:
                    cursor.close()
                    return
                q.put(res)

        cursor.close()
        return

class ConsumerThread(threading.Thread):
    def __init__(self, target=None, name=None):
        super(ConsumerThread, self).__init__()
```

```
        self.target = target
        self.name = name
        self.db = pymysql.connect(host='localhost',
                              user='***', passwd='***', db="DNSData")
        self.cursor = self.db.cursor()
        return

    def nxdomain(self, results):
        count = 1
        data = []
        i = 0
        for r in results:
            if (i < len(results)-1):
                pom = results[i + 1]
                t = r[1].replace("\n", "")
                time = datetime.strptime(t, '%Y-%m-%d %H:%M:%S.%f')
                t2 = pom[1].replace("\n", "")
                time2 = datetime.strptime(t2, '%Y-%m-%d %H:%M:%S.%f')

                if time2 < time + timedelta(minutes=10):
                    data.append(results[i])
                    count += 1
                else:
                    data.append(results[i])
                    if (count >= 10):
                        for d in data:
                            self.cursor.execute("INSERT INTO \
                         NXDomainResults(type_rq,time_of,query_address, \
                         rcode,id_q,domain_name) VALUES \
                         (%s,%s,%s,%s,%s,%s)", \
                         (d[0],d[1],d[2],d[3],d[4],d[5]))
                        self.db.commit()
                    count = 0
                    data = []
                i += 1
```

```
            else:
                if (count >= 10):
                    for d in data:
                        self.cursor.execute("INSERT INTO \
                         NXDomainResults(type_rq,time_of,query_address, \
                         rcode,id_q,domain_name) VALUES \
                         (%s,%s,%s,%s,%s,%s)", \
                         (d[0],d[1],d[2],d[3],d[4],d[5]))
                        self.db.commit()
                    return

    def run(self):
        db = pymysql.connect(host='localhost',
                             user='***', passwd='***', db="DNSData")
        cursor = db.cursor()
        while True:
            if not q.empty():
                item = q.get()
                cursor.execute("SELECT * FROM Data3 WHERE rcode =  \
                         'NXDOMAIN' and query_address like %s order by \
                         time_of asc",(item,))
                results = cursor.fetchall()
                self.nxdomain(results)
            else:
                 break
        cursor.close()
        return

if __name__ == '__main__':
    p = ProducerThread(name='producer')
    c = ConsumerThread(name='consumer')
    p.start()
    time.sleep(2)
    for i in range(4):
        c = ConsumerThread(name='consumer')
        c.start()
```