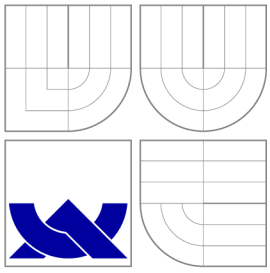
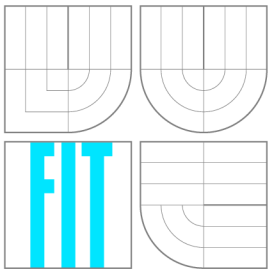


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



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

# DETEKCE SPAMU POMOCÍ DNS MX ZÁZNAMŮ

SPAM DETECTION USING DNS MX RECORDS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

ONDŘEJ PLOTĚNÝ

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. MICHAL KOVÁČIK

BRNO 2016

*Zadání bakalářské práce/18671/2015/xplote01*

**Vysoké učení technické v Brně - Fakulta informačních technologií**

Ústav počítačových systémů Akademický rok 2015/2016

# Zadání bakalářské práce

Řešitel: **Plotěný Ondřej**
Obor: Informační technologie
Téma: **Detekce spamu pomocí DNS MX záznamů**
**Spam Detection Using DNS MX Records**
Kategorie: Počítačové sítě

Pokyny:

1. Seznamte se se službou a protokolem DNS, nastudujte problematiku spamu na Internetu.
2. Seznamte se s formáty NetFlow, IPFIX a libpcap a zvolte vhodný typ vstupních dat z pohledu efektivity detekce.
3. Navrhněte metodu detekce spamu s použitím DNS dat. Vyjděte z již existujících metod.
4. Implementujte navrženou metodu detekce.
5. Ověřte implementovanou metodu na testovacích datech a na datech z reálného provozu.
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

Předmětem této práce je detekce stanic v síti rozesílající nevyžádanou poštu pomocí pasivní analýzy zachyceného DNS provozu. Představuje návrh a implementaci systému, který realizuje detekci DNS anomálií na základě vysokého počtu MX dotazů a poměru obdržených NXDomain odpovědí. Systém byl testován na DNS datech získaných z reálného provozu a jeho testováním a analýzou výsledků byla ověřena funkčnost implementovaných detektorů.



# Abstract

The aim of this thesis is the detection of malicious spammer hosts based on passive analysis of captured DNS traffic. It represents the design and implementation of a system which proceeds DNS anomaly detection based on high volume of MX query per host and high NXDomain ratio. The system was tested on DNS data obtained from the real traffic and the functionality of implemented detectors was verified by testing and analysis of results.



# Klíčová slova

DNS, MX, NXDomain, detekce, analýza pasivního DNS provozu, PCAP, botnet

# Keywords

DNS, MX, NXDomain, detection, passive DNS analysis, PCAP, botnet


# Citace

PLOTĚNÝ, Ondřej. *Detekce spamu pomocí DNS MX záznamů*. Brno, 2016. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Kováčik Michal.

# Detekce spamu pomocí DNS MX záznamů

## Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Michala Kováčika. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Ondřej Plotěný
17. května 2016

## Poděkování

Rád bych poděkoval vedoucímu bakalářské práce Ing. Michalovi Kováčikovi za cenné rady, poskytnuté materiály a připomínky při tvorbě práce.

© Ondřej Plotěný, 2016.
*Tato práce vznikla jako školní dílo na Vysokém učení technickém v Brně, Fakultě informačních technologií. Práce je chráněna autorským zákonem a její užití bez udělení oprávnění autorem je nezákonné, s výjimkou zákonem definovaných případů.*

# Obsah


**1 Úvod** **3**

**2 DNS** **4**
2.1 Architektura DNS . . . 4
2.1.1 Prostor doménových jmen . . . 4
2.1.2 DNS server . . . 5
2.1.3 Resolver . . . 6
2.2 Protokol DNS . . . 7
2.3 Rezoluce DNS . . . 8
2.4 Zabezpečení DNS . . . 9
2.5 Bezpečnostní rizika DNS . . . 10
2.5.1 Fast flux . . . 10
2.5.2 Odepření služby . . . 10
2.5.3 Zneužití vyrovnávací paměti . . . 10
2.5.4 Tunelování DNS . . . 11
2.5.5 Anomálie DNS provozu . . . 11

**3 SPAM a botnet** **12**
3.1 Botnet . . . 12
3.2 Životní cyklus botů . . . 13
3.3 Architektura botnetu . . . 13
3.4 Metody detekce botnetu . . . 14
3.4.1 Detekce pomocí NXDOMAIN . . . 14
3.4.2 Detekce pomocí MX . . . 14

**4 Monitorování síťových dat** **15**
4.1 NetFlow a IPFIX . . . 15
4.2 Souborový formát pcap . . . 15

**5 Návrh systému** **16**
5.1 Vstupní data . . . 16
5.2 Parser DNS . . . 17
5.3 Detektor anomálií . . . 17
5.3.1 Výpočet koeficientu . . . 18
5.3.2 Určení mezních hodnot . . . 18
5.4 Výstup . . . 18

**6 Implementace** **19**
6.1 Parametry programu . . . 20
6.2 Zpracování vstupních dat . . . 20
6.2.1 Zpracování binárního vstupu . . . 20
6.2.2 Zpracování textového vstupu . . . 21
6.3 Struktury hosta a jeho mapování . . . 21
6.4 Analýza DNS provozu . . . 21
6.5 Implementace detektoru . . . 23
6.6 Výstup . . . 24

**7 Testování** **25**
7.1 Modelová situace . . . 25
7.2 Reálný provoz . . . 26
7.2.1 Analýza provozu . . . 26
7.2.2 Testování detekce . . . 26
7.2.3 Zaznamenané anomálie . . . 27
7.3 Optimalizace . . . 28
7.4 Analýza výsledků . . . 29

**8 Závěr** **30**

**Literatura** **31**

**Přílohy** **33**
Seznam příloh . . . 34

**A Obsah CD** **35**

**B Metriky kódu a parametry pro spuštení** **36**

**C Vývojový diagram detektoru** **37**

**D Analýza modelu** **38**

**E Analýza provozu reálné sítě** **39**

**F Test 1** **41**

**G Detekce po optimalizaci** **42**

# Kapitola 1

# Úvod

Internetová komunikace tvoří neodmyslitelnou součást profesního i soukromého života. Otevřenost internetu však dává prostor k zneužití komunikačních kanálů a služeb k šíření reklamy nebo virů, které obtěžují běžné uživatele a přidělávají vrásky na čele nejednomu síťovému administrátorovi.

Jedním ze základních stavebních kamenů internetu je služba Domain Name System (zkráceně DNS), která usnadňuje uživatelům adresování jednotlivých stanic či sítí v rámci celého internetu. Jedná se o celosvětovou databázi IP adres a jejich slovních ekvivalentů, bez kterých by bylo praktické použití ostatních internetových služeb (webových, poštovních, aj.) značně neintuitivní. Překladem doménového jména začíná prakticky každá internetová komunikace, včetně výše zmíněné komunikace nevyžádané.

Tato práce v následující kapitole představuje princip fungování služby DNS, její architekturu a komunikační protokol. Dále uvádí možnosti zneužití a typy anomálií jejího provozu.

Třetí kapitola představuje možnost využití škodlivých sítí, tzv. botnetů k šíření nevyžádané pošty a techniky používané pro jejich detekci za pomoci pasivního monitorování DNS provozu.

Čtvrtá kapitola uvádí prostředky pro monitorování sítí a záznam celých paketů pro hlubší analýzu.

Pátá a šestá kapitola představují návrh a implementaci systému detekující anomálie provozu se zaměřením na odhalení škodlivých stanic rozesílajících nevyžádanou poštu z pohledu poskytovatele internetu.

Sedmá kapitola uvádí testování implementovaného systému a analýzu výsledků a v závěru práce jsou uvedeny návrhy na vylepšení implementovaného detektoru a možné pokračování práce.

# Kapitola 2

# DNS

Každé zařízení v Internetu má svůj globálně unikátní číselný identifikátor zvaný IP adresa. Pomocí IP adresy je možné jednoznačně identifikovat adresáta komunikace. Číselný formát IP adresy je vhodný pro strojové zpracování, avšak pro uživatele Internetu je příliš složitý a těžko zapamatovatelný (např. u IPv6 se jedná o 128 bitové číslo). Z tohoto důvodu vznikla služba DNS, která zavádí slovní identifikaci stroje (tzv. doménové jméno) a zajišťuje mapování ze slovního názvu na příslušnou IP adresu. Uživatel tedy do svého webového prohlížeče zadává pouze doménové jméno webové služby, např. *www.google.com*, namísto složité IP adresy serveru a portu *74.125.136.104:80*.

Překlad adres DNS využívá nejen protokol HTTP webového prohlížeče, ale všechny aplikační protokoly, kde lze adresu zadat v podobě doménového jména (Telnet, SMTP, FTP). Navázání spojení je totiž možné až po překladu doménového jména na IP adresu. DNS tak patří k nejpoužívanějším službám v Internetu, ačkoli běžný uživatel o využívání této služby nemusí vůbec vědět. Navíc DNS neprovádí pouze překlad doménových jmen na IP adresy, ale také poskytuje reverzní překlad z IP adresy na doménové jméno, uchovává informace o poštovních serverech v doméně či poskytuje podporu pro IP telefonii, aj.

## 2.1 Architektura DNS

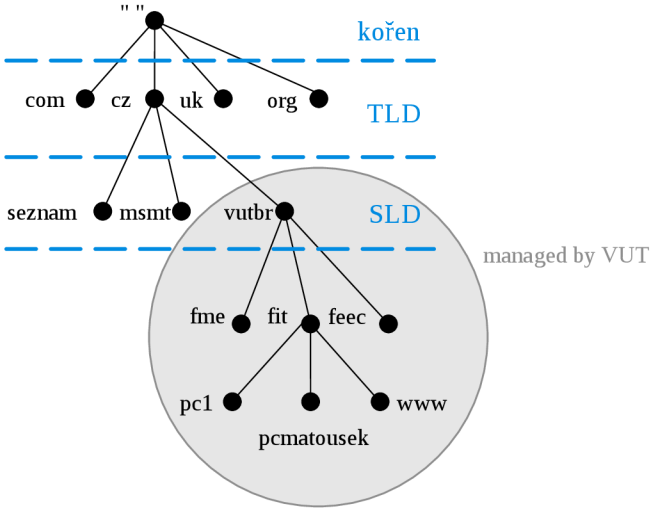
Původně překlad jmen prováděl každý počítač v síti sám za pomoci jediného souboru *HOSTS.TXT*. Tento soubor uchovával informace o mapování všech doménových jmen na IP adresy a jeho kopie byla distribuována všem počítačům připojených v síti. S rostoucím počtem doménových jmen, však přestal být překlad pomocí jediného souboru efektivní. Z tohoto důvodu vznikl systém DNS, který již nevyužívá souboru *HOSTS.TXT*, ale vytváří konzistentní databázi jmen (name space) uloženou na DNS serverech. Přístup k záznamům v databázi pak zajišťuje tzv. resolver pomocí dotazů na tyto servery [15].

### 2.1.1 Prostor doménových jmen

Standard [15] definuje prostor všech doménových jmen jako hierarchickou stromovou strukturu s jedním kořenem (označován jako *the root*), obr. 2.1. Uzly stromu a listy jsou označeny identifikátory o maximální délce 63 znaků, s výjimkou kořene, jehož identifikátor je prázdný řetězec. V rámci jedné úrovně stromu jsou identifikátory unikátní, avšak uzly různých úrovní mohou mít identifikátor stejný. Doménové jméno vznikne složením všech identifikátorů po cestě od listu až ke kořenu stromu. Identifikátory jednotlivých úrovní se oddělují tečkou.

Strom doménových jmen lze rozdělit na domény prvního řádu (TLD – Top-Level Domain), druhého řádu (SLD – Second-Level Domain) a další domény nižších řádů, obr. 2.1[1]. Domény prvního řádu jsou domény o úroveň nižší než je kořen stromu (např. národní TLD `.cz`, `.uk` nebo generické TLD `.com`, `.edu`). Domény druhého řádu jsou nazývány domény o 2 úrovně nižší než je kořen, atd.

Přidělování doménových jmen je taktéž hierarchicky uspořádané, kde každý uzel má svého registrátora (např. pro doménu `.cz` je to CZ.NIC). Primární autoritou mezi jednotlivými registrátory a celkovou správu DNS systému má na starosti organizace ICANN[2].




Obrázek 2.1: Hierarchická struktura doménových jmen v DNS [14]

### 2.1.2 DNS server

Jmenný server je program, který udržuje kompletní a důvěryhodné (autoritativní) informace o určité souvislé části doménového stromu, tzv. *zóně* a informace o dalších jmenných serverech důležitých pro vyhledávání v hierarchii (např. kořenové servery). Zóna je část stromu, kterou spravuje jeden správce. Může se jednat o jednu celou doménu, více domén nebo pouze o subdoménu. Jeden fyzický server může spravovat více zón. Stejně tak, o jednu zónu se může starat více serverů. V takovém případě je určen jeden primární server a jeden či více sekundárních serverů. Primární server obsahuje úplné a aktuální informace o spravovaných doménách a sekundární server si pomocí tzv. zónového přenosu tyto informace synchronizuje. Pouze odpovědi primárního a sekundárního serveru se považují za autoritativní.

Dále je možné využít tzv. *caching-only* serveru, což je záložní proxy server mezi klientem a autoritativním serverem. Odpovědi záložního serveru jsou sice neautoritativní a mohou být neaktuální, ale zrychlují proces překladu doménového jména.

Uspořádání serverů v systému DNS odpovídá hierarchické struktuře doménového stromu. Páteř celého systému tvoří 13 tzv. kořenových serverů (označeny písmeny A-M), které jsou na začátku hierarchie a uchovávají informace o serverech s TLD doménami. V praxi jsou tyto kořenové servery rozprostřeny mezi mnoho fyzických zařízení se stejnou anycastovou IP adresou.

[1] obrázek převzat a přidán modrý text

[2] Internet Corporation for Assigned Names and Numbers), www.icann.org

#### Záznamy

Jmenný server uchovává jednotlivé informace o zóně v textovém tzv. *zónovém souboru*, v podobě zdrojových záznamů (resources records). Zdrojové záznamy jsou v souboru zapsány ve formátu: jméno uzlu stromu (domény), platnost záznamu, třída (typicky IN, značící Internet), typ záznamu a hodnota (závisí na typu a třídě) [16]. Dle charakteru dat, které reprezentují, jsou rozlišovány záznamy mnoha typů, zde jsou uvedeny jen ty nejběžnější:

**SOA – Start of Authority**
: Záznam definuje uložení autoritativních dat pro danou zónu. Slouží jako hlavička pro každou zónu. Mimo jiné obsahuje jméno primárního serveru, informace o správci, dobu platnosti či sériové číslo aktualizace.

**NS – Name server**
: Záznam o autoritativním serveru pro danou doménu. Obsahuje informace o následnících v hierarchii DNS.

**A, AAAA – Address**
: Záznam o mapování doménového jména na IPv4 resp. IPv6 adresu.

**MX – Mail Exchanger**
: Záznam o poštovním serveru pro danou doménu. V záznamu se udává priorita poštovního serveru (nižší číslo = vyšší priorita). V případě nedostupnosti serveru s vyšší prioritou se pošta doručí serveru s prioritoiu nižší. Pokud pro danou doménu není MX uveden, není možné poštu doručit.

**CNAME – Canonical Name**
: Záznam o mapování více služeb na jedinou IP adresu. Záznam vytváří alias k oficiálnímu jménu stroje.

**PTR – Domain Name Pointer**
: Záznam o reverzním mapováním IP adresy na doménové jméno. Reverzní adresy tvoří vlastní hierarchický strom ve speciální doméně in-addr.arpa.

**NAPTR – Naming Authority Pointer**
: Záznam o mapování domény na SRV záznam. NAPTR záznam obsahuje regulární výraz a řetězec doménové adresy. Slouží např. pro podporu SIP protokolu.

**TXT – Text**
: Dodatečné textové informace o doméně.

**SRV – Service Record**
: Záznam upřesňující informace (jméno hostitele a číslo portu) o dostupných službách a serverech. Slouží např. pro rozdělení zátěže či IP telefonii.

### 2.1.3 Resolver

Resolver je klientský program (součást OS), který se dotazuje jmenného serveru na informace o doméně. Je nezbytné, aby měl resolver nakonfigurován přímý přístup alespoň k jednomu lokálnímu serveru a byl také schopný dotazovat se jiných serverů, na které obdrží referenci. Další funkcí resolveru je interpretovaní přijatých odpovědí v podobě záznamů či chybová hlášení.

## 2.2 Protokol DNS

Komunikace mezi DNS serverem a resolverem či jiným serverem probíhá pomocí aplikačního protokolu DNS na portu 53 [16]. Standardně se ke spojení používá transportní protokol UDP, který však nezaručuje bezchybný přenos. Proto se pro přenos zónových souborů mezi servery využívá spolehlivějšího transportního protokolu TCP. Protokol TCP je použit i v případě, že odpovědi serveru jsou větší než 512B – klient obdrží od serveru částečnou odpověď s příznakem *Truncation* v hlavičce a klient naváže spojení TCP.

### Formát DNS zprávy

Všechny zprávy protokolu DNS mají stejný formát a jsou složeny z 5 základních sekcí[16]:

**Havička** (Header) obsahuje základní informace o přenášených záznamech:

**ID** 16-bitový identifikátor DNS zprávy

**QR** příznak dotazu (0) nebo odpovědi (1)

**Opcode** označení typu dotazu (standardní dotaz, inverzní dotaz, status, aj.)

**AA** (Authoritative Answer) příznak autoritativní odpovědi

**TC** (Truncation)příznak signalizující zkrácení odpovědi

**RD** (Recursion Desired) požadavek na rekurzivní zpracování dotazu

**RA** (Recursion Available) příznak serveru, který poskytuje rekurzivní odpovědi.

**Res** Rezorvováno pro budoucí využití, obsahuje 0

**RCODE** (Response code) návratový kód označující výsledek dotazu

0. bez chyby
1. chyba ve formátu dotazu
2. chyba na straně serveru (porucha)
3. autoritativní server tím oznamuje, že požadovaný záznam neexistuje
4. nepodporovaný typ dotazu
5. odmítnuto

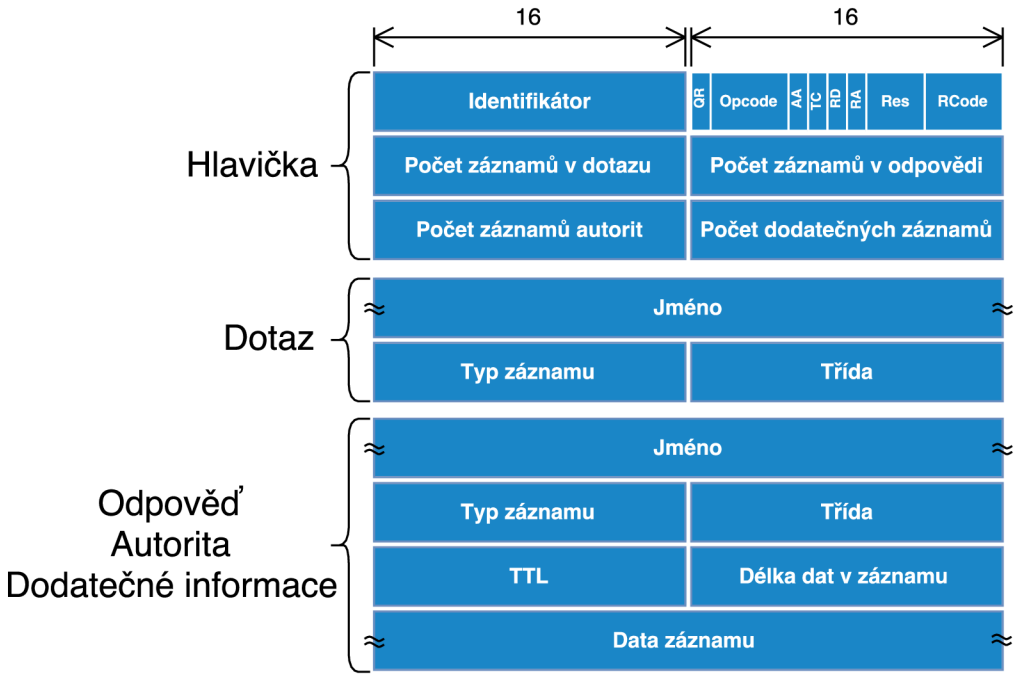
**Dotaz** (Question) dotaz klienta obsahující jméno dotazu, typ záznamu(*QType*) a třídu

**Odpověď** (Answer) DNS záznamy, které odpovídají na vznesený dotaz

**Informace o autoritě** (Authority) seznam autoritativních DNS serverů zóny, ze které byly záznamy získány

**Dodatečné informace** (Additional section)

Obr. 2.2 znázorňuje položky v jednotlivých sekcí (sekce Odpověď, Autorita a Dodatečné informace mají stejný formát).




Obrázek 2.2: Formát DNS zpráv

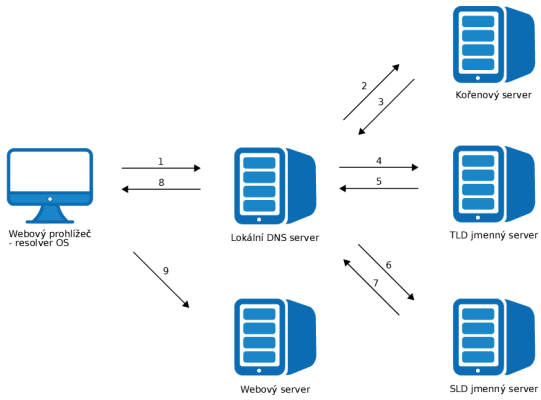
## 2.3 Rezoluce DNS

Rezoluce DNS je proces překladu doménového názvu v systému DNS [14]. Prakticky se jedná o vyhledání libovolného uzlu v DNS stromu.

Rezoluce začíná v klientském programu, který vytvoří požadavek na překlad doménového jména na IP adresu (např. již zmíněným zadáním doménového jména do webového prohlížeče). Klientský program předá požadavek resolveru operačního systému. Resolver nejprve zkusí doménu vyhledat ve své cache paměti. V případě neúspěchu přeposílá dotaz svému lokálnímu DNS serveru (konfigurace pomocí DHCP nebo v Unixu soubor `/etc/resolv.conf`.). Lokální DNS server (většinou jde o DNS server poskytovatele připojení) vyhledá odpověď ve svých záznamech a vlastní cache paměti. Pokud danou doménu nezná, musí ve stromové hierarchii najít její autoritativní server.

Vyhledávání ve stromu začíná od kořenových serverů. Pokud jakýkoli DNS server danou doménu přímo nezná, určí směr dalšího prohledávání stromu podle svých NS záznamů a nejlepší možné shody s identifikátory překládané domény (bráno zprava). Kořenový server tedy odpoví lokálnímu serveru svým NS záznamem pro příslušnou TLD doménu (např. `a.ns.nic.cz` pro `.cz`). Lokální server se postupně dotáže autoritativních serverů TLD domény, které obdržel v odpovědi od kořene. Pokud ani TLD servery neznají přímo odpověď, lokální server opět obdrží referenci na autoritativní servery příslušné SLD domény, na které lokální server má své požadavky směřovat. Proces vyhledávání určitého následníka ve stromové hierarchii pokračuje, dokud lokální server nenajde autoritativní server pro překládané doménové jméno. Tento autoritativní server již vrátí lokálnímu serveru záznam obsahující IP adresu (A záznam) webové služby. Lokální DNS si obdrženou odpověď uloží do cache paměti a předá ji resolveru. Resolver vrátí klientskému programu IP adresu a taktéž si odpověď uloží do své cache paměti. Celý proces rezoluce je naznačen na obr. 2.3.

Při rezoluci se rozlišují 2 typy dotazů – iterativní a rekurzivní. Zvolený typ dotazu záleží




Obrázek 2.3: Rezoluce DNS [17]

na konfiguraci zařízení. U výše popsané rezoluce je klientský resolver rekurzivní a lokální DNS server iterativní.

**Iterativní dotaz**
Iterativní způsob je šetrnější k serveru DNS. V tomto případě Resolver opakovaně (iterativně) posílá dotazy serverům v hierarchii a DNS servery vrací pouze nejlepší možnou odpověď.

**Rekurzivní dotaz**
Při rekurzivním dotazováním resolver obdrží buď odpověď na požadovaný záznam, nebo chybové hlášení. O průchod hierarchií se starají dotazované servery.

## 2.4 Zabezpečení DNS

Původní koncept systému DNS neposkytoval žádnou ochranu proti falšování údajů v databázi, proto byli v roce 2005 vydány standardy RFC 4033, 4034 a 4035 rozšiřující službu DNS o mechanismy ověřování pravosti údajů nebo ověření jejich neexistence, tzv. DNSSEC. K zabezpečení pravosti záznamu využívá DNSSEC asymetrickou kryptografii. Principem asymetrické kryptografie je použití dvou klíčů – soukromého klíče a veřejného klíče.

Primární server vygeneruje soukromý klíč, kterým podepíše jednotlivé zdrojové záznamy (vytvoří hash záznamu). Současně s tajným soukromým klíčem vygeneruje i volně dostupný veřejný klíč. Dvojce klíčů je na sobě algoritmicky závislá – danému soukromému klíči přísluší pouze jeden veřejný klíč, proto je možné pomocí veřejného klíče ověřit integritu záznamu a autentizaci vlastníka podpisu.

Pro potvrzení důvěryhodnosti klíčů je využíván další pár klíčů, tzv. Key Signing Key (KSK) - klíč pro podpis klíče. O ověření pravosti klíče KSK se stará hierarchicky nadřazený uzel, vzniká tak propojení klíčů v rámci DNS hierarchie zvané *řetězec důvěry* [14].

Dalším typem zabezpečení autentizace je podpis transakcí pomocí TSIG. Na rozdíl od DNSSECu, TSIG pracuje se symetrickou kryptografií a ověřuje transakce (zdroj a cíl), nikoliv jednotlivé záznamy DNS. Používá se např. při zónovém přenosu [22].

## 2.5 Bezpečnostní rizika DNS

DNS je otevřená, veřejná a nešifrovaná služba, je tedy snadné zneužít či jinak znehodnotit jak prostor doménových jmen tak samotnou serverovou infrastrukturu.

Příkladem zneužití doménového stromu je tzv. *Cybersquatting* a *Typosquatting*. Jedná se o registraci domény stejného či podobného názvu (např. záměna jednoho písmene) na úkor obchodní značky nebo známého jména. Účelem je prodej domény nebo pošpinění dobrého jména firmy. Dalším typem zneužití domény je krádež identity registrovaného majitele domény a následná změna údajů u registrátora tzv. *Domain name hijacking* nebo registrace domény ihned po uplynutí expirační doby registrace domény tzv. *Domain sniping* [10].

### 2.5.1 Fast flux

Fast flux domény jsou škodlivé domény s rychle se měnícími DNS záznamy. Typicky se jedná o domény C&C serverů, které se snaží vyhnout blokování svých IP adres pomocí IP black – listů. Rychlé změny v DNS záznamech se projevují v hodnotách TTL.

### 2.5.2 Odepření služby

Při útoku na DNS server se jedná zejména o techniku odepření služby (Denial of Service, DoS), kdy server není schopen kvalitně obsloužit požadavky klientů. Odepření služby je docíleno buďto vyčerpáním technických zdrojů stroje, změnou jeho konfigurace či zneužitím softwarových chyb.

Aby útočník byl schopen vyčerpat technické zdroje určitého serveru či sítě je zapotřebí většího množství počítačů (označované jako distribuované útoky), které jsou většinou součástí *botnetu*. Nejčastějším typem útoku je reflektivní útok, který pomocí podvržené IP adresy zneužije DNS server k zahlcení linky oběti. Tento útok se používá v kombinaci s technikou zesilujícího útoku (*DNS Amplification attacks*), při kterém se využívá vhodně zvolených dotazů, na které DNS server odpovídá až 7x většími odpověďmi (ve speciálních případech je zesílení až 70násobné).

Další možností je využití TCP handshaku a zahltit server TCP SYN pakety, které však nikdy nenavážou spojení. U rekurzivních serverů je možné využít tzv. NXDOMAIN útoku, kdy je server zaplaven dotazy na neexistující domény a jeho vyrovnávací paměť se zaplní NXDOMAIN záznamy.

### 2.5.3 Zneužití vyrovnávací paměti

Základem útoku je podvržení záznamů, které si server ukládá do své vyrovnávací paměti (cache poisoning). Falešné záznamy jsou pak doručovány klientům za účelem přesměrování komunikace na servery ovládané útočníkem. Útok zneužívá jednoduchosti autentifikace příchozích odpovědí od autoritativních DNS serverů. Stačí mít shodný cílový port, zdrojovou IP adresu a 16-bitový identifikátor transakce, který lze uhádnout hrubou silou a doručit odpověď dříve než odpoví legitimní DNS server. Útočník rozesílá své falešné DNS odpovědi buď při odposlechu dotazu cizího resolveru, nebo sám odesílá dotazy. Řešením toho otrávení záznamů je použití ověřování DNSSEC či TSIG.

### 2.5.4 Tunelování DNS

Jedná se o techniku zapouzdření cizích protokolů do DNS paketů. Zapouzdřením je možné obejít firewally v obou směrech (ven i dovnitř sítě), je však nutné využít tunelovací DNS server, který je schopen zapouzdřený protokol interpretovat.

### 2.5.5 Anomálie DNS provozu

Anomálie DNS provozu lze rozlišit podle příznaků [10]:

**Časové příznaky**
: Krátký život, denní podobnost, opakované vzory, poměr přístupů

**Příznaky DNS odpovědí**
: Počet odlišných IP adres, počet odlišných zemí, počet domén se shodnou IP adresou, výsledek reverzního DNS dotazu

**Příznaky TTL**
: Průměrná hodnota TTL, standardní odchylka od TTL, počet odlišných TTL hodnot, počet změn hodnoty TTL, Procentuální zastoupení určitých rozsahů TTL

**Příznaky doménového jména**
: % numerických znaků, % délka LMS

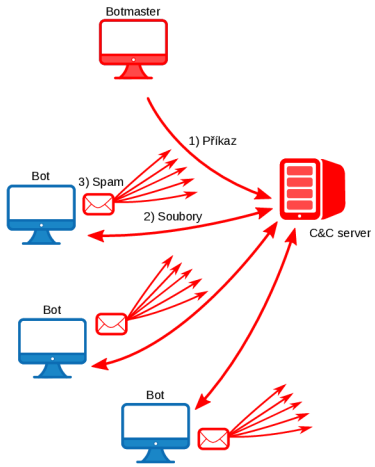
# Kapitola 3

# SPAM a botnet

SPAM je označení pro hromadnou, obtěžující a nevyžádanou elektronickou komunikaci třetí strany [19]. Zneužitým komunikačním kanálem může být elektronická pošta, Instant Mesenger (ICQ, Skype), Facebook či IP telefonie. Spam nejenom obtěžuje běžné uživatele nevyžádanou reklamou, ale je také využíván k šíření malwaru či phishingu a velké množství spamu může způsobit odepření služby či vyčerpání síťových zdrojů. V případě spamu šířeného e-mailem je zneužití snadné díky poštovnímu protokolu SMTP, který neposkytuje žádné mechanismy autentizace uživatelů. Podle společnosti ENISA v roce 2015 tvořilo spam 53% e-mailové komunikace. Dále uvádí, že zdrojem spamu jsou ze 70% botnety [12].

## 3.1 Botnet

Botnet je centrálně řízená síť počítačů infikovaných nějakým druhem malwaru[1]. Název botnetu je většinou odvozen právě od malwaru, který jej šíří. Řídícím bodem botnetu je tzv. botmaster, který pomocí Command and control (C&C) serveru kontroluje a zasílá příkazy infikovaným stanicím (tzv. botům). Typicky jsou boti zneužiti k nekalé činnosti na síti, např. pro DDoS útoky, phishing, rozesílání spamu (viz obr. 3) či šíření infekce na další počítače [9].




Obrázek 3.1: Struktura botnetu

[1] „malicious software“ - souhrnné označení pro škodlivý program, např. virus, trojský kůň spyware, rootkit aj.

## 3.2 Životní cyklus botů

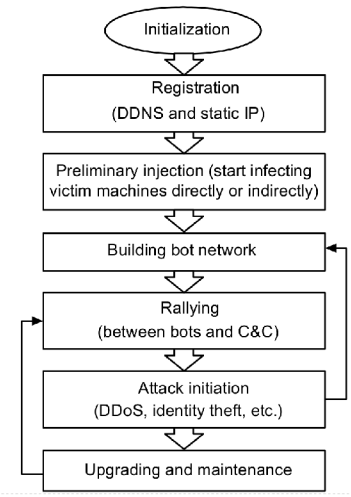
Obecně se práce botnetu rozděluje do čtyř fází: rozšíření infekce, navázání spojení s C&C serverem, provedení útoku a tzv. post – attack fáze [3].

Ve fázi rozšíření infekce nejprve botmaster nastaví parametry komunikace s botem (Initialization) a provede registraci statické IP adresy pro C&C server a registraci DDNS (Registration). Botmaster často využívá metody fast – flux.

Samotné šíření malwaru (preliminary injection) probíhá např. pomocí škodlivé přílohy nevyžádané pošty, vyměnitelných médií či stažením z pochybných webů uživatelem. Neopatrným či nechtěným spuštěním závadného softwaru dojde k vyhledání zdrojových kódů malwaru, jejich stažení ze vzdálené databáze a nainstalování do infikovaného systému (Building bot network). Pro stažení binárních kódů se nejčastěji využívá HTTP, FTP nebo p2p protokol. Po nainstalování se napadený systém stává botem.

Fáze spojení s C&C serverem (označovaná také jako Rallying) nastává po nainstalování do systému a poté při každém jeho restartu. V této fázi dochází k navázání spojení mezi botem a jeho řídícím serverem, aby mohl bot přijímat příkazy serveru. Jednou z hojně využívaných technik používaných k nalezení řídícího serveru je zasílání velkého množství domén vygenerovaných pomocí DGA (Domain Generation Algorithm).

V další fázi bot čeká na příkazy od serveru a provádí škodlivou činnost podle obdržených příkazů či jeho vnitřní konfigurace. Zároveň odesílá získaná data zpět na server. Poslední fází je tzv. post – attack nebo také údržbová fáze, kdy probíhá konfigurace botů pro další útoky. V této fázi probíhá také aktualizace binárních kódů botů, což umožňuje botnetu měnit své stereotypní chování a komplikovat tak své odhalení. Avšak samotné šíření aktualizací k jednotlivým botů je snadno detekovatelné [8].




Obrázek 3.2: Životní cyklus botnetu [8]

## 3.3 Architektura botnetu

Důležitou vlastností botnetu je možnost flexibilního propojení počítačů a jejich vzdálená kontrola. Podle typu infrastruktury mezi prvky botnetu se rozlišují 3 architektury, a sice centralizovaná (HTTP, IRC), decentralizovaná(p2p) a hybridní architektura.

## 3.4 Metody detekce botnetu

Metody detekce botnetu se rozlišují na pasivní a aktivní. Pasivní metody pouze monitorují síťový provoz, zatímco aktivní metody se sami zapojují do síťového provozu. Výhodou pasivních metod je, že nezatěžují linky provozem navíc, za cenu větší paměťové náročnosti.

Metody je možné dále dělit podle typu sledovaných dat na detekce anomálií (anomaly – based), nebo na tzv. Signiture – based metody. Výhodou metody detekce podle anomálií je možnost odhalení činnosti bota bez detailních znalostí botnetu a detekovat tak nově vznikající botnety, zatímco Signiture – based metody pracují s typickým chováním botnetu a patří mezi ně techniky založené na black – listech a white – listech.

Dalšími metodami jsou například Honeynety, které emulují napadnutelné systémy a analyzují nové typy malwaru, či tzv. host – based detektory, které sledují pouze změny na jednotlivých stanicích (např. antivirové programy). Dále se v práci budu zabývat pouze detekcemi založenými na pasivním monitorování anomálií DNS provozu na síti.

### 3.4.1 Detekce pomocí NXDOMAIN

Jedna z možných pasivních metod detekce botů pomocí DNS anomálií je představena v [21] a [1]. Detekce je založena na typickém chování bota v počáteční fázi útoku, kdy se snaží kontaktovat svůj C&C server s pomocí DGA. Při této fázi bot zasílá nadměrné množství DNS dotazů na pseudonáhodné domény. Na většinu dotazů obdrží odpověď, že doména neexistuje (`NXDOMAIN`, návratový kód 3). [1] uvádí, že většina stanic v síti (90 %) obdrží během dne řádově jednotky NXDOMAIN odpovědí, avšak bot je schopen generovat desítky až stovky těchto dotazů.

Statistický způsob detekce pomocí NXDOMAIN je použit v [13], kde pro každého hosta je spočítán poměr mezi počtem odeslaných dotazů a počtem obdržených DNS odpovědí s návratovým kódem 3. Stanice s poměrem vyšším než stanovené mezní hodnota je prohlášena za bota. Mezní hodnoty vycházely z analýzy provozu na lokální síti během 24 hodin a z počtu stanic v lokální síti.

V práci [18] je detekce založena pouze na DNS odpovědích. Poměr je určen mezi odpověďmi s návratovým kódem 0 (NoError) a odpověďmi s návratovým kódem 3 (NXDOMAIN). Autoři využívali analýzu paketů zachycených v `.pcap` souborech a detektor neexistujících domén kombinovali s detektorem TCP SYN – flooding útoků zachycených pomocí IPFIX.

Detekce založená na NXDOMAIN se zaměřuje na botnety využívající GDA (Conficker, Dridex). Komplikací pro použití této metody může být vyšší procento falešně pozitivních hlášeních, způsobených chybějícími AAAA záznamy či chybně nakonfigurovanými hosty a aplikacemi, a také u rozsáhlejších sítích překladem NAT. Obtížné je určení mezních hodnot. Výhodou metody je možnost odhalení jak samotného bota, tak i jeho C&C serveru. Podle [13] je tato technika efektivnější než analýza skladby doménového jména.

### 3.4.2 Detekce pomocí MX

Příliš časté dotazování na DNS záznamy typu MX (QType = 15) je typickým znakem botů rozesílajících spam, indikují totiž snahu poslat velké množství elektronické pošty. V práci [6] autor určil pro každou zdrojovou IP adresu počet MX dotazů a stanice překračující mez označil za spammery. I v tomto případě je mezní hodnota určena podle analýzy řádného provozu na síti (autor stanovil 100 dotazů). Nevhodně zvolenou mezní hodnotou dochází ke zvýšenému poštu falešně pozitivních hlášení.

# Kapitola 4

# Monitorování síťových dat

## 4.1 NetFlow a IPFIX

NetFlow je proprietární síťový protokol vyvinutý firmou Cisco. Umožňuje síťovým zařízením udržovat si přehled o síťovém provozu procházející skrze jejich rozhraní a odesílání nasbíraných dat na jiné zařízení pro jejich analýzu. Takto nasbírané informace o síťovém provozu jsou využity například pro bezpečnostní analýzy, vyúčtování zákazníků či sledování vytíženosti sítě.

Zařízení, provádějící sběr dat se označuje jako `exporter` a jedná se většinou o směrovač, přepínač nebo jde o samostatnou NetFlow sondu. Exporter provádí monitorování paketů na jednom či více ze svých rozhraní (označované jako *Observation point*) a vytváří záznamy o síťových tocích. Získané záznamy (Flow record) periodicky odesílá na `kolektor`, což je zařízení, které přijímá záznamy od jednoho či více exporterů, provádí analýzu a ukládá záznamy na disk. Síťový tok (IP flow) je definován jako jednosměrná skupina paketů se stejnými vlastnostmi procházející skrz zařízení [4]. Mezi definující údaje toku patří:

- zdrojová a cílová IP adresa
- zdrojový a cílový port
- protokol síťové vrstvy
- typ služby (ToS)
- vstupní rozhraní

NetFlow protokol existuje v několika verzích. Verze 9 umožňuje definovat strukturu záznamu pomocí šablon a rozšířit tak záznam o další volitelné položky jako např. MPLS, IPv6 adresy, časové značky či počty paketů, aj.

Protokol IPFIX (Internet Protocol Flow Information eXport) je protokol standardizovaný IETF, vycházející z protokolu NetFlow v9 [5]. Stejně jako jeho předchůdce, je díky využití šablon, IPFIX značně flexibilní. Navíc umožňuje sledování toku v obou směrech (biflow) [20].

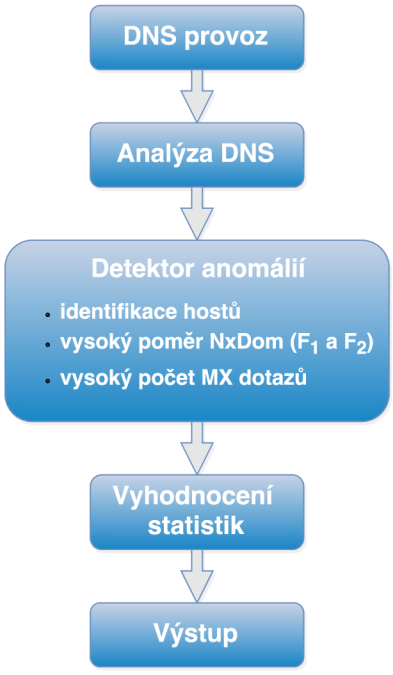
## 4.2 Souborový formát pcap

Binární souborový formát uchovávající kompletní informace o každém paketu procházející přes rozhraní. Vhodný je zejména pro hloubkovou analýzu paketů. Zachytávání paketů do `.pcap` souboru provádí např. program Wireshark, tcpdump nebo WinDump.

# Kapitola 5

# Návrh systému

Detekce navrhovaného systému spočívá v odhalení anomálie v DNS provozu představené v kapitole 3.4. Zkoumanou anomálií bude jednak vysoký počet dotazů na DNS MX záznamy a dále pak vysoký poměr dotazů na neexistující doménu (koeficienty $F$). Výsledkem by měly být informace o podezřelých tocích v síti, podle kterých by byla možná následná implementace bezpečnostních opatření.




Obrázek 5.1: Architektura systému

## 5.1 Vstupní data

Vstupními daty detektoru budou informace protokolu DNS odchycené ze síťového provozu. Aplikace získá DNS záznamy buď přímo ze síťového rozhraní, z komunikace uložené v `.pcap` souboru či z textového `.csv` souboru.

Vhodným prostředkem pro generování textového vstupu je DNS plugin pro Flowmon Exporter [11]. DNS plugin pracuje s IPFIX daty a umožňuje extrahovat jen některé části

DNS paketu a snížit tak paměťové nároky a výpočetní výkon, oproti záznamu celých paketů. Použití textového výstupu z DNS pluginu je vhodné pro analýzu rozsáhlejších sítí. Pro navrhovaný detektor je potřeba aby vstupní data obsahovala následující informace:

- Zdrojovou IP adresu
- Cílovou IP adresu
- Příznak QR
- Typ dotazu QType
- Návratový kód odpovědi Rcode

## 5.2 Parser DNS

V případě vstupu z `.pcap` souboru nebo přímo ze síťového rozhraní je DNS zpráva zapouzdřena v protokolech nižších vrsev TCP/IP modelu a je nutné celý rámec rozbalit a extrahovat položky DNS zpávy. Dle kapitoly 2.2 lze položky potřebné k detekci z DNS zprávy nalézt v hlavičce DNS (položka *Flags*) a v sekci s dotazem (položka *QType*). Není proto nutné zpracovat ostatní sekce DNS zprávy.

U textového vstupu provádí rozbalení a extrakci DNS položek Flowmon Exporter. Exporter však v základním režimu neposkytuje extrakci příznaku *QR*, proto je nutné tento příznak nahradit kombinací položek *ANSWER* a *RCODE*.

## 5.3 Detektor anomálií

Logika detektoru spočívá ve výpočtu statistiky provozu jednotlivých hostů v síti. Při odchycení DNS komunikace je nejdříve nutné určit, zda se jedná o dotaz na DNS záznam či odpověď, dle toho je možné určit IP adresa hosta, která poslouží jako jeho identifikátor v rámci detekce. Poté je možné dle typu záznamu inkrementovat informace o provozu hosta. Statistické údaje hostů zahrnují:

- Celkový počet odeslaných dotazů *host.query*
- Celkový počet obdržených odpovědí *host.resp*
- Počet dotazů na MX záznamy *host.mx_query*
- Počet odpovědí na MX záznamy *host.mx_resp*
- Počet odpovědí s návratovým kódem 3 *host.nxDom*
- Počet odpovědí s návratovým kódem 0 *host.noErr*
- Počet odpovědí s jiným návratovým kódem *host.other*

Po zpracování všech vstupních dat se provede výpočet statistik a koeficientu $F$. Host s hodnotami převyšující mezní hodnoty je označen jako potenciální bot. Potenciální bot s vysokým počtem dotazů na MX záznamy je označen jako podezřelý spam bot. Vývojový diagram navrhovaného algoritmu je uveden v příloze C.

### 5.3.1 Výpočet koeficientu

Určit hodnotu koeficientu $F$ udávající poměr validních a nevalidních dotazů pro každého hosta je možné pomocí 2 variant výpočtu:

1. poměr mezi počtem odpovědí s návratovým kódem 3 a všemi odeslanými dotazy

$$F_1 = \begin{cases} \frac{host.NxDom}{host.query} & \text{pro host.NxDom} > 0 \\ 0 & \text{pro host.NxDom} = 0 \end{cases}$$

2. poměr mezi počtem odpovědí s návratovým kódem 3 s návratovým kódem 0

$$F_2 = \begin{cases} \frac{host.NxDom}{host.NoErr} & \text{pro host.NxDom} > 0 \\ 0 & \text{pro host.NxDom} = 0 \\ 1 & \text{pro host.NoErr} = 0 \text{ a host.NxDom} > 0 \end{cases}$$

V navrhovaném systému budou nezávisle na sobě, zahrnuty oba způsoby výpočtu. Efektivnější způsob výpočtu koeficientu bude předmětem testování.

### 5.3.2 Určení mezních hodnot

Zásadním prvkem navrhovaného systému jsou mezní hodnoty. Jejich určení je závislé na provozu v dané síti. Optimální hodnoty je možné určit testováním či monitorováním zkoumané sítě.

Pro omezení falešných hlášení a zajištění objektivnosti detekce je nutné, aby host v síti měl určitý objem provozu. Pro zařazení do detekce je stanovena minimální mez 10 odeslaných dotazů pro koeficient $F_1$ a minimálně 10 přijatých DNS zpráv pro koeficient $F_2$.

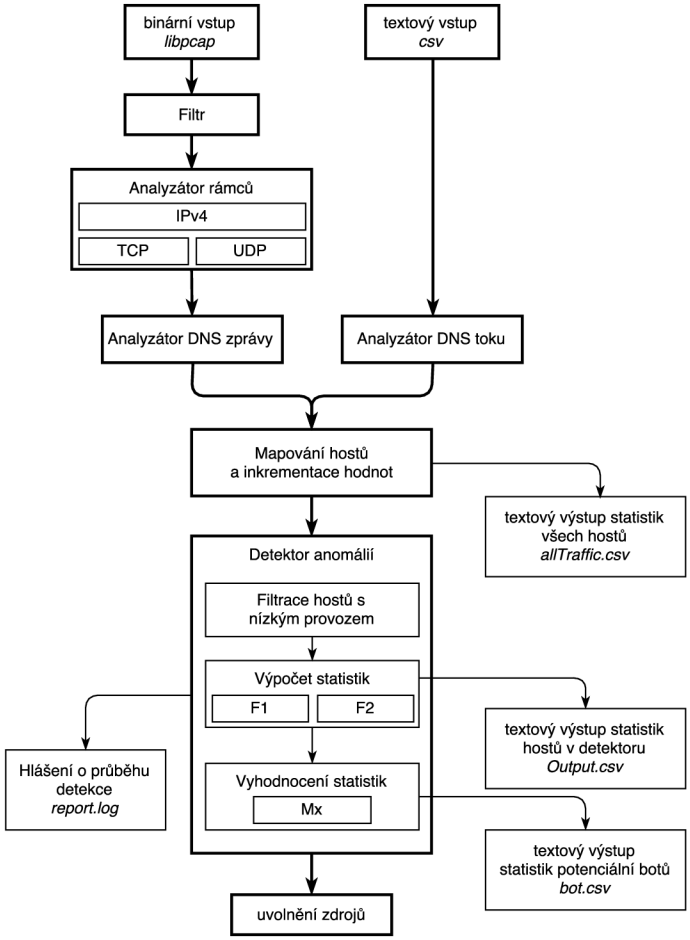
## 5.4 Výstup

Výstupem detektoru jsou výsledky statistik pro jednotlivé hosty ve formě tabulky a grafy celkového provozu. Vhodným formátem je textový `.csv` soubor, který lze využít pro další zpracování, např. pro vykreslení grafů pomocí nástroje `gnuplot`.

# Kapitola 6

# Implementace

Implementace sytému vychází z návrhu uvedeném v předchozí kapitole. Jedná se o konzolovou aplikaci implementovanou v programovacím jazyce C++. Nejdříve bylo nutné zpracovat vstupní data a následně implementovat navržené detektory. Jelikož IPv6 tvořilo jen 5% celkového provozu, je tento provoz zanedbán a použité struktury a filtry jsou optimalizovány pouze na provoz protokolu IPv4. Obr. 6.1 ukazuje architekturu detektoru. V následujících podkapitolách jsou popsány jednotlivé kroky.




Obrázek 6.1: Architektura implementovaného systému

## 6.1 Parametry programu

Aplikace při spuštění očekává parametry příkazové řádky, kterými lze specifikovat typ vstupu a mezní hodnoty pro detekci. Typ vstupu nelze kombinovat a je možné použít právě jeden z typu, stejně tak nelze zadat více vstupní souborů. Pokud uživatel nezadá mezní hodnoty pro detekci jsou použity výchozí hodnoty. Detailní popis parametrů je uveden v příloze B. Pro zpracování příkazové řádky je využito knihovny `getopt`.

## 6.2 Zpracování vstupních dat

Aplikace umožňuje použití 2 různých vstupů, a sice odchycených paketů v binární podobě anebo textového souboru se záznamem DNS toků. Binární vstup je vhodnější u menších sítí a pro online analýzu například lokálního DNS serveru. Textový vstup umožňuje analýzu rozsáhlých páteřních sítí.

### 6.2.1 Zpracování binárního vstupu

Pro načtení binárního vstupu byla využita knihovna `libpcap`. Tato knihovna umožňuje číst binární data přímo ze síťového rozhraní nebo ze souboru `.pcap`. Pokud uživatel zadá jako vstup síťové rozhraní, je rozhraní otevřeno pomocí funkce `pcap_open_online`, k zavření souboru dochází po přijetí signálu `SIGINIT` nebo `SIGTERM`. Pokud je zadán vstup `.pcap` soubor, je otevřen voláním funkce `pcap_open_offline`. Další práce s binárními daty je totožná a nezávisí na druhu (online – offline) vstupu.

Knihovna `libpcap` umožňuje filtrovat načítaná data. Filtr je aplikován pomocí knihovních funkcí `pcap_compile` a `pcap_setfilter`. Filtr je nastaven na propuštění pouze DNS provozu s následujícími parametry: Ethernetový rámec (včetně VLAN tagu[1]), IPv4 paket, transportní protokol UDP nebo TCP a port 53. Dále jsou odfiltrovány zprávy navazování a ukončování TCP spojení.

Načítání jednotlivých rámců probíhá ve funkci `pcap_loop`, která pro každý rámec zavolá funkci `dispatcher_handler`. Zde probíhá samotná analýza Ethernetového rámce a data jsou sem předána již bez preambule a SFD. Rámec neposkytuje žádné cenné informace pro detektor a tak je pouze zkontrolován typ Ethernetové hlavičky a určena délka hlavičky (pro optimalizaci standardu 802.1Q). Dle délky hlavičky rámce je možné určit začátek IPv4 hlavičky následující za rámcem a z ní vyčíst informace o IP adresách, transportním protokolu a velikosti přenášených dat. Položky IPv4 hlavičky jsou ve struktuře `ip_header`.

Pokud je transportním protokolem UDP, je transportní hlavička přeskočena a zbylá data jsou přímo analyzována jako DNS zpráva. Při využití transportního protokolu TCP může být analyzovaný paket pouze zkráceným paketem větší DNS zprávy, proto jsou data paketu nejdříve uložena a při obdržení TCP zprávy s příznaky ACK|PSH je kompletní zpráva znovu sestavena a předána k analýze DNS.

Pro opětovné sestavení zkrácených zpráv je nutné udržovat přehled jednotlivých TCP relací. K tomu slouží struktury `tcp_session`, které jsou svázány v obousměrném seznamu. Každá relace pak obsahuje svůj seznam již obdržených zkrácených paketů. Při obdržení TCP příznaku PUSH je identifikována relace, seřazeny pakety podle sekvenčních čísel TCP hlavičky a data spojena do jednoho celku a předána k analýze. K implementaci seznamu paketů je využito standardní knihovny `list`. Pokud právě zpracovávaný TCP datagram ne-

[1] formát rámce podle standardu 802.1Q

patří k žádné zaznamenané relaci a obsahuje příznak PUSH, jsou data předána další analýze bez zbytečného ukládání. Zpracování TCP paketů je implmentováno ve fukci `tcp_parse`.

### 6.2.2 Zpracování textového vstupu

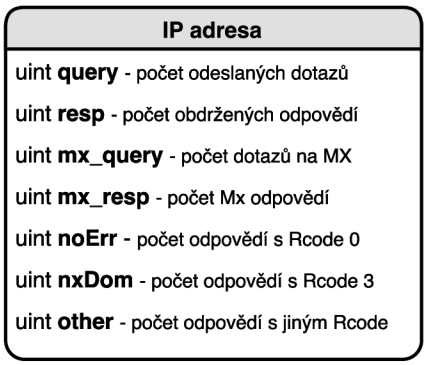
Textový vstup je načítán ve while cyklu po řádcích funkcí `fscanf`. Řádky vstupního souboru jsou ve formátu:
`<DST_IP>,<SRC_IP>,<BYTES>,<TIME_FIRST>,<TIME_LAST>,<PACKETS>,<DNS_ANSWERS>,`
`<DNS_PSIZE>,<DNS_QTYPE>,<DNS_RLENGTH>,<DNS_RCODE>,<DNS_NAME>`
Jednotlivé položky odpovídají výstupu Flowmon Exporteru. Aplikace automaticky přeskakuje první řádek souboru, kde očekává hlavičku sloupců.

## 6.3 Struktury hosta a jeho mapování

Statistické informace provozu jednotlivých hostů reprezentuje struktura `hostStat`, obr 6.2. Každý host je identifikován svou IP adresou a ke každé zachycené IP adrese je přiřazena právě jedna struktura `hostStat`. Vzájemné mapování IP adresy a příslušné struktury je implementováno pomocí standardní třídy `map` kombinující klíč a hodnotu. Třída `map` je implementována jako binární vyhledávací strom a zaručuje unikátnost klíče a jeho seřazení.

Klíčem je použita 32 bitová hodnota IP adresy a užitečnou hodnotou je ukazatel na strukturu statistik hosta: `std::map<uint32_t , hostStat *> hostMap`. Nevýhodou této varianty implementace je omezení pouze na IPv4 adresy, avšak provoz IPv6 tvořil pouze zanedbatelné procento celkového provozu testované sítě a proto je pro detekci jeho vynechání nepodstatné.

Pro práci s mapou hostů je implementována funkce `get_host`, která provede vyhledání podle zadaného klíče (IP adresy) a vrátí ukazatel na strukturu v případě shody, anebo v opačném případě vrátí ukazatel nově vytvořené struktury v mapě. Smazání mapy a uvolnění jejich zdrojů probíhá ve funkci `cleaner` po skončení detekce.




Obrázek 6.2: Struktura statistik hosta

## 6.4 Analýza DNS provozu

#### Analyzátor binárních DNS zpráv

Hlavičku DNS zprávu v binární podobě reprezentuje struktura `dns_header`. Detekce probíhá pouze na standardních dotazech s operačním kódem (OPCode) 0. Proto je nejdříve

proveden test položky OPCode v hlavičce DNS. Poté je testován příznak QR určující, zda se jedná o dotaz či odpověď. Podle Postup je zobrazen na následujícím algoritmu 1:

**Algoritmus 1:** Analýza DNS a inkrementace statistik

```
1:  if OPCode == 0 then
2:      if QR flag == 1 then
3:          host = get_host(destination IP address);
4:          host.resp++;
5:          if QType == 15 then
6:              host.mx_resp++;
7:          end if
8:          if RCode == 3 then
9:              host.NxDom++;
10:         else
11:             if RCode == 0 then
12:                 host.NoErr++;
13:             else
14:                 host.other++;
15:             end if
16:         end if
17:     else
18:         host = get_host(source IP address);
19:         host.query++;
20:         if QType == 15 then
21:             host.mx_query++;
22:         end if
23:     end if
24: else
25:     zpráva je zahozena
26: end if
```

### QType

Komplikací při analýze DNS zprávy je správně určit typu dotazu (položku *QType*), jelikož typ nemá své místo ve zprávě pevně dané, ale nachází se až za jménem s proměnlivou délkou. V případě dotazu je však zpráva složena pouze z hlavičky a sekce dotazu, proto je možné typ identifikovat záporným čtyř bytovým odsazením od konce zprávy:

```
Qtype = (uint16_t *) (dns_end - TYPE_OFFSET);
```

V případě DNS odpovědi nelze jednoduše určit konec sekce s dotazem, jelikož za sekcí s dotazem nasledují další sekce (odpověď, autority) taktéž s proměnlivou délkou. Proto je nutné projít jméno v sekci dotazu cyklem, alg. 2. Sekce dotazu se nachází za 12 bytovou DNS hlavičkou, proto je začátek jména identifikován 12B odsazením od začátku DNS zprávy. Jméno dotazu obsahuje na pozici teček počet znaků následující domény, celá doména končí nulovým bytem.

**Algoritmus 2:** Mapování 2B hodnoty QType v odpovědi

```
1:  int cnt = 0;
2:  dns_qst = ((u_char *) dns + 12);
3:  while dns_qst[cnt] != 0 do
4:      cnt = cnt + dns_qst[cnt] + 1;
5:  end while
6:  Qtype = (uint16_t *) &dns_qst[cnt + 1];
```

#### Analyzátor toků

Analyzátor toků zpracovává jednotlivé záznamy o tocích v textové podobě. Algoritmus zpracování je obdobný jako v případě binární podoby, alg. 1, avšak jednotlivé hodnoty statistik nejsou inkrementovány o 1, ale je přičítána hodnota `PACKETS`, udávající počet zpráv v rámci toku, např.:

```
host.resp +=  PACKETS;
```

Flowmon Exporter v základním režimu neumožňuje export příznaku QR, proto je pro rozlišení dotazu a odpovědi použita kombinace položek `DNS_ANSWERS`, `DNS_RCODE`, `DNS_RLENGTH` a `BYTES`. Náhrada příznaku vychází z předpokladu, že DNS dotaz má položky `DNS_ANSWERS`, `DNS_RCODE`, `DNS_RLENGTH` nulové, zároveň se kontroluje, zda paket není na dotaz příliš velký.

## 6.5 Implementace detektoru

Detekce probíhá po zpracování všech vstupních dat ve funkci `detect_map`. Iterátor projde cyklem všechny záznamy v mapě hostů `hostMap` a pro každý záznam vypočítá koeficienty. Zároveň se provoz jednotlivých hostů přičte do souhrné sumy ve struktuře `detectorSum`. Tato struktura uchovává souhrnné statistiky o průběhu detekce, které jsou po skončení výpočtů vypsány do souboru `report.log`.

$F_1$

Pro výpočet této metriky se pouze kontroluje, zda počet obdržených dotazů je vyšší než požadovaný minimální objem provozu (10 zpráv).

$F_2$

Důležité je ošetření nulových hodnot. Pokud host neobdrží žádné odpovědi s návratovým kódem 0 (NoError) a pouze dostává odpovědi NxDomain, je jeho koeficient nastaven na hodnotu 1, což z něj automaticky dělá potenciálního bota. Výjimkou je situace kdy host dostává pouze odpovědi s jiným návratovým kódem než je 0 nebo 3. V tomto případě je koeficientv $F_2$ nastaven na -1, jelikož tyto hodnoty můžou indikovat špatně nakonfigurovaného hosta.

**Algoritmus 3:** Výpočet koeficientu $F_2$

**Input**: MIN_HOST_RESP = 10

```
1:  if host.resp >= MIN_HOST_RESP then
2:      if host.noErr == 0 then
3:          if host.nxDom == 0 then
4:              F2 = -1.0;
5:          else
6:              F2 = 1.0;
7:          end if
8:      else
9:          F2 = host.nxDom / host.noerr;
10:     end if
11: end if
```

## 6.6 Výstup

Jak je patrné z obr 6.1, systém vygeneruje výstupní soubor v každé části detekce a umožňuje tak, vytvořit si obrázek o provozu na zkoumané síti a nastavit správné mezní hodnoty detekce. Výstupní soubory jsou umístěny do složky `out`. Součástí přiloženého CD jsou také skripty pro nástroj `gnuplot`, graficky zobrazující výsledky detekce.

**`report.log`** Celkové statistiky a výsledky detekce

**`allHosts.csv`** Zobrazuje detailní statistky všech zaznamenaných hostů

**`Outuput.csv`** Detailní statistiky hostů s požadovaným objemem provozu pro detekci

**`botF1.csv` a `botF2.csv`** Detailní statistiky hostů překračující mezní hodnotu daného koeficientu

První řádky souboru slouží jako hlavička sloupců. Jednotlivé řádky jsou vypsány pomocí funkce `fprintf` ve formátu:

`<HOST_IP>;<QUERY>;<RESP>;<NOERR>;<NXDOM>;<OTHER>;<F1>;<F2>;<MX_QUERY>;<MX_RESP>;`

# Kapitola 7

# Testování

Tato kapitola se zabývá analýzou výsledků detekce a testováním systému v průběhu vývoje. Testování bylo prováděno na modelovém `.pcap` souboru obsahující běžný DNS provozu na lokální síti a na datech zachycených exportéry v síti CESNET, poskytnuté vedoucím práce. Klíčovým krokem je určení mezních hodnot pro analýzu a ošetření anomálií při detekci.

| název | zdroj | délka | počet hostů |
|---|---|---|---|
| model.pcap | [2], [7] | 15 min | 46 |
| dnsdata_04252016_1105_1.csv | CESNET | 53 minut | 710000+ |

Tabulka 7.1: Testovací datasety

## 7.1 Modelová situace

Pro účely testování byl vytvořen modelový případ provozu na lokální síti s méně než 50 hosty. Dataset se skládá z 45 hostů[2] s normálním provozem a jedním hostem nakaženým virem Conficker [7], snažící se kontaktovat svůj C&C server s pomocí DGA algoritmu. Binární soubor `model.pcap` je součástí přiloženého CD.

Test by spuštěný s parametry: `-p model.pcap -e 0.8 -q 0.8`

Infikovaný host dle očekávání vykazuje vysoké hodnoty u obou koeficientů, příloha D. Provoz bota tvořil polovinu celkového provozu. Zvýšené hodnoty však vykazuje i jeden z normálních hostů, u kterého vysoké hodnoty způsobilo používání protokolu Bonjour od firmy Apple [23], pracující na podobném principu jak DGA. 4 hosté se zápornou hodnotou $F_2$, měli provoz složený pouze z reverzních dotazů PTR, na které obdrželi odpovědi s návratovým kódem 2. Ostatní hosté měli koeficienty téměř nulové nebo vůbec nedosáhli minimálního objemu provozu.Všechny stanice měli srovnatelný počet odeslaných dotazů a obdržených odpovědí, což koresponduje s výslednými hodnotami koeficientů. Na modelovém případě je patrný rozdíl mezi koeficienty normálních hostů a anomálií.

## 7.2 Reálný provoz

Jako testovací sada reálného provozu byl použit DNS provoz sítě CESNET ve formátu `.csv`, v čase od 10:19 do 11:12.

### 7.2.1 Analýza provozu

Testovací sadu tvořilo 10 milionů záznamů o tocích, ze kterých byli odfiltrován IPv6 provoz. IPv4 provoz měl následující charakteristiku (výpis ze souboru `report.log`):

```
Počet záznamů           :  9554866
Celkový počet DNS zpráv : 13082686
Celkový počet dotazů    :  6012283
Počet MX dotazů         :    42915
Celkový počet odpovědí  :  7070403
Počet NXDom odpovědí    :   371036
Počet odpovědí na MX    :    66444
Celkový počet hostů     :   718214
```

Histogramy provozu v síti pro jednotlivé hosty jsou uvedeny v příloze E. Z analýzy provozu je patrné, že většina hostů nedosáhne požadovaného objemu provozu 10 zpráv pro detekci a výrazně se tak sníží počet hostů v detekci. Střední hodnota počtu odeslaných dotazů je 8.37 a střední hodnota z počtu obdržených odpovědí je 9.84.

Dále z analýzy vyplývá, že provoz obsahuje vyšší počet odpovědí s MX typem v sekci dotazu než je samotných MX dotazů, avšak střední hodnoty jsou blízké 0. Při určení spam bota jsou proto porovnávány obě tyto hodnoty. Jelikož klesající trend histogramu dosahuje minima kolem hodnoty 20, je prahová hodnota pro detekci spam bota určena na 30 MX zpráv během jedné hodiny.

### 7.2.2 Testování detekce

První test byl spuštěn s parametry `-e 0.8 -q 0.8 -m 30`
Příloha F obsahuje histogramy pro jednotlivé koeficienty. Výpis ze souboru `report.log`:

```
Počet hostů s~min obdržených odpovědí  :  12529
Počet hostů s~min odeslaných dotazů    :  14358
Počet hostů zahrnutých v~detekci       :  20486
Počet hostů s~vysokým počtem Mx dotazů :    468
================================================
Koeficient F1
Prahová hodnota koeficientu F1     : 0.8000
Počet potenciálních botů           :    403
Počet počet potenciálních spammerů :     65
================================================
Koeficient F2
Prahová hodnota koeficientu F2     : 0.8000
Počet potenciálních botů           :   1761
Počet počet potenciálních spammerů :     60
================================================
Počet chybně nakonfigurovaných hostů (F2 == -1) : 102
```

### 7.2.3 Zaznamenané anomálie

V analýze výsledků testu se projevili některé anomálie:

1. vysoký počet hostů s koeficientem $F_2 = 1.0$
2. hosté s 0 odeslanými dotazy, přijímající odpovědi
3. hosté s 0 obdrženými odpověďmi, odesílající dotazy
4. velké rozdíly mezi počtem přijatých dotazů a odpovědí

První anomálie vyplývá z implementace výpočtu koeficientu $F_2$, alg. 3. Hodnotou 1 jsou označeni hosté, kteří neobdrželi žádné odpovědi s návratovým kódem 0 (NoErr), ale pouze odpovědi návratový kód 3 (NxDom). Typicky se jednalo o hosty zasílající reverzní dotazy na servery se službou DNS blacklistu, jako je např. *spamhaus.org*[1]. Příklad dotazu na blacklist databázi:

```
<DST_IP>      <SRC_IP>        <name>
....24.41    ....197.175     "88.99.x.y.sbl.spamhaus.org"
```

Druhá a třetí anomálie může značit útok na službu DNS, např. ampli – attack nebo DNS tunel, tyto hodnoty však můžou vykazovat i rekurzivní DNS servery nebo proxy DNS. Pro rozlišení, by však byla zapotřebí využít jiného detektoru provozu. Například host s nejvyšším počtem obdržených odpovědí (2.6M),byl pouze obětí distribuovaného útoku – jeho provoz tvořili pouze totožné odpovědi z různých IP adres ve stejný čas, se stejným jménem dotazu a typem dotazu *ANY*.

```
<DST_IP>      <SRC_IP>     <TIME>         <QTYPE>     <NAME>
....92.170   ....186.99    10:19:02.963    255      "cpsc.gov"
....92.170   ....186.99    10:19:02.963    255      "cpsc.gov"
      .
      .
      .
....92.170  ....57.157      10:19:02.963  255      "cpsc.gov"
```

Zároveň se tento útok projevil na položce `PACKETS` příslušného toku. Všechny toky s větší hodnotou `PACKETS` než 3, měli stejnou charakteristiku, projevující se jako útok.

Čtvrtá anomálie výrazně ovlivňuje výsledek koeficientu $F_1$. Nepoměr mezi odeslanými dotazy a obdrženými odpovědmi způsobuje vysoký počet falešných hlášení. Způsobit jí může stejně jako předchozím případě ampli – attack.

```
#Host         Query  Response  noErr  NxDom  other  F1         F2
....157.2     81     1118      1043   75     0      0.925926   0.071908
```

[1] https://www.spamhaus.org/

## 7.3 Optimalizace

Pro odstranění první výše zmíněné anomálie, byl na testovací data aplikován whitelist, odstraňující všechny dotazy směřující na 50. nejznámějších DNS blacklist databází, reverzní dotazy nebo například zprávy spojené s výše zmíněným protokolem Bonjour. Skript provádějící filtraci a seznam položek whitelistu je součástí přiloženého CD.

Aplikace whitelistu odstranila zhruba 7% záznamů, avšak počet potenciálních botů klesl v případě koeficientu $F_2$ o 85%. Výpis z ze souboru `report.log`:

```
Počet záznamů           :  8905384
Celkový počet DNS zpráv : 12433073
Celkový počet hostů     :   712107
============================================
Počet hostů zahrnutých v~detekci       :  18247
Počet hostů s~vysokým počtem Mx dotazů :    468
============================================
Koeficient F1
Prahová hodnota koeficientu F1     : 0.8000
Počet potenciálních botů           :    121
Počet počet potenciálních spammerů :     71
============================================
Koeficient F2
Prahová hodnota koeficientu F2     : 0.8000
Počet potenciálních botů           :    225
Počet počet potenciálních spammerů :     46
============================================
Počet chybně nakonfigurovaných hostů (F2 == -1) : 99
```

Dále byli do struktury `hostStat` přidány položky `dst_query` a `src_resp`. Proměnná `dst_query` uchovává počet dotazů adresovaných danému hostovi (cíl dotazu) a `src_resp` udává počet odpovědí, které host odeslal (zdroj odpovědi). Hosté, u kterých je počet dotazů nulový, nejsou cílem dotazu ani zdrojem odpovědi, ale přitom obdrželi více než minimální počet odpovědí, jsou vyřazeni z detekce a prohlášeni za DNS anomálii. Zároveň jsou statistiky jejich provozu vytisknuty do souboru `anomalyQ.csv`, který může sloužit například jako vstup detektoru ampli – attack nebo pro zpracování frekvenční analýzou doménového jména. Nevýhodou je nutnost mapovat struktury pro obě IP adresy DNS zprávy. Stejný postup je praktikován u hostů, kteří odesílají dotazy, ale počet odpovědí je nulový a nejsou zdrojem odpovědi ani cílem dotazu.

Po provedení optimalizace detektor ukázal zvýšení celkového počtu rozpoznaných hostů, avšak počet hostů v detekci klesl a detektor určil stejné spammery jako před optimalizací DNS anomálií. Zvýšení počtu celkového počtu hostů bylo způsobeno nutností zaznamenávat jak zdrojovou tak cílovou IP adresu.

```
Počet záznamů           :  8905384
Celkový počet hostů     :   795168
============================================
Počet hostů zahrnutých v~detekci       :  12524
Počet hostů s~vysokým počtem Mx dotazů :    413
============================================
```

```
Koeficient F1
Prahová hodnota koeficientu F1     : 0.8000
Počet potenciálních botů           :    121
Počet počet potenciálních spammerů :     71
=============================================
Koeficient F2
Prahová hodnota koeficientu F2     : 0.8000
Počet potenciálních botů           :    169
Počet počet potenciálních spammerů :     45
```

## 7.4 Analýza výsledků

V příloze G je uveden graf výsledků detekce po optimalizaci. Mezní hodnoty pro oba koeficienty jsou podle grafu určeny na 0.49 a hranice pro MX je 30 zpráv. Výsledkem detekce je následující hlášení

```
Koeficient F1
Prahová hodnota koeficientu F1     : 0.4900
Počet potenciálních botů           :    211
Počet počet potenciálních spammerů :     88
=============================================
Koeficient F2
Prahová hodnota koeficientu F2     : 0.4900
Počet potenciálních botů           :    241
Počet počet potenciálních spammerů :     48
```

Jako spammer bylo současně metrikou $F_1$ a $F_2$ označeno pouze 7 hostů. Mnoho hostů překračující hodnotu $F_1$ však doplatilo na nepoměr mezi celkovým počtem dotazů a odpovědí, proto je tato metrika neefektivní a pro další použití je vhodné zaměřit se pouze na metriku $F_2$.

### Ukázka provozu

Nejaktivnější spam bot obdržel 3375 odpovědí na dotaz typu MX.

```
<Host>       <TIME>       <Qtype>   <NAME>
....246.155 10:35:13.960  15        "info-care.pl"
....246.155 11:04:11.897  15        "bova-international.pl"
....246.155 10:35:23.795  15        "aaa23.pl"
```

# Kapitola 8

# Závěr

Zaměření této práce je detekce škodlivých stanic v provozu DNS. Úvodní kapitoly představují důležitost služby DNS a možnosti jejího zneužití. Navržený systém se zaměřuje na detekci anomálie způsobené nadměrným množstvím obdržených NXDomain odpovědí, která je typickým znakem pro botnety využívající DGA algoritmus.

Detekce botnetu je důležitá při prevenci rozesílání nevyžádané pošty, jelikož botnet je nejčastějším zdrojem spamu. Samotné rozesílání spamu se projevuje zvýšeným počtem MX dotazů.

Výsledkem práce je detektor použitelný především při správě a zabezpečení sítí, detekující anomálie zvýšeného objemu DNS NxDomain odpovědí a vysoký počet MX zpráv. Pro implementaci navrženého detektoru byl zvolený jazyk C++ a nástroj Gnuplot, pro grafické zobrazení výsledků detekce. Funkčnost implementovaného systému byla otestována na modelovém příkladu provozu ideální sítě a na provozu reálné páteřní sítě.

Pro určení škodlivých stanic byli v návrhu určeny 2 metriky – koeficienty $F_1$ a $F_2$. První metrika porovnávala počet odeslaných dotazů s počtem obdržených NxDomain odpovědí a metrika $F_2$ vyjadřovala poměr mezi počtem obdržených odpovědí s návratovým kódem 0 a 3. V modelovém případě obě metriky správně určili stanici používající DGA algoritmus, avšak v reálném provozu se ukázala metrika $F_1$ jako neefektivní s množstvím falešných hlášení.

Při testování na reálné síti se projevili některé anomálie detekce, např. nejvíce falešných hlášení bylo zaznamenáno u hostů využívající DNS blacklisty, tento problém se podařilo vyřešit aplikací whitelistu na použitou testovací sadu a odfiltrování dotazů na nejpoužívanější DNS blacklist domény. Implementace whitelistu přímo do detektoru je také možným rozšířením.

Další anomálií byli nulové hodnoty dotazů nebo odpovědí, které můžou značit útok na službu DNS. Detektor tyto stanice označuje a vypisuje do odděleného souboru a je tak možná jejich další analýza. Vhodné by bylo propojit tento výstup s frekvenční analýzou doménového jména, či detektorem distribuovaných zesilujících útoků.

Přínosem této práce je potvrzení funkčnosti navrženého systému, určení správných metrik a prahových hodnot pro detekci škodlivých stanic a prezentuje provoz, na který je detekce náchylná. Aplikaci je možné dále rozšiřovat a doplnit, anebo přidat podporu nových formátů vstupních dat. Zajímavým rozšířením se jeví modul pro systém Nemea pracující přímo se síťovými toky.

# Literatura


[1] Antonakakis, M.; Perdisci, R.; Nadji, Y.; aj.: From Throw-Away Traffic to Bots: Detecting the Rise of DGA-Based Malware. In *In Proc. of the 21th USENIX Security Symposium (Security'12)*, USENIX Association, Bellevue, Washington, USA: USENIX Association, srpen 2012, s. 48–61.

[2] Brad: *Traffic Analysis Exercises* [online]. 2016, [cit. 2016-03-1].
URL http://malware-traffic-analysis.net/

[3] Chen, C.-M.; Lin, H.-C.: Detecting botnet by anomalous traffic. *Journal of information security and applications*, , č. 21, 2015: s. 42–51, department of Information Management, National Sun Yat-sen University.

[4] Claise, B. E.: *Cisco Systems NetFlow Services Export Version 9* [online]. RFC 3954, říjen 2004.
URL http://www.rfc-editor.org/rfc/rfc3954.txt

[5] Claise, B. E.; Trammell, B.; Aitken, P.: *Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information* [online]. RFC 7011, září 2013.
URL http://www.rfc-editor.org/rfc/rfc7011.txt

[6] Doležal, J.: *Detekce škodlivých domén za pomoci analýzy pasivního DNS provozu*. Diplomová práce, FIT VUT v Brně, Brno, 2014.

[7] Garcia, S.; Grill, M.; Stiborek, H.; aj.: An empirical comparison of botnet detection methods. *Computers and Security Journal, Elsevier*, ročník 45, 2014: s. 100–123, http://dx.doi.org/10.1016/j.cose.2014.05.011, https://mcfp.felk.cvut.cz/publicDatasets/.

[8] Karim, A.; Salleh, R. B.; Shiraz, M.; aj.: Botnet detection techniques: review, future trends, and issues. *Journal of Zhejiang University SCIENCE C*, ročník 15, č. 11, 2014: s. 943–983, ISSN 1869-196X, doi:10.1631/jzus.C1300242.
URL http://dx.doi.org/10.1631/jzus.C1300242

[9] Khan, W. Z.; Khan, M. K.; Muhaya, F. T. B.; aj.: A Comprehensive Study of Email Spam Botnet Detection. *IEEE Communications Surveys Tutorials*, ročník 17, č. 4, Fourthquarter 2015: s. 2271–2295, ISSN 1553-877X, doi:10.1109/COMST.2015.2459015.

[10] Kováčik, M.: Detekce síťových anomálií a bezpečnostních incidentů s využitím DNS dat. 2014, pojednání k tématu disertační práce.

[11] Kováčik, M.: *Liberouter: DNS plugin* [online]. 2014 [cit. 2015-27-12].
URL https://www.liberouter.org/technologies/exporter/dns-plugin/

[12] Marinos, L.; Belmonte, A.; Rekleitis, E.: *ENISA Threat Landscape 2015* [online]. leden 2016, ENISA.
URL
https://www.enisa.europa.eu/publications/etl2015/at_download/fullReport

[13] Marko, P.; Vilhan, P.: Efficient Detection of Malicious Nodes Based on DNS and Statistical Methods. *10th IEEE Jubilee International Symposium on Applied Machine Intelligence and Informatics*, 2012: s. 227–230.

[14] Matoušek, P.: *Síťové aplikace a jejich architektura*. VUTIUM, 2014, ISBN 978-80-214-3766-1.

[15] Mockapetris, P.: *Domain names - Concepts and facilities* [online]. RFC 1034, 1987.
URL http://www.rfc-editor.org/rfc/rfc1034.txt

[16] Mockapetris, P.: *Domain names - Implementation and specification* [online]. RFC 1035, lisopad 1987.
URL http://www.rfc-editor.org/rfc/rfc1035.txt

[17] Ng, Y.: *DNS Lookup: How a Domain Name is Translated to an IP Address* [online]. [cit. 2016-5-2].
URL http:
//blog.catchpoint.com/2014/07/01/dns-lookup-domain-name-ip-address/

[18] Rincon, S. R.; Vaton, S.; Beugnard, A.; aj.: Semantics based analysis of botnet activity from heterogeneous data sources. In *IWCMC 2015 : 11th International Wireless Communications and Mobile Computing Conference – TRAC Workshop: Traffic Analysis and Characterization*, Dubrovnik, Croatia, srpen 2015, hal-01162734.

[19] Symantec Corporation.: *Internet Security Report 2015*, volume 20 [online],[cit. 1.2.2016].
URL https://www4.symantec.com/mktginfo/whitepaper/ISTR/21347931_
GA-internet-security-threat-report-volume-20-2015-appendices.pdf

[20] Trammell, B.; Boschi, E.: *Bidirectional Flow Export Using IP Flow Information Export (IPFIX)* [online]. RFC 5103, leden 2008.
URL http://www.rfc-editor.org/rfc/rfc5103.txt

[21] Villamarín-Salomón, R.; Brustoloni, J. C.: Identifying Botnets Using Anomaly Detection Techniques Applied to DNS Traffic. In *5th IEEE consumer communications and networking conference CCNC*, IEEE, 2008, s. 476–481.

[22] Vixie, P.; Eastlake, D.; Gudmundsson, O.; aj.: *Secret Key Transaction Authentication for DNS (TSIG)* [online]. RFC 2845, květen 2000.
URL https://www.ietf.org/rfc/rfc2845.txt

[23] Wikipedia: *Bonjour (software)* [online]. [cit. 2016-5-6].
URL https://en.wikipedia.org/wiki/Bonjour_(software)

# Přílohy

# Seznam příloh

A Obsah CD 35

B Metriky kódu a parametry pro spuštení 36

C Vývojový diagram detektoru 37

D Analýza modelu 38

E Analýza provozu reálné sítě 39

F Test 1 41

G Detekce po optimalizaci 42

# Příloha A

# Obsah CD

- src – adresář zdrojových souborů
- data – adresář obsahující testovací data
- out – prázdný adresář pro výstupní soubory detektoru
- script – adresář obsahující podpůrné skripty – whitelist, odstranění IPv6
- gnuplot – adresář obsahující skripty pro kreslení grafů
- xplote01 – Technická zpáva ve formátě pdf
- latex – Technická zpáva ve formátě tex
- README – pokyny pro spuštění aplikace

# Příloha B

# Metriky kódu a parametry pro spuštení

| | | |
|---|---|---|
| Počet souborů | : | 2 |
| Počet řádků kódu | : | 1500 |
| Velikost zdrojových souborů | : | 37 kB |
| Velikost binárního souboru | : | 96 kB |
| Požadavky | : | knihovna `libpcap` |
| Testováno na Fedora 64-bit | | |

**Parametry pro spuštění**

`-h`
: Vytiskne nápovědu

`-c <filename>`
: Specifikuje vstupní textový `.csv` soubor obsahující záznamy DNS toků.

`-p <filename>`
: Specifikuje vstupní binární `.pcap` soubor obsahující odchycené DNS pakety.

`-o <interface>`
: Specifikuje vstupní rozhraní pro online odchycení DNS provozu. Hodnota "any"pro všechny rozhraní.

`-m <value>`
: Specifikuje mezní hodnotu pro MX záznamy. Výchozí hodnota je 50.

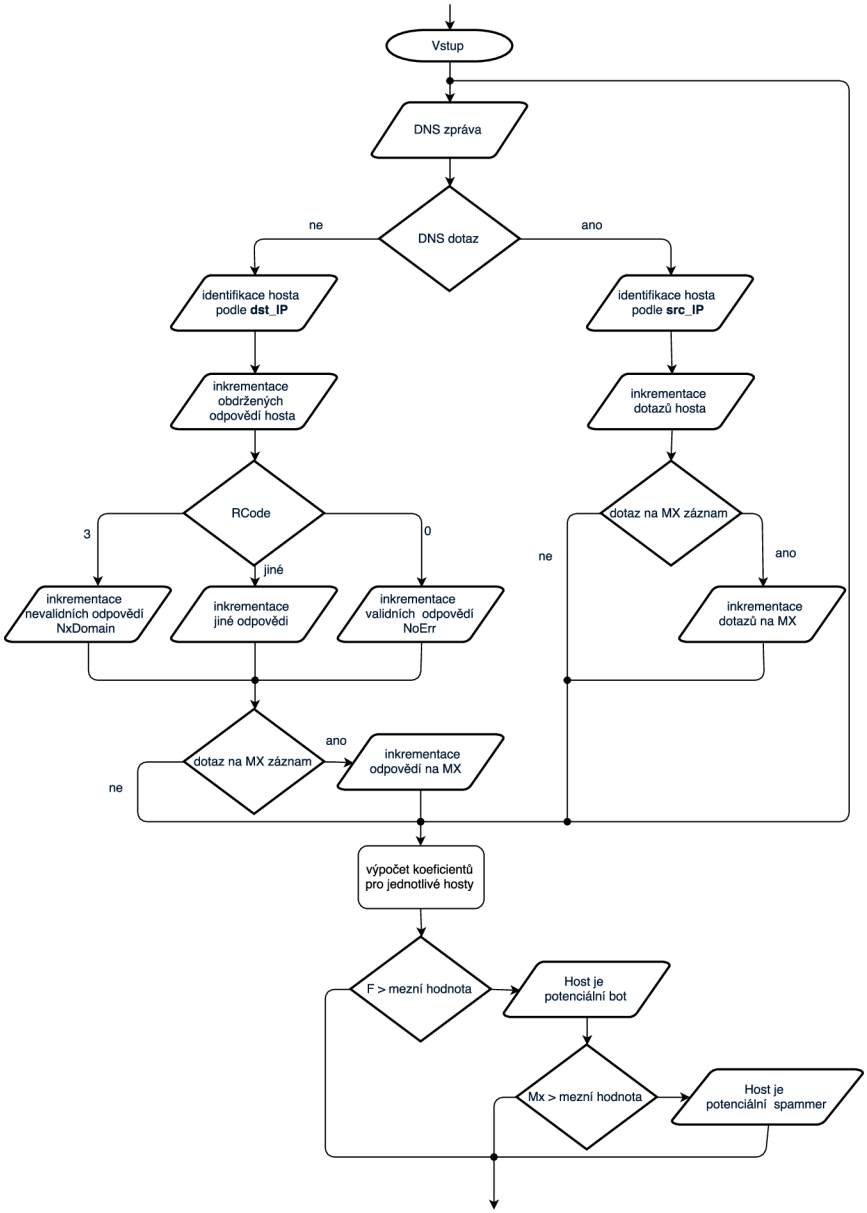
`-q <value>`
: Specifikuje mezní hodnotu pro koeficient $F_1$. Výchozí hodnota je 0.5.

`-e <value>`
: Specifikuje mezní hodnotu pro koeficient $F_2$. Výchozí hodnota je 0.5.

paramatery `-c`, `-p` a `-o` nelze kombinovat, musí však být zadán alespoň jeden z nich

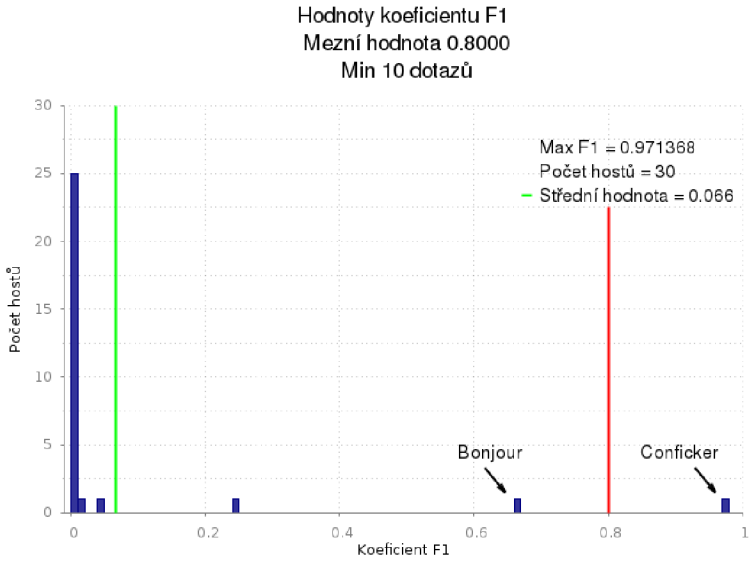
# Příloha C

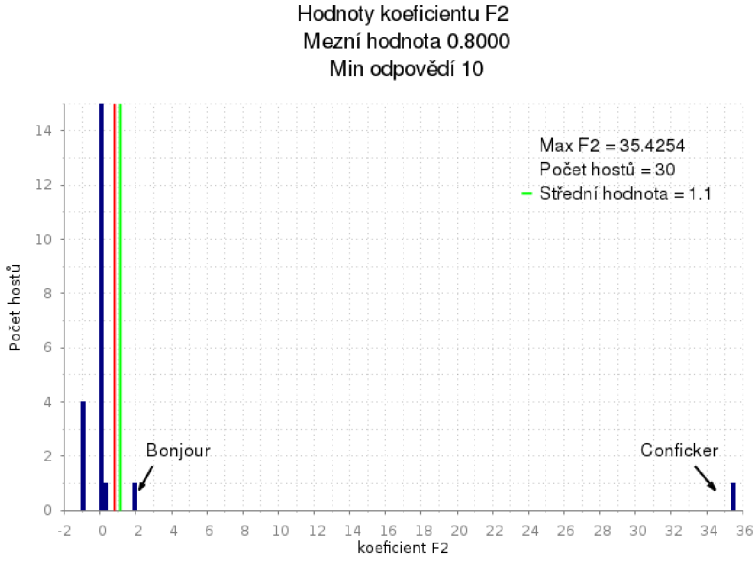
# Vývojový diagram detektoru



# Příloha D

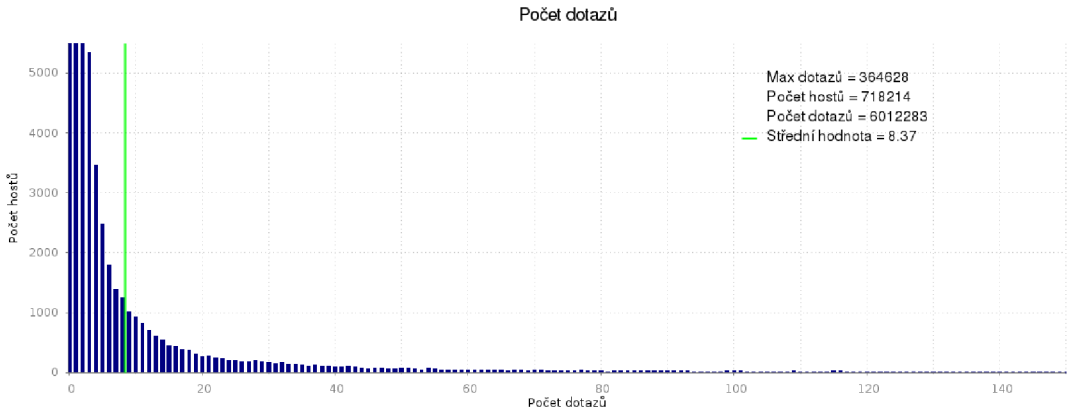
# Analýza modelu



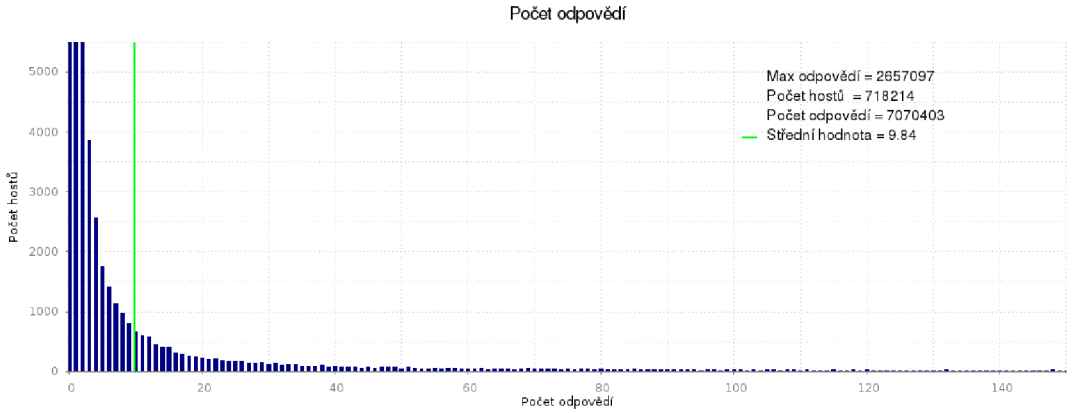



# Příloha E

# Analýza provozu reálné sítě

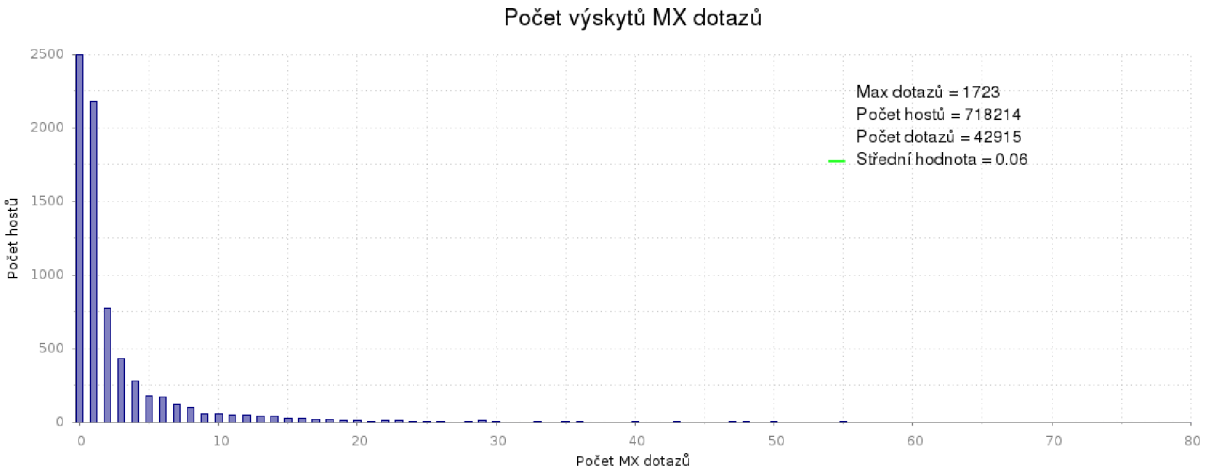



Obrázek E.1: Počet zachycených dotazů jednotlivých hostů




Obrázek E.2: Počet zachycených odpovědí jednotlivých hostů

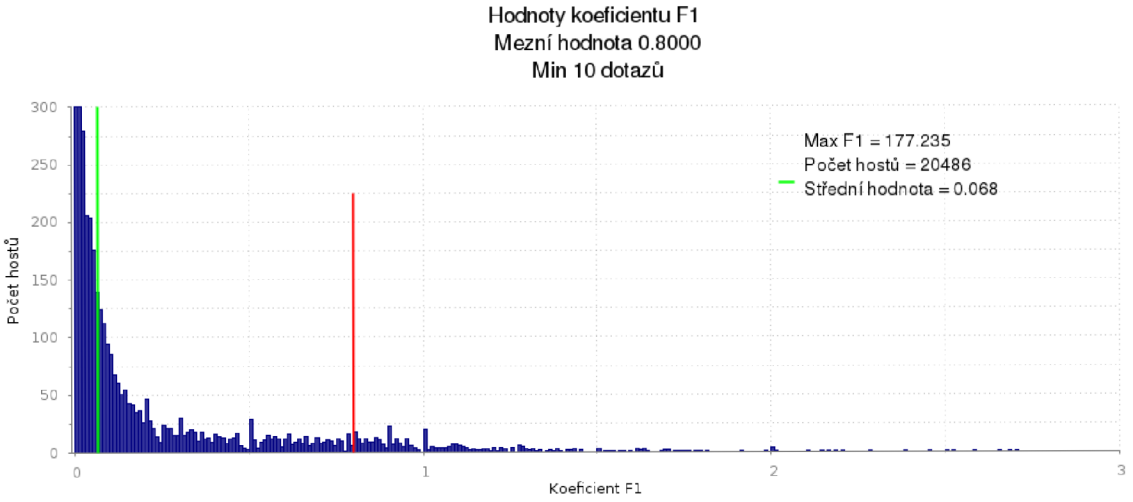
Obrázek E.3: Počet zachycených odpovědí na MX dotazy



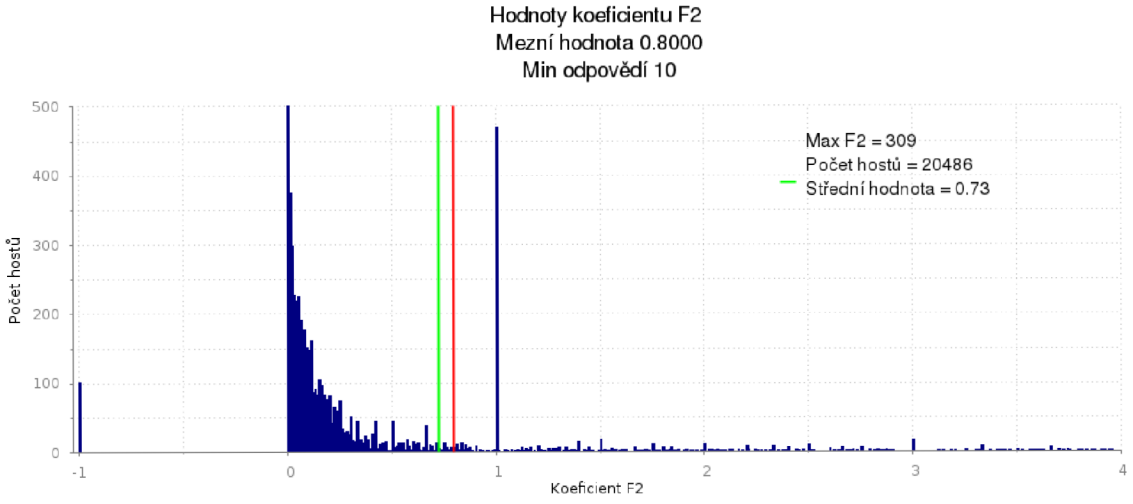

Obrázek E.4: Počet zachycených MX dotazů

# Příloha F

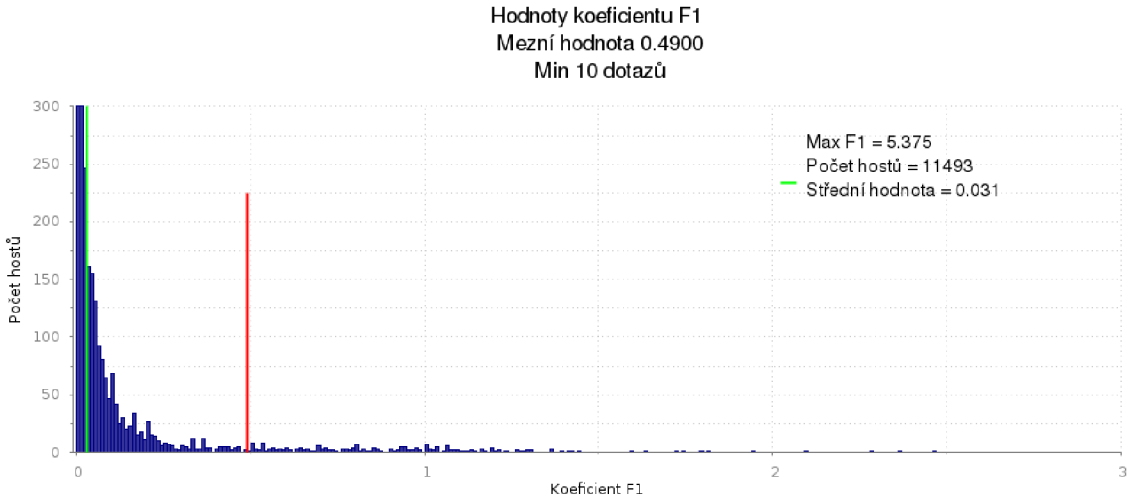
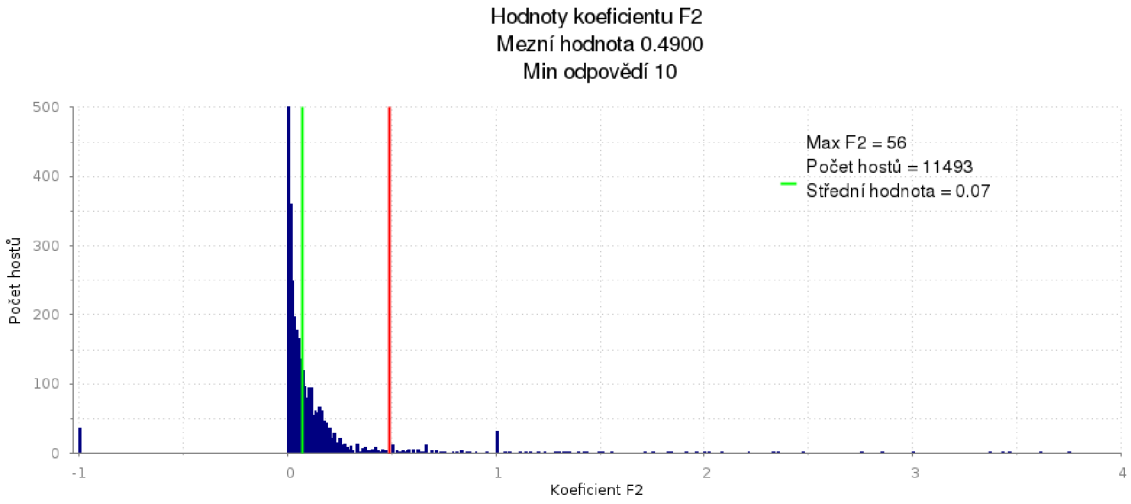
# Test 1




Obrázek F.1: Počet zachycených dotazů jednotlivých hostů




Obrázek F.2: Počet zachycených odpovědí jednotlivých hostů

# Příloha G

# Detekce po optimalizaci