UNIVERZITA PAVLA JOZEFA ŠAFÁRIKA V KOŠICIACH
PRÍRODOVEDECKÁ FAKULTA

APLIKAČNÝ FIREWALL PRE DNS SERVER

2016 **Štefan PORHINČÁK**

UNIVERZITA PAVLA JOZEFA ŠAFÁRIKA V KOŠICIACH
PRÍRODOVEDECKÁ FAKULTA

APLIKAČNÝ FIREWALL PRE DNS SERVER

BAKALÁRSKA PRÁCA

Študijný program:	informatika
Pracovisko (katedra/ústav):	Ústav informatiky
Vedúci záverečnej práce:	RNDr. JUDr. Pavol Sokol, PhD.
Konzultant:	Mgr. Lukáš Hlavička

Košice 2016 **Štefan Porhinčák**

Univerzita P. J. Šafárika v Košiciach
Prírodovedecká fakulta

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta:	Štefan Porhinčák
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: Aplikačný firewall pre DNS server

Názov EN: Application firewall for DNS server

Cieľ: (1) Analyzovať bezpečnostné hrozby DNS protokolu.
(2) Analyzovať a porovnať možnosti ochrany voči útokom na DNS protokol.
(3) Navrhnúť a implementovať aplikačný firewall pre DNS server.

Literatúra:
(1) Aitchison, Ron. Pro Dns and BIND 10. Apress, 2011.
(2) Dostálek, Libor, and Alena Kabelová. DNS in Action: A Detailed and Practical Guide to DNS
Implementation, Configuration, and Administration. Packt Pub Limited, 2006.
(3) Rash, Michael. Linux firewalls: attack detection and response with iptables, psad, and fwsnort. No Starch Pr, 2007.
(4) Gheorghe, Lucian. Designing and Implementing Linux Firewalls with QoS using netfilter, iproute2, NAT and l7-filter. Packt Pub Limited, 2006.
(5) Suehring, Steve, and Robert Ziegler. Linux Firewalls (Novell Press). Novell Press, 2005.

Anotácia: Cieľom práce je popísať bezpečnostné aspekty DNS protokolu, najmä bezpečnostné hrozby, ktoré ho ohrozujú. Práca sa venuje možnostiam obrany voči rôznym útokom na DNS protokol a rozoberá a porovnáva tieto možnosti. Cieľom práce je tiež implementovať aplikačný firewall na ochranu DNS servera.

Kľúčové slová: DNS, firewall, iptables, bezpečnostná hrozba

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

Dátum schválenia: 22.02.2016

prof. RNDr. Viliam Geffert, DrSc.
riaditeľ ústavu

Abstrakt v štátnom jazyku

Systém doménových mien (DNS, Domain name system) je služba, ktorá udržiava zoznam IP adries a ich prislúchajúcich doménových mien. Práca sa zameriava na bezpečnostné aspekty tejto sieťovej služby. Rozoberá najmä analýzu známych bezpečnostných hrozieb súvisiacich s touto službou, popisuje zvolené hrozby a útoky a napokon analyzuje a porovnáva dostupné spôsoby a metódy ochrany voči nim. Súčasťou práce je aj návrh a implementácia DNS firewallu, ktorý chráni DNS server voči zvoleným útokom. Pri výbere útokov sa vychádzalo z analýz a potrieb slovenskej špecializovanej jednotky pre riešenie počítačových incidentov v Slovenskej republike (CSIRT.SK). Navrhovaný firewall rieši problém filtrácie DNS komunikácie na strane poskytovateľa internetového pripojenia. Tiež sa zameriava na ochranu DNS servera proti neautorizovaným zónovým transferom pomocou filtrovania jednotlivých druhov DNS požiadaviek podľa zdrojovej IP adresy.

Abstrakt v cudzom jazyku

Domain name system (DNS) is a service providing database of translations from domain names to IP addresses and back. Our work is focusing on security aspects of this network service. It provides analysis of known security threats for this service, describes process of selected attacks and we are analysing available means and methods for protection against chosen threats. Part of our work is also proposal and implementation of application DNS firewall, which should protect DNS server against chosen attacks. Selection of threats and attacks was made based on analysis and needs of Computer Security Incident Response Team Slovakia (CSIRT.SK). Our proposed firewall improves possibilities of filtration of DNS communication via selected rules on side of ISP. Also, it focuses on protection of DNS server against not authorized zone transfers via filtering DNS requests by type and by source IP address.

Obsah

Zoznam skratiek a značiek

IPv4 Internet protokol verzie 4

IPv6 Internet protokol verzie 6

DNS Systém doménových mien (Domain name system)

RFC Request For Comments

TCP Protokol riadenia prenosu (Transmission Control Protocol)

UDP Používateľský datagramový protokol (User datagram protocol)

NVD Národná databáza hrozieb (National Vulnerability Database)

CVSS Štandardný systém hodnotenia zraniteľností (Common vulnerability scoring system)

Úvod

Dennodenne bezpečnostné tímy na celom svete zaznamenávajú veľké množstvo bezpečnostných incidentov. Tieto bezpečnostné hrozby vo veľkej miere využívajú aj protokol na správu doménových mien (DNS protokol), ktorý v súčasnej dobe využíva takmer každé zariadenie v počítačovej sieti Internet. Hlavným cieľom tejto záverečnej práce je vytvoriť bezpečnostný mechanizmus – firewall, pre ochranu niektorých špecifických prvkov protokolu DNS.

Hrozby, pre ktoré vytvárame moduly do firewallu boli zvolené na základe potrieb slovenskej špecializovanej jednotky pre riešenie počítačových incidentov v Slovenskej republike (CSIRT.SK). Z dôvodu pragmatickosti a udržateľnosti pre každú hrozbu vytvárame samostatný modul.

Táto záverečná práca je zložená zo štyroch kapitol. Úvodná kapitola práce predstavuje úvod do protokolu DNS. Rozoberá úvod do systému doménových mien, služby DNS, stručne popisuje protokol DNS, typy serverov a známe implementácie DNS protokolu.

V druhej časti našej práce sa zaoberáme analýzou známych bezpečnostných hrozieb a útokov súvisiacich so službou DNS a klasifikácii týchto hrozieb podľa ich cieľov. V tretej kapitole tejto práce sa venujeme analýzou a porovnaním dostupných spôsobov ochrany a návrhu nových možných metód obrany.

V štvrtej časti práce sa venujeme návrhu firewallu pre DNS server chrániaceho voči skupine zvolených útokov. Tento server je nesaditeľný na strane poskytovateľa pripojenia pre DNS servery. Súčasne analyzujeme možnú úspešnosť ochrany voči jednotlivým skupinám útokov v závislosti od intenzity a miery distribuovanosti jednotlivých útokov. Súčasťou kapitoly je aj implementácia firewallu, ktorý chráni voči zvoleným typom útokov a rieši špecifický problém filtrácie typov DNS komunikácie na strane poskytovateľa pripojenia. Nami vytvorené riešenie takisto otestujeme na testovacej sieti Študentských domovoch a jedálňach UPJŠ v Košiciach, kde odmeriame dopad nášho riešenia na výkonnosť DNS serveru.

1 Správa doménových mien

Táto kapitola predstavuje úvod do problematiky tejto záverečnej práce a zaoberá sa protokolom pre správu doménových mien. Bližšie rozoberá komunikáciu v rámci tohto protokolu a štruktúru prenášaných údajov. V rámci kapitoly sa venujeme aj implementáciám tohto protokolu.

1.1 Úvod do správy doménových mien

Správa doménových mien (z anglického **Domain Name System - DNS**) [1] je celosvetová, distribuovaná, hierarchicky organizovaná databáza doménových mien. Dnes je DNS prakticky základom pri používaní siete internet, mnohé celosvetovo používané protokoly ako napríklad SMTP (Simple mail transfer protocol), HTTP (Hypertext transfer protocol), HTTPS (Hypertext transfer protocol secure), XMPP (Extensible Messaging and Presence Protocol) a ďalšie sa spoliehajú na DNS, ktorý využívajú na distribúciu informácii potrebných pre ich správnu funkcionalitu. Aj napriek existencii projektov ako P2PDNS [2] a pod., aktuálne nám nie je známy protokol, ktorý by bol schopný nahradiť funkcionalitu DNS v dostatočnej miere pre využívanie siete internet tak, ako nám je známe dnes.

Autorom prvého návrhu a implementácie je **Paul Mockapetris** (na podnet amerického vedca menom John Postel z University of California) v roku **1983**. Protokol DNS bol navrhnutý a vytvorený ako náhrada vtedajšieho spôsobu prekladu ľudsky rozpoznateľných mien počítačov na IP adresy. Predošlý systém zakladal na centrálne udržiavanom zozname počítačov v sieti **ARPANET** (**Advanced Research Projects Agency Network**), udržiavanom Standfordským výskumným inštitútom. Celý zoznam mien počítačov bol uložený v súbore HOSTS.TXT, ktorý bol manuálne distribuovaný na každý počítač pripojený do siete. Tento systém bol však nedostatočne pružný pre potreby rastúcej siete ARPANET, z ktorej sa neskôr vyvinula sieť Internet [3]. Postupom času boli možnosti protokolu DNS rozširované a prispôsobované aktuálnym požiadavkám.

1.2 Popis DNS protokolu

V nasledujúcej kapitole sa oboznámime s protokolom DNS, a to konkrétne s typmi záznamov, ktoré DNS poskytuje, infraštruktúrou, ktorá slúži na ukladanie. Súčasťou je tiež oboznámenie so spôsobom prenosu týchto záznamov, spôsobom komunikácie medzi klientom a serverom a tiež štruktúrou DNS rámca.

1.2.1 DNS záznamy a infraštruktúra

Záznamy v databáze DNS majú rôzne účely. Slúžia napríklad na priradenie doménového mena k IP adrese, ako kanonické záznamy, záznamy o serveroch zodpovedajúcich za doručovanie emailu pre danú doménu (MX), záznamy o službách poskytovaných v doméne (SRV), ale aj napríklad textové záznamy (TXT). Textové záznamy samotné môžu mať rôzne využitie [4].

Návrh DNS protokolu počítal s distribuovanou povahou databázy, ktorá je hierarchicky usporiadaná. Informácie o *n*+1 zóne v hľadanom doménovom mene je možné získať od DNS servera v *n*-tej zóne, pričom za 0-tú zónu považujeme koreňové servery. Týchto serverov je celosvetovo 13 (13 IP adries pri IPv4 protokole). Jednotlivé servery sú však mnohonásobne duplikované a umiestnené na viacerých miestach.

1.2.2 Štruktúra DNS rámca

Pre potreby bakalárskej práce popisujeme staršiu verziu DNS hlavičky, ktorá je rozširovaná organizáciou **IANA** (Internet assigned numbers authority). V tejto verzii sú vynechané časti zaoberajúce sa rozšírením **DNSSEC** (Domain Name System Security Extensions), ktorému sa v práci špeciálne **nevenujeme**. Štruktúru hlavičky, ktorú je potrebné vysvetliť pre účely nasledujúcich kapitol, nám ilustruje obrázok 1. Na danom obrázku vidíme označenie jednotlivých bitov DNS hlavičky [4]:

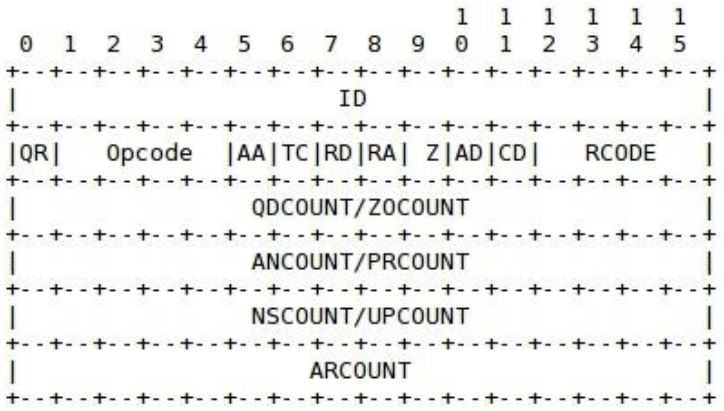
- **ID** je 16 bitové číslo, potrebné na identifikáciu a spárovanie otázky a odpovede u klienta;
- **QR** je rozhoduje, či je paket otázkou alebo odpoveďou;
- **OpCode** určuje typ akcie, ktorá je požadovaná paketom. Aktuálne je používaných 5 kódov, a to:
    - **0** – Otázka;
    - **1** - Reverzná otázka;
    - **2** - Požiadavka na status;
    - **4** - Notifikácia a
    - **5** - Aktualizácia.
    - Operačné kódy 3, 6 až 15 sú ponechané pre budúce použitie.

• Bity **AA, TC, RD, RA, Z, AD, CD**. Týchto 7 bitov určujú jednotlivé príznaky paketu:

- **AA** určuje, či je paket autoritatívnou odpoveďou;
- **TC** indikuje, že odpoveď bola skrátená, kvôli prekročeniu maximálnej veľkosti odpovede;
- **RD** znamená vyžadovanie rekurzívneho vyhľadania odpovede na strane servera. Podpora tohto bitu nie je nutná;
- **RA** označuje, že server poskytuje možnosť rekurzívneho vyhľadania odpovede pre klienta;
- **Z** je rezervované pre budúce použitie;
- **AD** pri použití DNSSEC označuje, že odpoveď bola autentifikovaná serverom;
- **CD** oznamuje, že overovanie odpovede pomocou DNSSEC má byť vynechané v prípade, že DNS klient toto overovanie podporuje.

• **RCODE** predstavuje kód, ktorý určuje stav odpovede. Pre naše potreby uvažujeme 6 stavov:

1. **stav** je označovaný taktiež NoError. Indikuje žiaden chybový stav, tj. preklad prebehol bez problémov;
2. **stav** indikuje chybu formátu otázky;
3. **stav** oznamuje chybu servera;
4. **stav** oznamuje neexistenciu záznamu pri odpovedi od autoritatívneho servera;
5. **stav** indikuje, že daná operácia nebola implementovaná;
6. **stav** oznamuje, že vykonanie operácie bolo odmietnuté kvôli pravidlám na strane servera.



Obr. 1 Štruktúra hlavičky DNS paketu [4]

1.2.3 Komunikácia

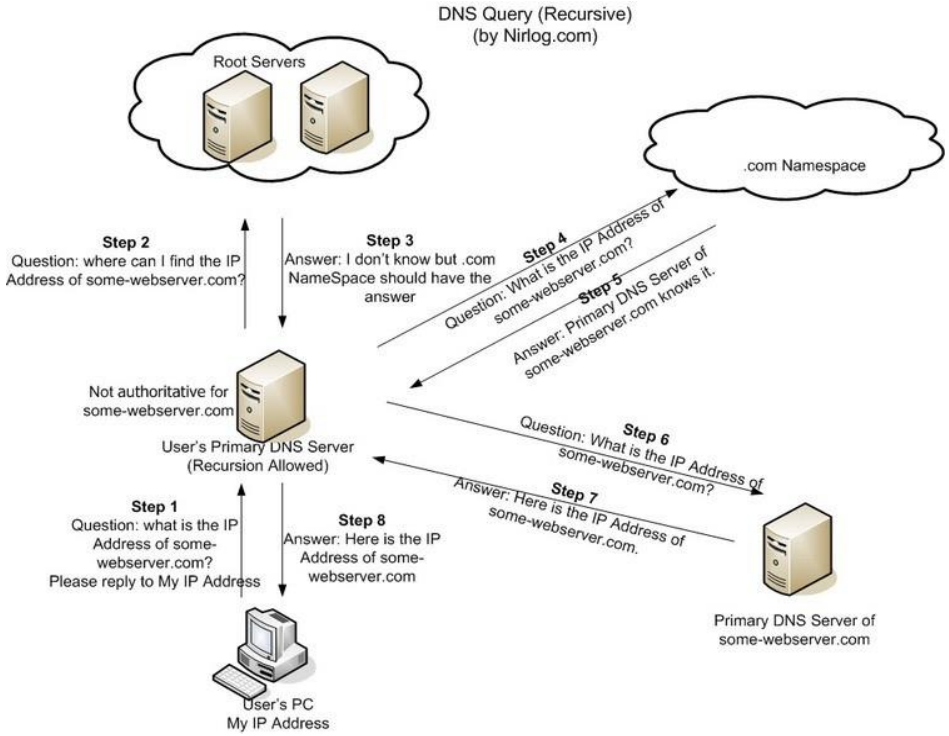
Požiadavka na získanie údajov (preklad doménového mena na IP adresu, informácie o mailovom serveri alebo iný záznam) z DNS servera prebieha zvyčajne nasledovným spôsobom:

1. Klient (aplikácia, operačný systém alebo iné) vytvorí DNS paket skladajúci sa z hlavičky a tela, ktoré obsahuje informácie o tom, aké údaje si klient žiada.
2. DNS paket sa odošle prostredníctvom protokolu UDP alebo TCP, ak je to potrebné na jeden z DNS serverov, ktoré sú nastavené v systéme alebo na jeden z koreňových serverov. Niektorí klienti dovoľujú špecifikovať, na ktorý server sa má odoslať požiadavka.
3. DNS server môže odpovedať na požiadavku odpoveďou, ktorá obsahuje informácie o požadovaných údajoch, alebo o tom, že tieto údaje neposkytuje, alebo odmieta poskytnúť. K odpovedi môže priložiť dodatočné informácie o tom, ktorý server je zodpovedný za zónu, v ktorej sa nachádzajú požadované informácie.
4. Ak klient získal informácie, ktoré potreboval, algoritmus končí.
5. Server, od ktorého požadujeme informácie, sa zmení buď na jeden z koreňových serverov, ktorých IP adresy sú verejne známe alebo na server, ktorého adresu sme dostali v kroku 3 a pokračuje sa znovu krokom 2.

Tento postup je znázornený na obrázku č.2, kde sa klient snaží získať IP adresu pre „some-webserver.com“ a na sieti sa nachádza rekurzívny DNS server. V horeuvedenom postupe sa klientom stáva tento rekurzívny server a postupne sa pýta na informácie koreňového servera, potom tld servera pre doménu .com a následne doménového servera pre „some-webserver.com“.

Na získanie prekladu doménového mena (alebo iného typu záznamu) sú s ohľadom na rýchlosť získania informácie používaný UDP protokol. Avšak za určitých okolností môže byť použitý aj TCP protokol. Dôvodom pre použitie TCP protokolu môže byť napríklad prítomnosť truncate bytu (bytu, ktorý označuje, že správa nebola doručená celá, t.j. jej časť

bola odseknutá – truncated), zónový transfer alebo vyžiadanie použitia TCP protokolu zo strany klienta.



Obr. 2 Príklad prekladu doménového mena [5]

1.2.4 Zónové transfery

Keďže DNS protokol pre zabezpečenie redundancie a dostupnosti poskytovanej služby vyžaduje definovanie aspoň dvoch serverov poskytujúcich záznamy o zóne. V tomto smere je žiadúce, aby tieto údaje boli rovnaké na všetkých serveroch, bolo potrebné zaviesť spôsob synchronizácie údajov. **Zónový transfer** je typ DNS požiadavky, využívaný na synchronizáciu sekundárnych DNS serverov s primárnym. Synchronizácia prebieha pomocou kopírovania celej zóny pri type prenosu **AXFR** alebo len časti zóny, ktorá bola zmenená za poslednú dobu v prípade inkrementálneho typu prenosu **IXFR.**

1.3 Typy DNS serverov

1.3.1 Autoritatívny DNS server

Autoritatívny DNS server je DNS server, ktorý je nakonfigurovaný tak, že poskytuje len autoritatívne odpovede na zóny, ktoré boli nastavené administrátorom.

Štandardne by sa malo použiť viacero serverov, najmä v dôsledku zabezpečenia dostupnosti v prípade výpadku jedného zo serverov. Zvyčajne ide o jeden **hlavný DNS server** (**master DNS server**), na ktorom sú aplikované zmeny zón. Tieto zóny majú byť poskytované jedným alebo viacerými **podriadenými DNS servermi** (**slave DNS server**). Tieto servery sú automaticky synchronizované podľa hlavného DNS Servera.

Po zaregistrovaní domény u správcu domén druhej úrovne (napr. doména example.com je doména druhého rádu a doména com je doména prvého rádu) je vyžadované dodať adresy dvoch serverov, ktoré budú slúžiť ako menné servery pre registrovanú zónu. Ak berieme v úvahu to, že sa odporúča aby master server nebol používaný priamo klientami a aby klientov obsluhovali len slave servery, znamená to že by sme mali mať tri DNS servery (jeden master server a dva slave servery).

1.3.2 Rekurzívne a cache DNS server

Rekurzívny DNS server a cache DNS server sú servery, ktoré sú nakonfigurované tak, aby slúžili na zrýchlenie DNS infraštruktúry. Inými slovami, ich účelom je zníženie komunikácie klientov priamo s autoritatívnymi servermi a zníženie doby, za akú dostane klient odpoveď.

Pri požiadavke na **rekurzívny server** sa tento server správa ako klient. Sám začne získavať odpoveď pre klienta podobne, ako keby túto odpoveď získaval klient. Tento server sa využíva hlavne pre zablokovanie priamej komunikácie klientov v počítačovej sieti, v ktorej to nie je žiadané. **Cache DNS server** si odpovede, ktoré získaval, udržiava vo svojej internej pamäti štandardne po dobu, ktorá je určená v **TTL záznamoch**. TTL (Time to live) záznamy predstavujú údaj o tom, ako dlho môže byť platný záznam uložený v cache pamäti.

Vyššie uvedené servery sú zvyčajne nasadené v lokálnych počítačových sieťach, najbližšie ku klientom. Zvyčajne sa využívajú napríklad vo WiFi smerovačoch alebo u poskytovateľov internetového pripojenia.

Z teoretického pohľadu pre fungovanie DNS protokolu v sieti Internet postačujú len autoritatívne servery. V tomto prípade by sa ale každá požiadavka musela začať požiadavkou na koreňovú zónu DNS, čo by nebolo efektívne.

1.4 Implementácie DNS protokolu

V súčasnej dobe existuje pomerne veľké množstvo implementácii protokolu DNS. V nasledujúcich podkapitolách sa bližšie zameriame na tieto najrozšírenejšie implementácie:

- BIND;
- Knot DNS;
- DNSmasq a
- Microsoft DNS.

1.4.1 BIND

BIND (**Berkeley Internet Doman Name**) [6] je aktuálne celosvetovo najpoužívanejší, open-source implementujúci DNS server. Je považovaný za referenčnú implementáciu DNS protokolu. V dobe písania tejto práce je dostupná stabilná verzia 9.9.5-W1 a vývojová verzia 9.10.0rc1. Obe verzie sú použité v rámci testovania nášho riešenia. Inštitúcia **ISC (Internet Systems Consortium)** vyvíjala aj verziu 10 DNS servera BIND, ale vývoj bol ukončený v Apríli 2014 kvôli nedostatku zdrojov na vývoj.

1.4.2 Knot DNS

Knot DNS server [7] je vysoko výkonný DNS server špecializovaný na poskytovanie len autoritatívnych odpovedí. Je vyvíjaný správcom českej domény, organizáciou CZ.NIC [8]. Táto implementácia DNS servera je špecificky vyvíjaná na splnenie požiadaviek kladených na koreňové menné servery, určený pre použitie v prostredí vyžadujúcom veľmi vysokú rýchlosť odpovede. Podporovanými platformami sú všetky široko rozšírené UNIX-like operačné systémy a architektúry procesorov X86 a X64.

1.4.3 Dnsmasq

Dnsmasq [9] predstavuje o open-source, minimalistickú implementáciu DNS a DHCP servera, často používanú kvôli svojej nenáročnosti a nízkym nákladom. Podporuje väčšinu z funkcionality, ktorá je poskytovaná väčšími projektami. Tento DNS server sa používa napríklad v projekte OpenWRT, čo je distribúcia Linuxu, orientovaná na využitie v routeroch a podobných zariadeniach, ale aj v ďalších projektoch.

1.4.4 Microsoft DNS

Microsoft DNS [10] môže byť konfigurovaný ako autoritatívny, rekurzívny alebo hybridný DNS server. Je integrovaný s Active Directory, čo ho robí štandardne použitým serverom v firemných sieťach. Dovoľuje aj vytváranie zón podľa štandardných DNS zónových súborov. Tento DNS server podporuje DNSSEC od verzie Windows Serveru 2012, správu kľúčov a ďalšie funkcionality spojené s prácou v Active Directory.

2 Bezpečnostné hrozby voči DNS protokolu a jeho implementáciám

V tejto kapitole sa venujeme bezpečnostným aspektom DNS protokolu z pohľadu existujúcich hrozieb tohto protokolu a jeho implementácií. Kapitolu rozdeľujeme na podkapitoly z hľadiska CIA (dôvernosť – integrita - dostupnosť) modelu pre riadenie informačnej bezpečnosti v spoločnosti [11]. Jednotlivé kategórie bezpečnostných hrozieb sú doplnené o konkrétne zraniteľnosti.

2.1 Hrozby a útoky voči dôvernosti DNS protokolu a jeho implementáciám

Skupina hrozieb voči dôvernosti DNS protokolu zneužíva hlavne chýbajúce overovanie autenticity odpovede a možnosti sfalšovania odpovede útočníkom. Pri návrhu DNS protokolu v roku 1987 ešte neexistoval internet ako ho poznáme dnes a taktiež sa nepredpokladalo že DNS protokol bude zneužívaný.

2.1.1 DNS footprinting

Footprinting ako typ útoku, ktorého cieľom je získanie "digitálneho odtlačku“ obete je častokrát považovaný za predchodcu alebo prvé štádium útoku. Útočník sa snaží získať detailný obraz o infraštruktúre a stave zabezpečenia obete z častí verejne dostupných informácii, pričom pri tomto útoku nenarúša bezpečnostný perimeter. Častokrát sú využívané verejne dostupné nástroje a techniky, ktorých používanie je v mnohých prípadoch považované za nie nebezpečné [12].

DNS footprinting je typ útoku, ktorý na získanie informácii o obeti, využíva údaje, ktoré sú uverejnené v databáze DNS. V mnohých prípadoch sú (napríklad kvôli nesprávnej konfigurácii) zóny pre intranet a internet zlúčené do jednej zóny. V protokole DNS nie je implementovaná bezpečnostná politika obmedzujúca získanie informácii o DNS záznamoch pre intranet na základe niečoho iného, ako je IP adresa odosielateľa požiadavku. Teda, ak je DNS server nakonfigurovaný nesprávne a je možné vykonávať požiadavky na zónu ktorá bola pôvodne určená pre ukladanie informácii o lokálnej sieti, DNS protokol nevie následnému zneužitiu zabrániť. Nemenej častým javom je skutočnosť, že vo verejnej časti

DNS sú uvedené údaje, ktoré v kombinácii s ďalšími údaje dokážu odhaliť slabé miesta obete, a teda poskytnúť útočníkovi výhodu pri útoku [13, 14].

2.1.2 Hrozby zónových transferov

Zónové transfery ako metóda získania celého obrazu DNS zóny môžu byť pri nesprávnej konfigurácii alebo chybe v implementácii servera zneužité. Štandardne sa zónové transfery používajú na synchronizáciu DNS serverov, a teda po ukončení zónového transferu sa v databáze zón sekundárneho DNS servera bude nachádzať celá zóna, ktorá bola synchronizovaná. Útok na zónový transfer nemusí byť známkou prelomenia zabezpečenia protokolu. Môže ísť o zneužitie chybnej konfigurácie servera. Výsledkom zneužitia bude pre útočníka získanie obrazu o celej DNS zóne, a teda počtu zariadení, ich určenie (podľa ich názvov, ak sú zmysluplné) a následne zoznam potencionálnych ďalších cieľov.

2.2 Hrozby a útoky voči integrite DNS protokolu a jeho implementáciám

Do kategórie hrozieb voči integrite DNS protokolu môžeme zaradiť aj otrávenie DNS cache, špeciálnu verziu podvrhnutia DNS odpovede, o ktorej píšeme v predchádzajúcej kapitole.

2.2.1 Podvrhnutie DNS odpovede

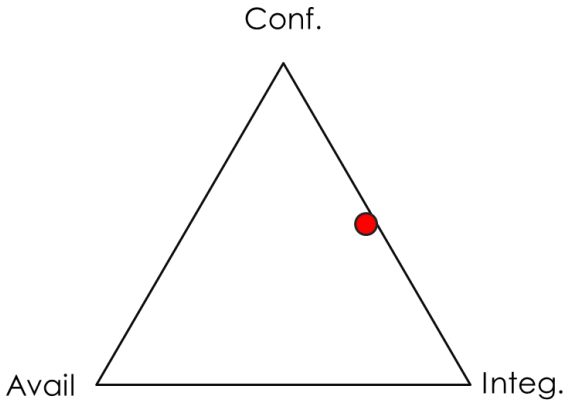
Podvrhnutie DNS odpovede (**DNS spoofing**) je typ útoku, ktorý zneužíva identifikáciu DNS otázok a odpovedí pomocou Identifikačného čísla (ID). Útočník potrebuje zistiť toto ID z otázky a následne môže poslať odpoveď s týmto ID a IP adresou odosielateľa zmenenou na IP adresu cieľového servera z otázky. Na odoslanie takejto odpovede sa útočník musí nachádzať na ceste medzi klientom a serverom. Ak sa útočník nenachádza na ceste medzi klientom a serverom a útočník „háda“ ID, nesmie sa vykonávať kontrola odchádzajúcich paketov na sieťových prepínačoch, cez ktoré sa pripája útočník. Ak by cieľom útoku nebol klient, ale DNS cache server, potom by po správnom vykonaní útoku bola cache pamäť servera „otrávená“ a cache server by následne odpovedal svojim klientom s podhodenými odpoveďami.

2.2.2 Otrávenie DNS cache

Cieľom **Otrávenia DNS cache** je prinútenie DNS cache servera k poskytovaniu nesprávnych odpovedí na otázky klientom, ktorí sa budú pripájať na tento server. Jednou z možností o ktorých vieme je aj podvrhnutie DNS odpovedí smerovaných na DNS server. Vo výsledku, DNS cache server bude mať po určitú dobu vo svojej cache zlé záznamy, ktoré bude ďalej šíriť. Teda dôjde k podobnému efektu ako pri podvrhnutí DNS odpovede.

2.2.3 Chyby v DNS implementácii

V tejto kapitole rozoberáme hrozby spočívajúce v zneužití chyby v implementácii DNS protokolu, ako je napríklad známe **pretečenie zásobníka**. Do tejto kategórie tiež zaradzujeme chyby v programovom vybavení, ktoré priamo využívajú protokol DNS. Príkladom je hrozba pod označením **CVE-2013-1923**. CIA model pre túto zraniteľnosť je



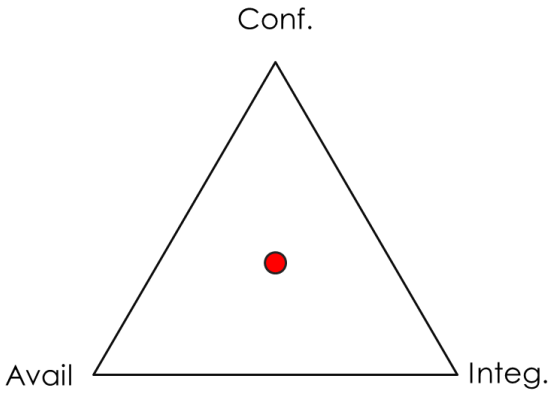
Obr. 3 CIA model pre CVE-2013-1923

zobrazený na obrázku č.3. V tejto hrozbe démon rpc-gssd (protokol na zabezpečenie vzdialených volaní) vo verzii balíka nfs-utils (sada nástrojov na prácu s sieťovým súborovým systémom) nižšej ako 1.2.8 vykonáva reverzný DNS preklad pre mená serverov počas GSSAPI (Generic Security Service Aplication Programming Interface) autentifikácie, čo môže dovoliť vzdialenému útočníkovi prečítať ináč nedostupné súbory pomocou podvrhnutia DNS odpovedí. Nakoľko nie je známa existencia kódu, ktorý by zneužíval túto zraniteľnosť (takýto kód nazývame exploit) a komplexnosť útoku je vysoká, hodnotenie podľa CVSSv2 je 3.2, čo predstavuje nízku mieru ohrozenia.

2.3 Hrozby a útoky voči dostupnosti DNS protokolu a jeho implementáciám

2.3.1 Chyby v DNS implementácii

V tejto kapitole popisujeme hrozby snažiace sa zneužiť chyby v implementácii DNS protokolu, nakoľko ich zneužitím je možné DNS server vyradiť z prevádzky alebo dokonca prinútiť vykonať útočníkov kód. Príkladom je hrozba pod označením **CVE-2015-6125.** CIA model pre túto zraniteľnosť je zobrazený na obrázku č.4.



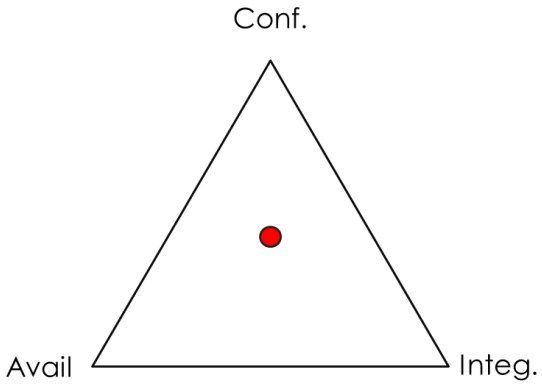
Obr. 4 CIA model pre CVE-2015-6125

V prípade tejto hrozby sa využíva chyba „Use after free“, teda využívanie časti pamäte aj po tom, ako bola uvoľnená. Hrozba sa nachádza v operačných systémoch Windows Server 2008 a Windows Server 2012 a dovoľuje vykonanie podvrhnutého kódu. Táto chyba sa dotýka všetkých kategórií CIA modelu, pretože spustenie útočníkovho kódu na serveri môže narušiť dôvernosť, integritu a aj dostupnosť. Podľa National Vulnerability Database (ďalej NVD) je ohodnotenie tejto zraniteľnosti pomocou CVSSv2 9.3, čo túto zraniteľnosť radí do kategórie s vysokou mierou ohrozenia.

2.3.2 Odoprenie služby

Hrozba odopretia služby napríklad pomocou DoS (denial of service, odopretie služby) útoku pomocou veľkého množstva prichádzajúcich spojení ktoré narušia prevádzku servera alebo server len dočasne preťažia. Príkladom je hrozba s označením **CVE-2015-7547**. CIA model pre túto zraniteľnosť je zobrazený na obrázku č.5. V rámci tejto hrozby dochádza k pretečeniu zásobníka vo funkciách send_dg a send_vc (funkcie na odoslanie UDP a TCP

paketu) v knižnici glibc (libc6). Táto hrozba umožňuje útok odopretia služby (DoS)



Obr. 5 CIA model pre CVE-2015-7547

pomocou špeciálne vytvorenej DNS odpovede na duálne A/AAAA požiadavky. Podobne ako v predchádzajúcom príklade zraniteľnosti, aj pri tomto je možné konštatovať, že sa táto hrozba dotýka každej z kategórií CIA modelu, avšak oproti vyššie spomenutej hrozbe je jej dopad oveľa rozsiahlejší. Dôvodom je to, že ide o pravdepodobne všeobecne používanú knižnicu, ktorá je používaná takmer v každej verzii operačného systému Linux. Pre túto zraniteľnosť NVD určuje skóre 8.1 pomocou schémy CVSSv3 a skóre 6.8 pomocou schémy CVSSv2.

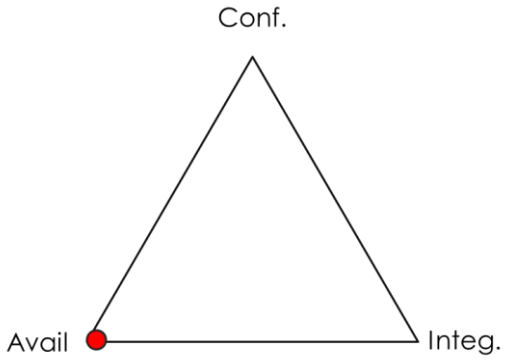
2.3.3 DNS zosilňujúci útok

Tento útok spočíva vo využití pomeru dĺžok otázky a odpovede v komunikácii s DNS serverom. Tento pomer je mnohokrát hodnota vyššia ako dva, čo pri jednoduchom zamenení zdrojovej adresy znamená zvýšenie sieťového toku. Správne prevedený tento útok je schopný dosiahnuť desať a viac násobne zvýšenie objemu prenášaných údajov približne.

Nebezpečnosť tohto typu útoku narastá s zavádzaním DNSSEC (DNS security extensions, rozšírenie protokolu DNS), ktoré v majoritnej väčšine prípadov vyžaduje rozšírenie EDNS0 (rozšírenie protokolu DNS), ktoré mení limit veľkosti odpovede z 512 bytov na zvyčajne 4096 bytov. Z tohto dôvodu efektívnosť takéhoto útoku narastá viacnásobne. Pri použití DNSSEC sa nám v testovacích podmienkach podarilo dosiahnuť zvýšenie objemu údajov smerujúcemu k obeti až 50-násobne.

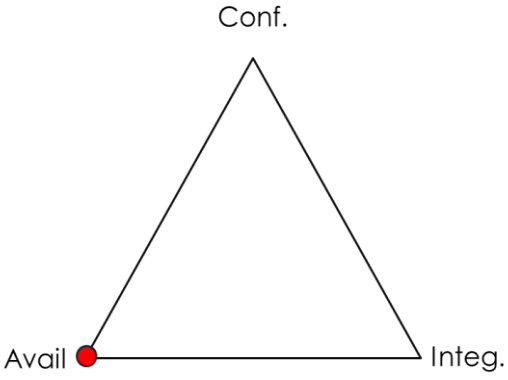
2.3.4 Hrozby typu „paket smrti“

Cieľom je čiastočné spomalenie alebo vyradenie z prevádzky cieľový DNS server. Útočník môže zneužiť chybu v implementácii DNS protokolu, ktorá môže pri nesprávnej konfigurácii viesť k prerušeniu prevádzky alebo ovládnutiu servera. Príkladom môže byť ľubovoľné spustenie kódu v prípade MaraDNS servera [15]. Podstatou tejto hrozby je opakovaná žiadať pre DNS server o vykonanie náročnej operácie. Tieto žiadosti vyťažia server natoľko, že nebude schopný odpovedať na regulárne požiadavky. Príkladmi tejto hrozby sú zraniteľnosti CVE-2013-2266 [16], CVE-2015-8719[17] a CVE-2016-2088[18].



Obr. 6 CIA model pre CVE-2015-8719

Zraniteľnosť s označením **CVE-2015-8719** sa prejavuje v chybe spracovania DNS paketov s EDNS0[19] Client Subnet s možnosťou spôsobujúcou pád aplikácie Wireshark – nástroja na analýzu sieťovej komunikácie. CIA model pre túto zraniteľnosť je zobrazený na obrázku č.6. U tejto chyby je viditeľné, že chyby týkajúce sa DNS sa netýkajú len servera a klienta, ale aj ďalších aplikácií, ktoré spracúvajú DNS komunikáciu. Miera zraniteľnosti má podľa NVD hodnotu 5.5, čo predstavuje strednú mieru kritickosti.



Obr. 7 CIA model pre CVE-2016-2088

Ďalším príkladom je zraniteľnosť pod označením **CVE-2016-2088**. CIA model pre túto zraniteľnosť je zobrazený na obrázku č.7. Táto zraniteľnosť spočíva v chybe DNS servera BIND vo verziách pred 9.10.3.P4 umožňujúca útok odopretia služby pomocou upraveného paketu s viac ako jednou cookie (údaje slúžiacena identifikáciu sedenia (session)). Poslanie špeciálne vytvoreného paketu umožnuje útočníkovi ukončiť vyššie uvedený DNS server z dôvodu chybného spracovania paketu na strane servera, čím dôjde k prerušeniu poskytovania služby. Táto chyba bola ohodnotená v schéme CVSSv2 hodnotou 4.3, nakoľko pri jej zneužití nedôjde k dopadu na dôvernosť alebo integritu.

3 Ochrana DNS protokolu a jeho implementácií

Tretia kapitola tejto záverečnej práce nadväzuje na predchádzajúcu kapitolu a zameriava sa na ochranu DNS protokolu a jeho implementácií voči bezpečnostným hrozbám uvedeným v predchádzajúcej kapitole. Podobne ako predchádzajúcu kapitolu aj túto rozdeľujeme na podkapitoly z hľadiska CIA modelu pre riadenie informačnej bezpečnosti v spoločnosti.

3.1 Ochrana dôvernosti DNS protokolu a jeho implementácií

DNS protokol v prvej verzii neobsahoval žiadne mechanizmy slúžiace na zabezpečenie dôvernosti a integrity prijatej odpovede a overenie totožnosti odosielateľa/prijímateľa správ [9].

Rozšírenia protokolu DNS, a to konkrétne **ENDS0** [19] a **DNSSEC** [21] poskytujú možnosť kryptograficky overiť dôvernosť odpovede a vďaka použitiu asymetrickej kryptografie s využitím verejného kľúča vieme určiť autora správy ako majiteľa privátneho kľúča, pomocou ktorého bola odpoveď podpísaná.

Vďaka existencii týchto mechanizmov zabezpečenia integrity a dôvernosti komunikácie medzi koncovými zariadeniami v protokole DNS predpokladáme, že hrozby proti dôvernosti záznamov poskytovaných pomocou DNS budú klesať úmerne s rastúcim nasadzovaním týchto mechanizmov.

3.1.1 Zabezpečenie zónových transferov

Implementácie DNS serverov (napríklad BIND servera) dovoľujú podľa [6] vo svojej konfigurácii definovať, z ktorých IP adries je možné vykonať zónový transfer. Konfigurácia sa vykonáva v konfiguračných súboroch servera a pre BIND je relatívne jednoduchá [22]. Najprv definujeme prístupovú sieť (ACL network), ktorá pomenováva a obsahuje zoznam IP adries alebo sietí. To môžeme vykonať napríklad takto:

acl zonetransfers { 192.168.1.1; };

Následne v definícii zóny pomocou parametru **allow-transfer** vyberieme pre ktoré siete je povolený zónový transfer:

zone example.com {type master; file example.zone; allow-transfer { zonetransfers }; };

Takáto konfigurácia je jednoduchá a vo väčšine prípadov aj účinná. Za túto konfiguráciu zodpovedá administrátor daného DNS servera.

3.1.2 Detekcia DNS Tunelovania

DNS tunelovanie je mechanizmus slúžiaci na obchádzanie filtrovania počítačovej siete, ktorý zneužíva skutočnosť, že vo väčšine sietí je povolená komunikácia pomocou protokolu DNS s vonkajšou sieťou [23]. To je vyriešené buď priamou cestou alebo cez DNS cache server poskytovateľa. V oboch prípadoch je možné preniesť údaje z lokálnej počítačovej siete do vonkajšej tak, že údaje sú zakódované do znakovej sady, ktorá je používaná pre doménové mená a následne sú generované požiadavky v tvare **<náhodne vyzerajúce doménové meno>.doména.tld**, ktoré sú odosielané buď priamo na server útočníka, alebo cez DNS cache server na doménu, ktorú má útočník registrovanú.

Následne sa na útočníkovom serveri dekódujú naspať do pôvodnej podoby. Existuje množstvo softwarových riešení, ktoré poskytujú túto funkcionalitu, ako je napríklad **dns2tcp** [24], **DNScapy** [25] a iné. Možnými ochranami voči takémuto zneužitiu DNS protokolu je napríklad:

- **monitorovanie počtu DNS požiadaviek na DNS server** - za predpokladu, že útočník má len jeden DNS server, na ktorý posiela komunikáciu, bude počet DNS paketov odchádzajúcich na tento server oveľa vyšší v porovnaní so zvyškom počítačovej siete.
- **štatistická analýza náhodnosti požiadavky** - nakoľko pravdepodobnosť toho, že údaje, ktoré budú zakódované do textovej podoby, budú zrozumiteľné a ich entropia bude nízka, je možné sledovať počet unikátnych požiadaviek odchádzajúcich od klienta a na základe zvýšenej náhodnosti požiadaviek predpokladať existenciu DNS tunela.
- **monitorovanie DNS paketov na známe príznaky v požiadavkách** - niektoré zo softvérových riešení na DNS tunelovanie majú známu formu požiadaviek, ktorú by malo byť možné detegovať napríklad pomocou systému pre detekciu narušení **Snort** [26].

3.2 Ochrana integrity DNS protokolu a jeho implementácií

3.2.1 Firewall

Pod pojmom **firewall** si môžeme predstaviť logickú bariéru zabraňujúcu neautorizovanej alebo nechcenej komunikácii medzi časťami počítačovej siete. Takáto bariéra môže byť umiestnená buď na koncových zariadeniach, alebo, častejšie na sieťových smerovačoch prepájajúcich lokálnu počítačovú sieť s Internetom. V našej práci sa budeme venovať prioritne práci s firewallom umiestneným na takomto smerovači. Teda predpokladáme, že zdrojom ani cieľom komunikácie, ktorú chceme filtrovať nie je zariadenie, na ktorom chceme filtrovanie vykonávať. Zdroj a cieľ komunikácie sa nachádzajú v rôznych logických častiach počítačovej siete, a teda komunikácia je len preposielaná (forwarding) [29].

Keďže spôsob komunikácie protokolu DNS využíva hlavne UDP spojenia a TCP spojenia sú využívané hlavne na zónové transfery. V niekoľkých prípadoch môže byť rovnaká požiadavka vykonávaná cez oba typy spojenia. Je tomu potrebné prispôsobiť aj pravidlá pri kontrole komunikácie vykonávanej firewallom.

Zatiaľ čo pri UDP spojeniach stačí kontrolovať údaje pre jediný paket. Naopak, pri TCP je potrebné najprv nadviazať spojenie a až následne kontrolovať obsah údajov, ktoré sa nachádzajú vo viacerých paketoch a na cieľovú stanicu môžu doraziť v rôznom poradí. To znamená, že firewall by musel znovu zostaviť časti komunikácie medzi klientom a serverom, ktoré môžu byť rozdelené vo viacerých TCP paketoch, teda vykonávať vytvorenie TCP paketu (TCP reassembly) [30], ktoré je väčšinou vykonávané sieťovou vrstvou operačného systému na strane prijímateľa. Nakoľko existujú rôzne algoritmy pre takéto skladanie TCP paketov a vo všeobecnosti nevieme, aký algoritmus použije koncová stanica pri zasielaní prekrývajúcich sa častí správy pochádzajúcich od útočníka, nevieme určiť obsah správy. Ide o známu techniku **vyhýbania sa detekcii**.

3.2.2 Prístupové listy

Prístupové listy (access lists) predstavujú bezpečnostnú funkcionalitu dostupnú najmä na sieťových prepínačoch. Ide o typ bezpečnostného mechanizmu slúžiaceho na filtrovanie komunikácie medzi počítačovými sieťami. Zvyčajne poskytuje možnosť definovať, aké typy paketov je povolené preposlať do inej časti siete [23]. Pre naše potreby zabezpečenia DNS

servera by bolo možné použiť napríklad nasledujúce pravidlo pre prístupový zoznam (access list) pre sieťové prepínače a sieťové smerovače CISCO:

access-list 110 permit udp any eq domain host 192.168.1.1 gt 53

Vyššie uvedené pravidlo povoľuje komunikáciu pre UDP pakety smerujúce na stroj s IP adresou 192.168.1.1 a portom 53. Táto funkcionalita je užitočná pre základné zabezpečenie a pre filtrovanie komunikácie podľa IP adries ešte pred tým, než sa dostanú na firewall, čím by sa odľahčila časť počítačovej siete. V súčasnej dobe nemáme vedomosť o existencii prístupového zoznamu, ktorý by dovoľoval vykonávať hĺbkovú analýzu paketov. Na základe tohto usudzujeme, že ochranný mechanizmus prístupových zoznamov nie je vhodný ako platforma na zabezpečenie komunikácie s DNS serverom podľa nami zvolených požiadaviek.

3.3 Ochrana dostupnosti DNS protokolu a jeho implementácií

3.3.1 Útoky preťažujúce server

Proti tejto kategórii hrozieb je možné na strane DNS servera sa brániť detekciou zvýšeného počtu požiadaviek prichádzajúcich od útočníka (samozrejme, ak ide o jedného útočníka). To je možné buď zahadzovaním nadmernej komunikácie na strane DNS servera, alebo požiadaním o blokovanie prichádzajúcej komunikácie už skôr, ako sa dostane na cieľový server, čím môžeme predísť zahlteniu dostupného pripojenia.

Keďže primárnym protokolom pre DNS je protokol UDP, u ktorého je veľmi ľahko možné pozmeniť zdrojovú IP adresu, takýmto zahadzovaním alebo blokovaním by sme mohli odoprieť služby regulárnemu užívateľovi. Danú situáciu je možné riešiť pomocou odoslania **odpovede „TCP only“** za určitý časový interval a povolenia TCP spojenia na DNS (v kombinácii s TCP cookies).

Na strane obete môžu byť hrozby proti dostupnosti smerované na server, s ktorým sa obeť snaží komunikovať, použité ako prostriedok na zvýšenie účinnosti iného typu útoku.

.

3.3.2 DNS zosilňujúci útok

Odporúčaným opatrením pre tento typ útokov je tzv. **RRL** (**Response Rate Limiting –** limitovanie počtu odpovedí za určitú dobu) mechanizmus, ktorý obmedzuje počet odpovedí odoslaných na zdrojovú adresu, pričom odporúčaný limit je 5 požiadaviek z jednej IP adresy za sekundu [22].

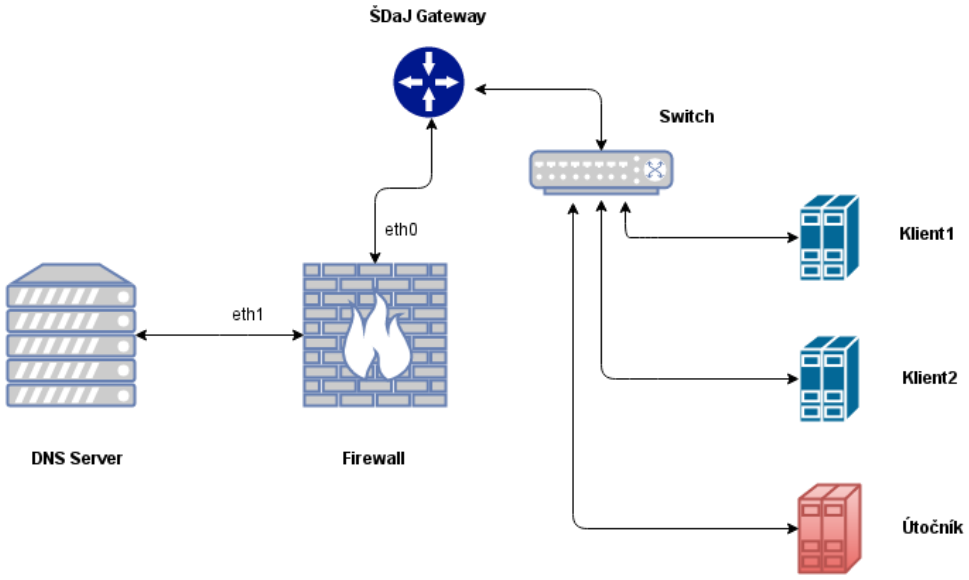
Tento spôsob ochrany je podľa nášho názoru nedostatočný pre typy útokov, ktoré využívajú veľký počet DNS serverov súčasne. Počet paketov smerujúci na jeden DNS server, zúčastnený v útoku je len 1/n, z množstva n paketov, ktoré by prijal server pri útoku s použitím jedného DNS servera. Útočníkovi teda stačí na odoslanie útoku pozostávajúceho z m paketov vybrať m/5 serverov, ktoré sa zúčastnia a na každý z nich odoslať 5 paketov za sekundu. Takto na jednotlivých serveroch nebude zachytený útok, pretože neprekročí limit počtu paketov za sekundu. Navyše, pre obeť bude oveľa ťažšie brániť sa náporu prichádzajúcich údajov, keďže tie budú prichádzať z minimálne n rôznych IP adries.

4 Aplikačný firewall pre DNS server

Záverečná kapitola tejto práce sa venuje návrhu a implementácii ochrany pre DNS server. Na účely implementácie sme si zvolili implementáciu DNS servera - **BIND 9.9** [6], ktorá ako sme už vyššie spomenuli, patrí k najrozšírenejším verziám DNS servera.

4.1 Testovacie a vývojové prostredie

Základným predpokladom pre úspešnú implementáciu navrhovaného bezpečnostného riešenia je vhodné testovacie prostredie. Priestory na vytvorenie uvedenej počítačovej siete nám poskytli Študentské domovy a jedálne UPJŠ v Košiciach. Architektúra testovacej a vývojovej počítačovej siete je znázornená na obrázku č. 8.



Obr. 8 Architektúra testovacej a vývojovej počítačovej siete

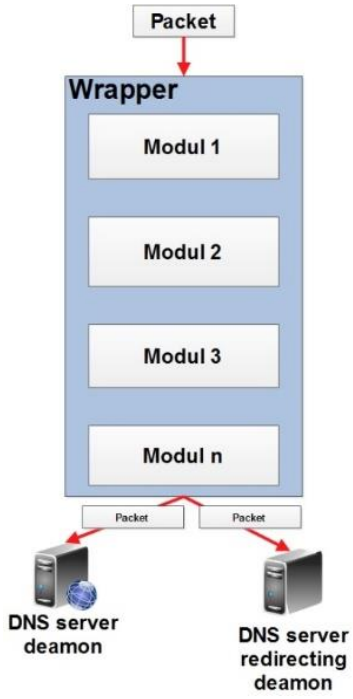
Ako prostredie pre implementáciu firewallu sme si zvolili operačný systém Linux vo verzii jadra 3.x s programovacím jazykom Python 2.7 [31], ktorý v sebe ako modul obsahuje veľmi známy firewall netfilter. Pre prácu s týmto modulom využívame **knižnicu nfqueue**, ktorá dovoľuje spracovanie paketov pomocou procesov v používateľskom móde systému. Táto knižnica je implementovaná á v jazyku C, avšak pre potreby zjednodušenia implementácie využívame **knižnicu NetfilterQueue** [32], ktorá zaobaľuje volania knižnice nfqueue v jazyku Python. Pri spracovaní paketov pracujeme s **frameworkom Scapy** [33], ktorý je určený na vysokoúrovňovú prácu s paketmi v objektovo orientovanom prostredí

jazyka Python. Výhodou frameworku Scapy je jeho rozsiahlosť, nakoľko nám dovoľuje analyzovať jednotlivé časti paketov. Zabudovaná podpora DNS protokolu je veľkým prínosom, nakoľko nie je potrebné opätovne písať vlastnú implementáciu spracovania DNS paketov. Dôležitým faktorom je jednoduchosť implementácie logiky firewallu s použitím knižnice NetfilterQueue a frameworku Scapy, čo nám to umožňuje udržať prehľadnosť a modularitu kódu.

4.2 Návrh systému

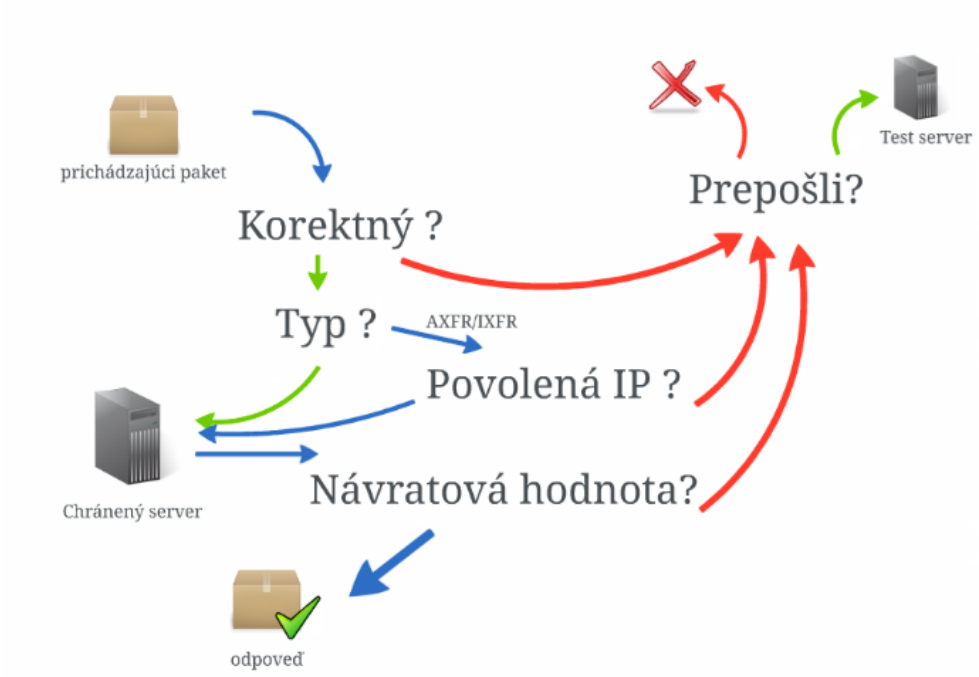
Podstatou našej implementácie bude vytvorenie systémového démona, ktorý bude spustený na pozadí aktívneho sieťového prvku s operačným systémom Linux. Tento prvok plní úlohu firewallu pomocou modulu **Netfilter** [34]. Nami vytvorený démon bude rozširovať existujúci firewall o možnosti filtrovania prichádzajúcej a odchádzajúcej komunikácie DNS servera. Takéto implementačné riešenie dovoľuje nasadiť démon buď na dedikovanom stroji, ktorý filtruje komunikáciu pre viacero počítačov, alebo na samotnom počítači s DNS serverom,. Navrhované riešenie bude spĺňať funkcionalitu DNS firewallu, ktorý bude chrániť DNS server pred 3 bezpečnostnými hrozbami:

- zónové transfery z nesprávnych IP adries,
- invalidné DNS pakety a
- skenovanie DNS (footprinting).



Obr. 9 Návrh systému - moduly

Každý modul predstavuje implementáciu ochrany voči konkrétnej hrozbe. Na obrázku č. 9 je znázornený všeobecný návrh systému. Podstatou systému je to, že každý paket prichádzajúci cez firewall je postupne testovaný filtrami v rámci jednotlivých modulov. Ak paket splní test filtra v rámci modulu, je posunutý na nasledujúci modul. V opačnom prípade dochádza k jeho zahodeniu, resp. presmerovaniu. Paket je vo všeobecnosti v rámci systému presmerovaný vtedy, ak sa má zahodiť, ale bolo nastavené presmerovanie na testovací, resp. logovací server. Schéma sekvenčného spracovania paketu je zobrazená na obrázku č. 10.



Obr. 10 Schéma sekvenčného spracovania paketu

4.2.1 Ochrana pred invalidnými DNS pakety

Ako sme už uviedli v predchádzajúcej časti práce, DNS protokol je binárnym protokolom s presne danou štruktúrou. Na základe toho vieme definovať, ako vyzerá regulárny DNS paket. Podstatou tohto modulu je **zostavenie regulárneho výrazu**, ktorý aplikujeme na každý prechádzajúci paket. Ak paket vyhovie podmienky regulárneho výrazu, pustí sa ďalej. Podobne ako v predchádzajúcom prípade sa paket zahodí, alebo presmeruje. Paket sa zahodí v prípade, ak štruktúra paketu nekorešponduje s vopred definovaným regulárnym výrazom. Najčastejšie pôjde o nasledujúce prípady:

- paket neobsahuje DNS informácie podľa špecifikácie protokolu,
- paket bol pozmenený pri prenose, alebo
- ide o pokus o útok na DNS server.

Regulárny výraz v rámci tohto modulu sa dá meniť podľa požiadaviek administrátora. V rámci toho je možné aj zakázať jednotlivé typy požiadaviek na server.

4.2.2 Ochrana pred zónovými transfermi z nesprávnych IP adries

Podstatou tohto modulu je filtrovanie DNS paketov podľa ich typu. Pri zónových transferoch (typ AXFR/IXFR) ide o prenos, ktorý slúži na synchronizáciu master a slave serverov. V tomto prípade sa prenáša obsah celej zóny a navrhovaný modul filtruje DNS pakety podľa IP adresy. V prípade, že IP adresa bude v **zozname povolených IP adries** (tzv. **whitelist-e)**, paket sa pustí. V opačnom prípade sa paket zahodí, alebo presmeruje. Paket sa zahodí v prípade, ak je paket zónovým transferom a pochádza z IP adresy, ktorá nie je na zozname povolených IP adries. Bezpečnosť v rámci tohto modulu je založená na rozhodnutí administrátora ktoré IP adresy zaradí na zoznam povolených IP adries. Napriek tomu, že v konfiguráciách DNS serverov je možné konfigurovať pre ktoré servery sú povolené transfery zón, môžeme tento modul považovať za ďalšiu vrstvu ochrany.

4.2.3 Ochrana pred DNS footprintingom

Pri **DNS footprintingu** sa venujeme sledovaniu odpovedí od DNS servera. Ak za sledované obdobie dostaneme veľa odpovedí rovnakého typu (napríklad informácie o neexistujúcom zázname) pre rovnakú IP adresu, začneme blokovať komunikáciu s danou IP adresou pre UDP pakety. V tomto prípade predpokladáme, že ide o DNS footprinting nášho DNS servera. Ak za sledované obdobie dostaneme veľa odpovedí aj pre TCP pakety, blokujeme následne aj tie.

Pri tomto module je dôležitá správna údajová štruktúra. Vstupnou požiadavkou pri návrhu tejto štruktúry bolo, aby bola vopred definovateľná veľkosť dátovej štruktúry. Možným riešením reprezentácie dátovej štruktúry by mohol byť binárny strom. Avšak, veľká časť takéhoto stromu by bola nevyužitá, nakoľko štandardne server nekomunikuje s každou IP adresou, ktorá existuje. Výhodou takéhoto riešenia by bola možnosť rýchleho výpočtu počtu požiadaviek z časti počítačovej siete. Napokon sme sa nechali inšpirovať

implementáciami bezpečnostného mechanizmu RRL a rozhodli sme sa použiť ako údajovú štruktúru hašovaciu tabuľku (hash-table**)** s obmedzenou konfigurovateľnou veľkosťou.

4.3 Implementácia riešenia

V nasledujúcej časti sa budeme venovať implementačným detailom firewallu a jednotlivých modulov.

4.3.1 Implementácia ochrany proti anomáliám v DNS požiadavkách

Pri implementácii ochrany proti anomáliám v DNS požiadavkách sme sa sústredili na špecifikovaniu povolených typov požiadaviek. Zamerali sme sa na obmedzenie DNS paketu na pakety, ktoré sú požiadavkami. To znamená, že majú nastavený QR bit na hodnotu 0. V tomto prípade ich typ je 0, teda "standard query“ opcode=0 a majú nastavené bity AA, TC, RD, RA a Z na hodnoty 0. Takisto filtrujeme požiadavky, ktoré majú nastavený RCODE na hodnotu rôznu od 0. Bližšie informácie k jednotlivým položkám DNS paketu sme uviedli v úvodnej kapitole. Nižšie uvádzame ukážku zdrojového kódu, ktorá implementuje ochranu proti anomáliám v DNS požiadavkách.

```
def handle_dns_udp_packet(packet)
        p=packet
        if "UDP" in p and "DNS" in p:
                dns_packet=p[IP]
               if p[DNS].opcode == 0 and \
               p[DNS].qdcount == 1 and \
                (dns_packet.aa,
                dns_packet.tc,
                dns_packet.rd,
                dns_packet.ra,
                dns_packet.rz) == (0,0,0,0,0) :
                        pkt.accept()
                        return true
       else:
               pkt.drop()
                return false
```

Spracovanie DNS UDP paketu je relatívne jednoduché, a keďže sa predpokladá, že táto funkcia bude volaná veľmi často, oddelili sme ju od metódy, ktorá spracováva TCP DNS pakety. Dôvodom bola skutočnosť, že pri TCP DNS paketoch nie je potrebné vykonávať sledovanie spojenia.

Pri použití TCP DNS prenosu filtrujeme súčasne požiadavky, ktoré majú nastavenú hodnotu **QDCOUNT** (počet požiadaviek) na vyššiu ako 1 a sú rozdelené do viac ako 2 paketov. Túto činnosť si vyžaduje nutnosť rekonštrukcie TCP paketov. Ak by sme sa pokúsili implementovať rekonštrukciu TCP paketov, útočník by mohol použiť jednu z techník vyhnutia sa detekcie využívajúcu prekrývajúce sa segmenty zasielané v rôznom poradí.

4.3.2 Implementácia ochrany proti neautorizovaným zónovým transferom

Autorizáciu zónových transferov vykonávame inšpekciou TCP spojenia obsahujúceho DNS rámec s požiadavkou na zónový transfer. Následne je overené, či sa IP adresa odosielateľa nachádza v zozname povolených adries. V rámci implementácie ochrany kontrolujeme každý TCP paket obsahujúci určitý obsah. Pakety nesplňujúce požiadavky pre filter následne zahadzujeme s tým, že otvorené spojenie zatvoríme zaslaním TCP paketu s príznakom RST.

Komunikácia cez TCP je na rozdiel od UDP rozdelená do viacerých paketov, ktoré pre potreby kontroly musíme zaznamenať a následne vyskladať. To znamená nutnosť implementácie pamäte slúžiacej na zapamätanie prijatých údajov a taktiež aj metód správy tejto pamäte (vypršanie času, preplnenie, atď.).

Zdrojový kód spracúvajúci ochranu proti neautorizovaným zónovým transferom je rozsiahlejší. Nižšie uvádzame len ukážku zdrojového kódu, do ktorého sme časti kontroly ochrany proti neautorizovaným zónovým transferom presunuli do samostatných metód. Celý zdrojový kód sa nachádza v prílohe B.

```
tcps=[]

def handle_zone_transfer(packet):
 p=packet
 if "TCP" in p:
                if "Raw" in p:
                        id=str(p[IP].src)+str(p[TCP].sport)+str(p[IP].dst)+str(
 p[TCP].dport)
                        if id in tcps:
                                #try to build DNS packet from data

                         dns,rebuild_complete=rebuild_packet(tcps,p)
                         if rebuild_complete and is_ip_allowed(dns):
                                 pkt.accept()
                                 return true
                         else:
                                 return false
```

```
                        return true

                    else:
                            t=TCPRecord()
                            t.timestamp=int(time.time())
                            t.data=p[Raw].load
                            tcps[id]=t
                            pkt.accept()
                            return true

       elif type(p.payload.payload) is scapy.packet.NoPayload and p.closing_co
nnection():
        pkt.accept()
               return true

return false
```

Metóda, ktorej vstupom je IP paket (musíme pracovať s IP paketmi, nakoľko potrebujeme riešiť problém uzatvárania pripojenia) najprv overí, či ide o TCP Raw paket. Následne, ak nejde o prvý paket v spojení, ktoré je určené zdrojovou a cieľovou IP adresou a portami (pre korektnosť uvádzame aj cieľový port, hoci by mal byť vždy rovnaký), modul firewallu sa pokúsi v pomocnej metóde zostaviť DNS paket. Metóda pre vytvorenie paketu je relatívne striktná. Neakceptuje prekrývajúce sa TCP pakety. Pakety zostavuje rovnakým spôsobom ako v jadre Linuxu. Toto obmedzenie je z dôvodu už vyššie spomenutého problému s TCP reassembly útokmi. Pri tomto type prenosov sa nepredpokladá stav, kedy by bolo potrebné riešiť problémy sieťovej infraštruktúry, ktoré by mohli meniť veľkosť TCP paketov.

Ak je vytvorenie kompletné a súčasne ide o korektnú DNS požiadavku na zónový transfer, modul v pomocnej metóde overí, či je IP adresa medzi povolenými adresami. Povolené IP adresy sú tie, ktoré boli načítané pri spustení firewallu z konfiguračného súboru.

Ak ide o prvý paket v spojení, modul inicializuje pomocné pole na dočasné uloženie TCP paketov, ktoré sa bude pokúšať pri každom ďalšom pokuse vytvoriť. Následne takýto paket povolí. To dovolí nadviazať TCP spojenie pomocou (TCP handshake) medzi chráneným master serverom a slave serverom, ktorý sa pokúša o zónový transfer.

4.3.3 Implementácia ochrany proti footprintingu

Modul pre **ochranu proti footprintingu** spočíva v detekcii počtu odpovedí od DNS servera, ktorých kód odpovede je nenulový. Teda modul indikuje chybu na strane DNS

servera pri spracovaní požiadavky. Zachytávanie komunikácie medzi DNS serverom a klientom je vykonávané rovnakým spôsobom ako pri iných moduloch firewallu, keďže odpovede prechádzajú cez firewall.

Ak detegujeme zvýšený počet takýchto odpovedí za určitý časový interval, ide s vysokou pravdepodobnosťou o footprinting. Následne môžeme uložiť do záznamu (logu) informáciu o možnom footprinting útoku a znížiť, prípadne kompletne zastaviť komunikáciu s klientom. V implementácii tohto modulu si ukladáme počet prijatých požiadaviek a počet negatívnych odpovedí pre každú klientsku IP adresu rozdelených podľa typu za určité časové obdobie. Nižšie uvádzame ukážku kódu (v referenčnej implementácii sme použili databázu).

```
def handle_footprinting(dns_packet):
        c.execute("select count(*) from connections where IP=?",(dns_packet[IP]
.src,))
        resp=int(c.fetchone()[0])
        limtime=int(currtime-CONN_THRES)
        conn.commit()
        c.execute("delete from connections where time <= ?",(limtime,))
        c.execute("insert into connections values(?,?)",(dns_packet[IP].src,cur
rtime))
        conn.commit()

        # handle response

        if type(resp)==type(0) and resp>CONN_LIMIT:
                log_msg("dropping resp>CONN_LIMIT")
                pkt.drop()
                return false

        pkt.accept()
        return true
```

Metóda, ktorej vstupom je paket s invalidnou odpoveďou (validita odpovede sa zisťuje v inej časti kódu), sa skladá z dvoch blokov. Prvý blok používa databázu na zistenie a aktualizovanie počtu spojení. Naopak druhý blok zisťuje, či bol prekročený počet odpovedí.

Záver

V rámci tejto záverečnej práce sme sa venovali problematike bezpečnosti DNS protokolu a jeho implementácií. Ako sme ukázali v prechádzajúcich kapitolách, daná problematika je neustále aktuálna, čoho dôkazom je aj pomerne vysoké množstvo zraniteľností objavujúcich sa na DNS serveroch, ktoré zabezpečujú praktickú implementáciu DNS protokolu.

Úvodná kapitola tejto práce sa venovala správe doménových mien, najmä popisu DNS protokolu, jeho štruktúre, záznamom a komunikácii. Keďže v práci sme navrhovali ochranu pre zónové transfery, súčasťou kapitoly je aj popis týchto transferov. V rámci úvodnej časti práce sme zamerali aj na súčasne implementácie DNS protokolu.

V práci sme tiež analyzovali bezpečnostné hrozby DNS protokolu a jeho implementácií a súčasne opatrenia voči týmto hrozbám. Vychádzali sme z CIA modelu informačnej bezpečnosti a na základe tohto modelu sme rozdelili jednotlivé hrozby a opatrenia. V druhej kapitole sme uviedli niekoľko hrozieb a útokov, ktoré priamo smerujú voči DNS protokolu, ale najmä voči jeho implementáciám. Najväčšie množstvo hrozieb je namierených voči integrite a dostupnosti DNS protokolu a jeho implementácií (ak neberieme v úvahu DNSSEC). V tretej kapitole sme sa následne zamerali na ochranu voči hrozbám uvedeným v druhej kapitole tejto práce. Jedným z častých problémov všetkých opatrení sú postranné útoky.

Navrhovaný firewall je určený pre nasadenie na strane poskytovateľa internetového pripojenia pre DNS servery. Súčasťou návrhu firewallu je aj analýza úspešnej ochrany voči zvoleným útokov v závislosti od intenzity a miery distribuovateľnosti jednotlivých útokov. Pre účely implementácie sme si zvolili DNS server BIND 9. Navrhovaný firewall rieši problém filtrácie typov DNS komunikácie na strane poskytovateľa internetového pripojenia. Tiež sa zameriava na ochranu DNS servera proti neautorizovaným zónovým transferom pomocou filtrovania jednotlivých druhov DNS požiadaviek podľa zdrojovej IP adresy. Napokon naše riešenie umožňuje vytvárať štatistiku DNS požiadaviek a ich odpovedí a na základe tejto štatistiky upozorňovať zodpovednú osobu na možný fingerprinting útok. Nami vytvorené riešenie bolo otestované v rámci testovacej sieti ŠDaJ UPJŠ v Košiciach.

Zaujímavým problémom, s ktorými sme sa v práci stretli, je problematický mechanizmus vytvorenia TCP z pohľadu kontroly obsahu paketov v firewalle vďaka postranným útokom a možnosti vyhnutia sa kontroly týchto paketov.

Zoznam použitej literatúry

[1] RFC 799 - Internet Name Domains. In: IETF [online]. Dostupné z https://www.ietf.org/rfc/rfc799, [Naposledy navštívené: 17.4.2016]

[2] Projekt P2P-DNS [online]. Dostupné z https://github.com/Mononofu/P2P-DNS, [Naposledy navštívené: 17.4.2016]

[3] MOCKAPETRIS, P. V. DNS encoding of network names and other types. 1989.

[4] RFC 1035 - Domain names - concepts and facilities. In: IETF [online]. Dostupné z https://www.ietf.org/rfc/rfc1035, [Naposledy navštívené: 17.4.2016]

[5] DNS Amplification Attack, nirflog.com [online]. Dostupné z http://nirlog.com/2006/03/28/dns-amplification-attack/, [Naposledy navštívené: 17.4.2016]

[6] BIND server [online]. Dostupné z http://www.isc.org/downloads/BIND/, [Naposledy navštívené: 17.4.2016]

[7] Knot DNS server [online]. Dostupné z https://www.knot-dns.cz/ [Naposledy navštívené: 17.4.2016]

[8] CZ.NIC [online]. Dostupné z https://www.nic.cz/ [Naposledy navštívené: 17.4.2016]

[9] Dnsmasq server [online]. Dostupné z http://www.thekelleys.org.uk/dnsmasq/doc.html [Naposledy navštívené: 17.4.2016]

[10] Microsoft DNS [online]. Dostupné z https://technet.microsoft.com/sk-sk/network/bb629410.aspx [Naposledy navštívené: 17.4.2016]

[11] PERRIN, Chad. The CIA triad. [online]. Dostupné z http://www. techrepublic. com/blog/security/the-cia-triad/488, 2008, [Naposledy navštívené: 17.4.2016]

[12] MCCLURE, Stuart, et al. Hacking exposed: network security secrets and solutions. New York: McGraw-Hill/Osborne, 2005.

[13] BLYTH, Andrew. Footprinting for intrusion detection and threat assessment. Information Security Technical Report, 1999, 4.3: 43-53.

[14] RFC 3833 - Threat Analysis of the Domain Name System (DNS). In: IETF [online]. Dostupné z https://www.ietf.org/rfc/rfc3833, [Naposledy navštívené: 17.4.2016]

[15] MaraDNS: Arbitrary code execution [online]. Dostupné z http://seclists.org/fulldisclosure/2011/Nov/331 [Naposledy navštívené: 17.4.2016]

[16] Zraniteľnosť CVE-2013-2266 [online]. Dostupné z https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2013-2266 [Naposledy navštívené: 17.4.2016]

[17] Zraniteľnosť CVE-2015-8719 [online]. Dostupné z https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2015-8719 [Naposledy navštívené: 17.4.2016]

[18] Zraniteľnosť CVE-2016-2088 [online]. Dostupné z https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-2088 [Naposledy navštívené: 17.4.2016]

[19] VIXIE, Paul. Extension mechanisms for DNS (EDNS0). 1999.

[20] RFC 1034 - Domain names - concepts and facilities. In: IETF [online]. Dostupné z https://www.ietf.org/rfc/rfc1034, [Naposledy navštívené: 17.4.2016]

[21] ATENIESE, Giuseppe; MANGARD, Stefan. A new approach to DNS security (DNSSEC). In: Proceedings of the 8th ACM conference on Computer and Communications Security. ACM, 2001. p. 86-95.

[22] Transit Access Control Lists: Filtering at Your Edge [online]. Dostupné z http://www.cisco.com/c/en/us/support/docs/ip/access-lists/44541-tacl.html [Naposledy navštívené: 17.4.2016]

[23] BORN, Kenton; GUSTAFSON, David. Detecting dns tunnels using character frequency analysis. arXiv preprint arXiv:1004.4358, 2010.

[24] Projekt dns2tcp [online]. Dostupné z http://www.hsc.fr/ressources/outils/dns2tcp/ [Naposledy navštívené: 17.4.2016]

[25] Projekt DNScapy [online]. Dostupné z https://github.com/FedericoCeratto/dnscapy/commits/ master [Naposledy navštívené: 17.4.2016]

[26] Snort [online]. Dostupné z https://www.snort.org/ [Naposledy navštívené: 17.4.2016]

[27] Response Rate Limiting in the Domain Name System. [online]. Dostupné z http://www.redbarn.org/dns/ratelimits [Naposledy navštívené: 17.4.2016]

[28] SANS Institute: Why is securing DNS zone transfer necessary [online]. Dostupné z http://www.sans.org/reading-room/whitepapers/dns/securing-dns-zone-transfer-868, [Naposledy navštívené: 17.4.2016]

[29] WHITE, Lee J.; LEUNG, Hareton KN. A firewall concept for both control-flow and data-flow in regression integration testing. In: Software Maintenance, 1992. Proceerdings., Conference on. IEEE, 1992. p. 262-271.

[30] WAGENER, Gérard; DULAUNOY, Alexandre; ENGEL, Thomas. Towards an estimation of the accuracy of TCP reassembly in network forensics. In: Future Generation Communication and Networking, 2008. FGCN'08. Second International Conference on. IEEE, 2008. p. 273-278.

[31] Python 2.7 [online]. Dostupné z https://www.python.org/download/releases/2.7/ [Naposledy navštívené: 17.4.2016]

[32] Knižnica NetfilterQueue [online]. Dostupné z https://pypi.python.org/pypi/NetfilterQueue/0.3 [Naposledy navštívené: 17.4.2016]

[33] Scapy [online]. Dostupné z http://www.secdev.org/projects/scapy/ [Naposledy navštívené: 17.4.2016]

[34] Netfilter [online]. Dostupné z http://www.netfilter.org/ [Naposledy navštívené: 17.4.2016

Prílohy

Príloha A: Skript load_config.py a dns_config.cfg

Príloha B: Skript handle_zone_transfer.py

Príloha C: Skript handle_footprinting.py

Príloha A: Skript load_config.py a dns_config.cfg

Zdrojový kód na načítanie konfiguračného súboru.

```
def load_config():
        config=SafeConfigParser()
        #nacitanie konfiguracneho suboru
        log_msg("Nacitane konfiguracne subory:")
        log_msg(config.read(config_candidates))
        allowed_ip=config.get('Allowed','transfer').split()
        verbose=config.getboolean('Global','verbose')
        global QUEUE_NUM
        QUEUE_NUM=int(config.getint('Global','queuenum'))
        global RESP_LIMIT
        RESP_LIMIT=int(config.getint('Global','resp_limit'))
        global CONN_LIMIT
        CONN_LIMIT=int(config.getint('Global','conn_limit'))
        global RESP_THRES
        RESP_THRES=int(config.getint('Global','resp_thres'))
        global CONN_THRES
        CONN_THRES=int(config.getint('Global','conn_thres'))
        log_msg("queuenum",int(config.getint('Global','queuenum')))
        log_msg(QUEUE_NUM)
        log_msg("Povolene IP adresy:")
        log_msg(allowed_ip)
```

Konfiguračný súbor.

```
[Allowed]
transfer =
        192.168.1.1
         192.168.1.2

[Global]
verbose = 1
queuenum = 1
resp_limit = 5
conn_limit = 10
resp_thres = 60
conn_thres = 60
```

Príloha B: Skript handle_zone_transfer.py

Zdrojový kód na ochranu proti neautorizovaným zónovým transferom.

```
allowed_ip=[]
tcps=[]

# load config je vykonany pri starte len raz
load_config()

def handle_zone_transfer(packet):
        p=packet
        if "TCP" in p:
                if "Raw" in p:
                        id = packet_id(p)
                        if id in tcps:
                                #try to build DNS packet from data
                                 dns,rebuild_complete=rebuild_packet(tcps,p)
                                 if rebuild_complete and is_ip_allowed(dns):
                                         pkt.accept()
                                         return true
                                 else:
                                         return false
                                 return true

                        else:
                                t = TCPRecord()
                                t.timestamp = int(time.time())
                                t.data = p[Raw].load
                                tcps[id] = t
                                pkt.accept()
                                return true

        elif type(p.payload.payload) is scapy.packet.NoPayload and p.closing_co
nnection():
                 pkt.accept()
                return true

        return false

def packet_id(p):
        id=str(p[IP].src)+str(p[TCP].sport)+str(p[IP].dst)+str(p[TCP].dport)
```

```
        return id

def rebuild_packet(tcps,p):
        id = packet_id(p)
        if id in tcps:
                # just simple attach to end of packet
                payload = tcps[id].load + p[Raw].load
                if DNS(payload):
                        return payload,true
        return none,false

def is_ip_allowed(dns):
        src_addr = IP(dns).src
        if src_addr in allowed_ip:
                return true
        else:
                log_msg("LOG: src_addr: {} not allowed for zone transfer".forma
t(src_addr))
```

Príloha C: Skript handle_footprinting.py

Ukážka kódu na ochranu proti footprintingu.

```
def handle_footprinting(dns_packet):
        c.execute("select count(*) from connections where IP=?",(dns_packet[IP]
.src,))
        resp=int(c.fetchone()[0])
        limtime=int(currtime-CONN_THRES)
        conn.commit()
        c.execute("delete from connections where time <= ?",(limtime,))
        c.execute("insert into connections values(?,?)",(dns_packet[IP].src,cur
rtime))
        conn.commit()

        # handle response

        if type(resp)==type(0) and resp>CONN_LIMIT:
                log_msg("dropping resp>CONN_LIMIT")
                pkt.drop()
                return false

        pkt.accept()
        return true
```