FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

Zadání bakalářské práce

Název:	Analýza bezpečnosti DNS serverů
Student:	Michal Kovačič
Vedoucí:	Ing. Viktor Černý
Studijní program:	Informatika
Obor / specializace:	Bezpečnost a informační technologie
Katedra:	Katedra počítačových systémů
Platnost zadání:	do konce letního semestru 2021/2022

Pokyny pro vypracování

Kategorizujte známé typy útoků na servery DNS a popište jejich model hrozeb.
Sestavte list aktuálně dostupných DNS serverů a jejich bezpečnostních vlastností.
Spolu s vedoucím práce vyberte typy útoků, které lze replikovat v emulovaném prostředí.
Vybrané útoky otestujte v emulátoru na DNS serverech, které vzešly z analýzy.
Výsledkem práce bude srovnání DNS serverů podle jejich zranitelnosti na vybrané útoky.

Elektronicky schválil/a prof. Ing. Pavel Tvrdík, CSc. dne 6. února 2021 v Praze.

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Bakalářská práce

Analýza bezpečnosti DNS serverů

Michal Kovačič

Katedra Počítačových systémů
Vedoucí práce: Ing. Viktor Černý

27. června 2021

Poděkování

Největší poděkování patří vedoucímu práce Ing. Viktorovi Černému, za jeho vzorové vedení práce, nekonečnou trpělivost a cenné rady.
Dále bych rád poděkoval svým blízkým a své rodině za mentální podporu a za jejich projevenou důvěru.
Nakonec bych rád poděkoval všem přátelům, spolužákům, učitelům a trenérům, na které jsem se mohl v případě nouze obrátit a díky kterým jsem byl schopen dokončit studium i přes všechny jeho překážky.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 2373 odst. 2 zákona č. 89/2012 Sb., občanský zákoník, ve znění pozdějších předpisů, tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou a veškeré jejich dokumentace (dále souhrnně jen „Dílo"), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu) licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 27. června 2021 . . . . . .

Odkaz na tuto práci

Kovačič, Michal. *Analýza bezpečnosti DNS serverů*. Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2021.

Abstrakt

Tato práce se zabývá bezpečnostní analýzou běžně používaných DNS serverů, konkrétně jejich odolností proti vybraným známým útokům. Teoretická část začíná popisem protokolu DNS a pokračuje seznámením se s vybranými útoky, na které budu servery testovat. Tato část také obsahuje modelování hrozeb těchto útoků. V praktické části je popsána realizace simulace útoků pomocí emulátoru GNS3. Výsledkem práce je srovnání jednotlivých DNS serverů podle jejich zranitelnosti na dané útoky.

Klíčová slova DNS, bezpečnostní analýza, modelování hrozeb, GNS3, srovnání implementací

Abstract

This thesis is dedicated to security analysis of widely used DNS servers, focusing on their resistance to chosen known attacks. Theoretical part begins with introduction to DNS protocol and continues with familiarization of the most common attacks on DNS servers. It also contains their threat model. In the practical part is described the realization of chosen attacks in the GNS3 emulator. The outcome of this thesis is comparison between individual DNS servers, by their vulnerability to given attacks.

Keywords DNS, security analysis, threat modeling, GNS3, implementation comparison

Obsah

Seznam obrázků

Úvod

Jedním z hlavních rysů komunikace v počítačové síti je možnost kontaktování protější strany pomocí jejího doménového jména. Díky doménovým jménům si uživatel nemusí pamatovat IP adresu protějšku, kterého chce kontaktovat. Aby uživatelé mohli využívat doménových jmen a počítače mohli dále pracovat s IP adresami, je nutné umět tyto dvě entity mezi sebou překládat. O obousměrný překlad IP adres a doménových jmen se stará systém doménových jmen (DNS), který činí komunikaci v síti značně jednodušší a pohodlnější. Společně s výhodami přichází i nevýhody. Jednou z nich je zabezpečení.

DNS obsahuje mimo jiné takzvané „nameservery“, což jsou servery, které mimo jiné obsahují data o překladu všech IP adres v jim přidělené DNS zóně. Na světě existují veřejné DNS nameservery, jejíž zóna obsahuje veřejně dostupné IP adresy a privátní nameservery, které mají svou zónu omezenou pouze na privátní IP adresy. Tyto servery je nutné přidat do sítě jako další komponentu, což přirozeně zvětší povrch útoku dané sítě. Přesto mnoho nameserverů zůstává bez individuální ochrany, spoléhající se na ochranu sítě.

Důvodem, proč jsem zvolil za své téma právě útoky na DNS nameservery, je fakt, že málokdo se zabývá zabezpečením menších privátních nameserverů. Také jsem chtěl získat zkušenosti s běžnými útoky a vidět, jak fungují. Hlavní cíl práce je otestovat, zda často používané implementace nameserverů poskytují možnost ochrany proti běžným útokům a vzájemně jednotlivé implementace v tomto ohledu porovnat.

Struktura práce je rozdělena na rešeršní a praktickou část. Rešeršní část se zabývá teorií DNS, popisuje a kategorizuje známé zranitelnosti DNS nameserverů, obsahuje model hrozby těchto zranitelností a seznam aktuálně dostupných implementací systému DNS s jejich bezpečnostními vlastnostmi. V praktické části je popsána emulace DNS v emulátoru GNS3, kde používané nameservery jsou ty, které vzešly z analýzy. Poté jsou tyto servery otestovány na útoky, které vzešly z analýzy. Na závěr je k dispozici shrnutí schopnosti testovaných implementací odolat vybraným útokům.

Kapitola 1

Cíl práce

Cílem práce je vyzkoušet, zda běžně používané DNS nameservery poskytují ochranu proti běžným útokům. Konkrétně se tato práce zaměřuje na následující body:

1. **Analýza systému DNS, včetně používaných implementací nameserverů**
   Ke splnění výše uvedeného bodu je potřeba důkladně prostudovat uživatelskou a vývojářskou dokumentaci systému DNS a pečlivě analyzovat existující implementace nameserverů.

2. **Analýza známých útoků na službu DNS**
   Mimo souhrnu postupů a principů bude poskytnut jejich model hrozeb modelovaný pomocí standardních metodik.

3. **Návrh a postup realizace emulace systému DNS**
   Emulace systému DNS musí být natolik pružná, aby jednotlivé síťové komponenty představující nameservery byly jednoduše zaměnitelné. Návrh emulace bude reflektovat poznatky získané při analýze existujících implementací nameserverů. V práci bude detailně popsáno, jak emulovat každou jednotlivou použitou komponentu.

4. **Návrh a postup realizace testů odolnosti jednotlivých nameserverů**
   Emulace testů bude provedena v prostředí navrhnutého pro emulaci systém DNS. U každého testu na každý nameserver musí být zřejmé, jak útok probíhá a nakolik byl úspěšný.

5. **Srovnání DNS nameserverů podle jejich možností obrany na testované útoky**
   Závěrem práce bude shrnující tabulka obsahující výsledky dle jejich odolnosti.

Za své osobní cíle považuji:

1. **Ucelit a rozšířit své teoretické znalosti v oboru počítačových sítí**
   Počítačové sítě je jeden z oborů, který mě na studiu informačních technologií zajímá. Na ČVUT FIT není na tento obor kladen tak velký důraz jak bych si přál a proto jsem si vybral práci, ke jejíž vypracování je nutné rozumět základní teorii tohoto oboru.

2. **Rozšířit své povědomí o možnostech útoku na počítačovou síť**
   Podle mých dosavadních zkušeností platí, že v oboru bezpečnosti je nutné své znalosti neustále rozšiřovat. Tuto práci velice rád využiji jako možnost rozšířit svůj obzor v oboru, který studuji.

3. **Získat osobní zkušenost s útoky, které se naučím**
   Ačkoliv si velmi dobře uvědomuji, že emulované prostředí má od reálného prostředí velmi daleko, věřím, že vyzkoušení si útoků v emulovaném prostředí mi značně pomůže s pochopením daného útoku.

Vzhledem k povaze mých osobních cílů bude tato práce poměrně teoretičtější, než většina bakalářských prací na ČVUT FIT a nemyslím si, že bude využitelná širokou veřejností. Jsem si ovšem jistý, že práce poskytne dostatečné podklady pro kohokoliv, kdo by se zajímal o emulaci sítí s komplexními prvky, práci se systémem DNS, nebo zabezpečení systému DNS.

Kapitola 2

DNS

DNS je distribuovaný systém pro pojmenování koncových bodů v síti, které využívají IP adresu. Je detailně popsán v RFC 882[1] a RFC 883[2], které byly později aktualizovány na RFC 1034[3] a RFC 1035[4].
DNS usnadňuje komunikaci v síti uživatelům. Pro člověka je mnohem jednodušší zapamatovat si jméno koncového bodu počítačové sítě ve formě textového řetezce (např. fit.cvut.cz), než IP adresu daného koncového bodu. Zároveň, ostatní počítačové protokoly fungují na bázi IP adres. DNS se stará, aby si uživateli stačilo zapamatovat doménové jméno a aby zbytek protokolů mohl dále fungovat na bázi IP adres.
Před vznikem DNS byla jeho funkce vykonávána převodní tabulkou v souboru `hosts`, který mohl vypadat následovně:

```
127.0.0.1       localhost
192.168.0.3     example.com
192.169.0.4     example2.com
```

Na systém DNS se přešlo kvůli nepraktičnosti souboru `hosts`, který nezvádal provádět aktualizace s rostoucím počtem uživatelů.

2.1 Komponenty DNS

DNS se skládá ze tří primárních komponent:

- **Domain Name Space**, česky prostor doménových jmen, je stromová struktura, jejíž uzly označujeme jako „domény“. V každé doméně se vyskytuje její název a informace relevantní k jejímu názvu. Prostor doménových jmen se dělí na tzv. zóny, což jsou skupiny sousedících vrcholů, které obvykle spravuje jeden nameserver, nebo jeden cluster nameserverů v případě větších sítí[3]
- **DNS servery** (nameservery) obsahující informace o prostoru doménových jmen. Typicky, nameserver obsahuje kompletní informace o jemu přidělené

zóně prostoru doménových jmen a ukazatele na jiné nameservery, které obsahují zbytek informací o prostoru doménových jmen[3]

- **DNS resolvery** jsou programy které vystupují jako prostředník v komunikaci mezi serverem a klientem. Jejich úkolem je předložit klientův dotaz nameserveru a následně předložit klientovi odpověď, jenž z nameserveru extrahuje[3].

2.2 Hierarchie DNS

DNS je decentralizovaný systém, jehož hierarchii představuje stromová struktura s jedním kořenem. Tento kořen je označován jako root, česky kořenový uzel. Symbolizuje se tečkou. Kořenový uzel tvoří zónu 2.1, jenž je spravována takzvanými kořenovými nameservery. Z logického pohledu se na světě nachází 13 kořenových nameserverů, fyzicky se jedná o stovky serverů, které se chovají jako jeden z 13 logických nameserverů[5].
Domény, se kterými sousedí kořenový uzel prostoru doménových jmen se nazývají „domény nejvyššího řádu“ (anglicky top level domains – TLD). TLD mohou být tvz. generické domény (např. com, edu), nebo domény označující kód země (např. cz, sk). Seznam generických TLD se neustále rozšiřuje, např. tento rok byla vydána TLD „beauty“[1]. TLD také tvoří samostatné zóny, jenž spravují servery nejvyššího řádu. O českou doménu nejvyššího řádu „cz“ se stará organizace jménem „CZ.NIC“[2].
Všechny ostatní domény se označují jako poddomény. U poddomén nemusí platit, že každá doména tvoří zónu a je obvyklé, že jeden nameserver spravuje více poddomén. Hieararchie DNS je ilustrována na obrázku 2.1[3] (upraveno autorem).

2.3 Zónový soubor a zdrojový záznam

Zónový soubor je soubor nacházející se uvnitř nameserveru, který danou zónu spravuje. Obsahuje všechny informace o zóně, jenž server spravuje. Jeho nejdůležitější částí jsou zdrojové záznamy (resource records).
Zdrojový záznam nese informace o konkrétním koncovém bodě v zóně, např. mapování doménových jmen a IP adres. Dělí se na následující části:

- **Name** značí doménové jméno uzlu prostoru doménových jmen 2.1
- **Type** určuje typ obsahu dat zdrojového záznamu (sekce RData)
- **Class** specifikuje třídu obsahu zdrojového záznamu (sekce RData)

[1] Kompletní seznam TLD je k dispozici na `https://www.namecheap.com/domains/new-tlds/explore/`.

[2] Kompletní informace o této doméně jsou k dispozici na `https://nic.cz`.

- **Time to live (TTL)** je životnost zdrojového záznamu v sekundách
- **RDLength** udává délku obsahu zdrojového záznamu (sekce RData)
- **RData** reprezentuje obsah zdrojového záznamu. RData je nejdůležitější část celého zdrojového záznamu

Zónový soubor i zdrojový záznam je definován v RFC 1034[3] a RFC 1035[4]. Formát zdrojového záznamu je ilustrován na obrázku 2.2.

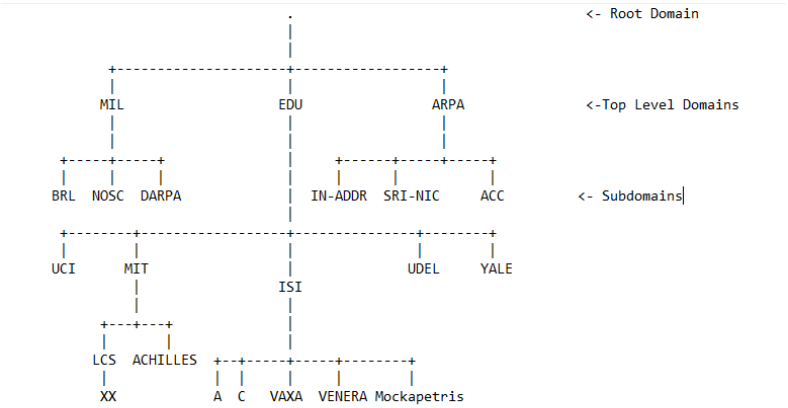
2.4 Doménové jméno

Doménové jméno je textový řetězec, který reprezentuje koncový bod v síti, nebo doménu. Definuje se jako posloupnost názvů domén od určitého uzlu až ke kořenu prostoru doménových jmen 2.1. Doménové jméno vzniklé z listu prostoru doménových jmen označujeme jako „plně kvalifikované doménové jméno“ [3].
Na obrázku 2.1 se vyskytuje mnoho doménových jmen, např. `ISI.EDU.` nebo `MIT.EDU.` Plně kvalifikovaná doménová jména z tentýž obrázku jsou např. `ACHILLES.MIT.EDU.` nebo `ACC.ARPA.`

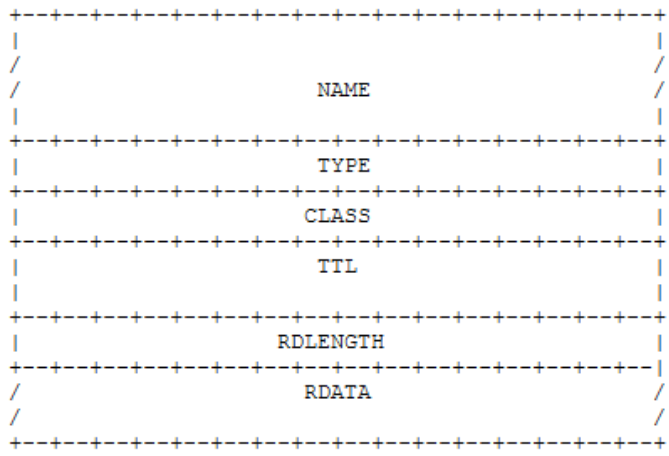
2.5 DNS zpráva

Pojmem DNS zprávy jsou myšleny všechny zprávy, které putují v systému DNS. Jsou to nejen požadavky uživatele na nameserver, ale i interní komunikace mezi jednotlivými nameservery. DNS zpráva se dělí na následující části:

Obrázek 2.1: Domain Name Space



Obrázek 2.2: Zdrojový záznam

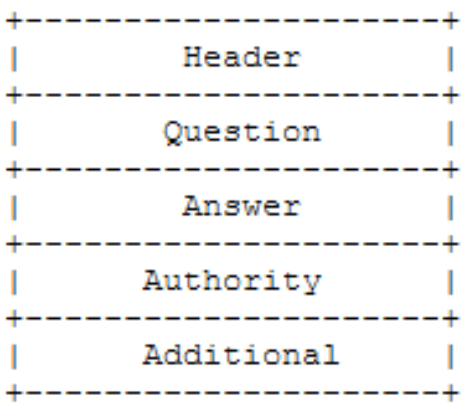


- **Header**, neboli hlavička zprávy obsahuje atributy specifikující strukturu dané zprávy (např. specifikují, zda zpráva je dotaz, nebo odpověď). Formát hlavičky je definován v RFC 1035[4], stejně jako formát zprávy
- **Question** popisuje dotaz, který byl položen nameserveru
- **Answer** obsahuje odpověď na dotaz v sekci „question“ ve formě zdrojového záznamu
- **Authority** obsahuje informace ohledně nameserveru spravujícího danou zónu. Pole má formát zdrojového záznamu
- **Aditional** je pole, kde se nachází doplňující informace ve formě zdrojového záznamu

Nejběžnějšími zprávami jsou „DNS dotazy“, neboli „DNS queries“. Jsou to dotazy položené uživatelem – překlad doménového jména na IP adresu, nebo překlad IP adresy na doménové jméno.
Formát DNS zprávy je definován v RFC 1034[3] a RFC 1035[4]. Je ilustrován na obrázku 2.3.
DNS zpráva může mít několik typů:

- **Typ A** obsahuje výsledek dotazu překladu IP adresy na doménové jméno.
- **Typ PTR** je inverzní obdoba A záznamu.
- **Typ CNAME** obsahuje alias pro jiné doménové jméno.
- **Typ CAA** upřesňuje certifikační autoritu, jenž mohla vydat doméně certifikát.

Obrázek 2.3: Formát DNS zprávy



- **Typ TXT** obsahuje libovolný textový řetězec.
- **Typ SRV** je upřesnění informací o dostupných službách na doméně.

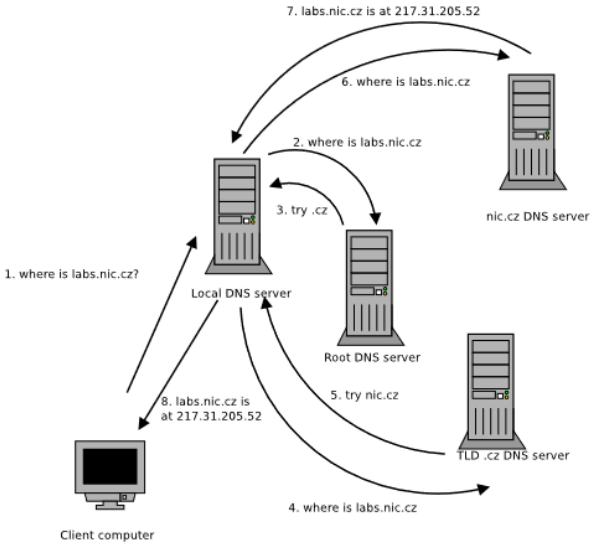
Více typů DNS zpráv a upřesňující popis je dostupný na `https://www.immagic.com/eLibrary/ARCHIVES/GENERAL/WIKIPEDI/W120423L.pdf`, včetně jejich definujících RFC dokumentů.

2.6 DNS resolution

DNS resolution je proces získání odpovědi na DNS query 2.5. Konkrétně se jedná buďto o překlad doménového jména na IP adresu, překlad IP adresy na doménové jméno, nebo o interní dotaz mezi jednotlivými DNS servery. Jako příklad a ukázku jak překlad probíhá uvádím překlad doménového jména „labs.nic.cz“ na IP adresu[6].

1. Klient zadá doménové jméno lokálnímu DNS resolveru.
2. V případě, že lokální DNS resolver přímo nezná odpověď, zeptá se kořenového DNS serveru na doménu cz.
3. Kořenový DNS server vrátí odpověď, ve které uvádí DNS server nejvyšší úrovně, který je zodpovědný za nejvyšší doménu (v našem případě doména .cz).
4. Lokální DNS resolver se zeptá daného serveru nejvyšší úrovně.
5. Server nejvyšší úrovně vrátí odpověď, ve které uvádí adresu nameserveru zodpovědného za doménu nic.cz. Tento server označuji v sekci 2.2 jako server nižší úrovně.
6. Lokální DNS resolver se zeptá daného serveru.

Obrázek 2.4: DNS Resolution



7. Daný server ví přesnou IP adresu zadaného doménového jména a může ji tedy v odpovědi vrátit lokálnímu DNS resolveru.

8. Lokální DNS resolver přepošle IP adresu klientovi.

Tento příklad je ilustrován na obrázku 2.4[6].
DNS resolution může být iterativní, nebo rekurzivní. Rekurzivní postup je znázorněn v ukázce. Iterativní postup je obdobný, s rozdílem, že každou jednotlivou odpověď každého DNS serveru resolver vrací klientovi, který se poté musí znovu zeptat resolveru. V dnešní době je skoro každý DNS resolution rekurzivní.

2.7 DNSSEC

DNSSEC je standardizované rozšíření protokolu DNS, které umožňuje zabezpečit veškeré DNS zprávy putující sítí proti podvržení pomocí digitálního podpisování. Je definovaný v RFC 2535[7].
V této práci se zabývám zabezpečením nameserverů bez rozšíření DNSSEC a proto nemá smysl toto rozšíření dlouze popisovat. Účelem této sekce je, aby čtenář vzal na vědomí existenci DNSSEC a aby věděl, co tento pojem znamená.

Kapitola 3

Dostupné implementace DNS nameserverů

V této kapitole se nachází jednoduchý popis nejčastěji používaných implementací služby DNS a jejich bezpečnostní vlastnosti. Na světě existuje nespočet veřejných implementací a jsem si jistý, že stejný počet soukromých implementací. V této kapitole zmíním pouze veřejně dostupné implementace, které jsou běžně používané a má smysl testovat jejich bezpečnost. Abych do práce nekopíroval kompletní bezpečnostní dokumentaci jednotlivých implementací, zmíním v kapitole pouze vybrané bezpečnostní vlastnosti.

3.1 BIND DNS

Stručné informace o BIND DNS

BIND DNS je zcela transparetní open-source implementace DNS protokolu poskytující službu DNS serveru a resolveru v jednom. Verze BIND4 a BIND8 jsou zastaralé a obsahují mnoho implementačních i bezpečnostních chyb. Tyto verze již nejsou udržované, ačkoliv stále fungují. S verzí BIND9 byla implementace kompletně přepsána a stále je udržována autory[3].

Bezpečnostní vlastnosti BIND DNS

BIND verze 9.0 a výše je považována za standard implementace DNS. Vývojáři implementaci pravidelně vylepšují a pravidelně opravují implementační i bezpečnostní chyby, teoreticky se tedy jedná o jednu z nejbezpečnějších implementací DNS, jaká ja k dispozici. Přehled základních bezpečnostních vlastností:

- Verze implementací nameserveru BIND 9.3 a všechny vyšší jsou zcela kompatibilní s rozšířením DNSSEC

[3]Kompletní informace jsou k dispozici na `https://www.isc.org/bind/`.

- Dovoluje i doporučuje omezit IP adresy, které mohou služby nameserveru využívat. Je také možné omezit adresy, pro které bude k dispozici rekurzivní DNS Resolution 2.6.
- Obsahuje možnost limitování počtu odpovědí na jednu adresu
- Loguje veškerý tok dat, který přes server přechází
- Obsahuje soubor `named.conf`, který specifikuje list klientů, jenž mohou požádat server o rekurzivní DNS Resolution. Pomocí tohoto souboru lze také omezit DNS Servery, od kterých implementace BIND přijme DNS zprávu. Rekurzivní resoluci lze zcela vypnout
- Obsahuje službu `SPLIT DNS`, jenž rozpůlí službu DNS na interní a externí část. Cílem služby je neprozradit strukturu interní sítě společnosti
- Implementace nabízí možnost rozeznávání tzv. „Bogus Domains“ a tím zabránit útokům typu NXDomain attack 5.1
- Implementace poskytuje možnost takzvaných DNS Cookies, které umožňují identifikovat zdroj informací. Tímto způsobem by implementace měla být schopna rozlišit podvrhnuté zprávy
- Implementace BIND se navenek může maskovat jako jiná verze této implementace.

Bezpečnostní vlastnosti implementace BIND jsem čerpal z oficiální dokumentace BIND[8] a ze zpravodaje `ITProToday`[9].

3.2 KNOT DNS

Stručné informace o KNOT DNS

KNOT DNS je open-source implemetace DNS serveru a resolveru v jednou vyrobená společností CZ.NIC. Touto společností je implementace aktivně udržována. KNOT DNS se pyšní vysokým výkonem, stabilitou a bezpečností[4].

Bezpečnostní vlastnosti KNOT DNS

Z bezpečnostního hlediska si implementace KNOT DNS nevede vůbec špatně. Stejně jako je tomu u BIND DNS, vývojáři pravidelně pracují na vylepšení a na opravách chyb implementaci KNOT DNS. Implementace mimo jiné:

- Nabízí úplnou kompatibilitu s rozšířením DNSSEC u všech podporovaných verzí

[4]Kompletní informace jsou k dispozici na `https://www.knot-dns.cz/`.

- Dovoluje snadno omezit IP adresy, které mohou služby nameserveru využívat.
- Obsahuje možnost limitování počtu odpovědí na jednu adresu
- Loguje veškerý tok dat, který přes server přechází
- Umožňuje identifikovat jiné nameservery pomocí techniky NSID
- Nabízí možnost rozeznávání tzv. „Bogus Domains“ a tím zabránit útokům typu NXDomain DoS 5.1.

Bezpečnostní vlastnosti KNOT DNS jsem čerpal z oficiální dokumentace[10].

3.3 CORE DNS

Stručné informace o CORE DNS

CORE DNS je open-source implementace DNS serveru napsaná v jazyce GO zaměřená na jednoduchost a rozšiřitelnost. Na rozdíl od ostatních implementací poskytuje pouze elementární funkce nameserveru – cokoliv navíc lze do implementace přidat v podobě rozšíření[5].

Bezpečnostní vlastnosti CORE DNS

CORE DNS člení většinu svých funkcí do rozšíření a pluginů, aby zachovala svou flexibilitu a jednoduchost. Bezpečnostní vlastnosti serveru bez pluginů nejsou oficiálně zdokumentované (dokumentace zde: https://coredns.io/manual/toc/). Oficiální podporované bezpečnostní pluginy nabízí:

- Rozšíření DNSSEC
- Access Control List, jenž nabízí možnost omezit IP adresy, které mohou požádat o DNS Resolution 2.6.
- Možnost zastavit DNS Resolution po časovém úseku
- Možnost automatického zotavení po selhání systému
- Randomizace pořadí zpráv typu A, AAAA a jejich odpovědí 2.5.
- Možnost sledování průběhu DNS Resolution

Kompletní seznam oficiálně podporovaných pluginů je k dispozici online[6].F

[5] Kompletní informace jsou k dispozici zde: `https://coredns.io/`.
[6] `https://coredns.io/plugins/`

3.4 dnsmasq

Stručné informace o dnsmasq

Dnsmasq je open-source implementace DNS a protokolu DHCP určený pro malé sítě. Ze všech zde uvedených implementací je nejjednodušší a klade nejnižší nároky na systém[7].

Bezpečnostní vlastnosti dnsmasq

Dnsmasq je z bezpečnostního hlediska černou ovcí DNS implementací. V nedávné minulosti bylo zdokumentováno mnoho bezpečnostních zranitelností, které oficiálně nebyly nikdy opraveny. Mezi zmíněné zranitelnosti patří např. DNS Cache Poisoning, na nějž byl server testován v lednu roku 2021[11].
Oficiální dokumentace dnsmasq obsahuje pouze instalační a konfigurační návod[8], bezpečnostní vlastnosti jsem našel až při zkoumání konfiguračních souborů. Mezi bezpečnostní vlastnosti dnsmasq patří:

- Nejnovější verze již jsou kompatibilní s rozšířením DNSSEC.
- Implemetace dovoluje snadno omezit IP adresy, které mohou služby nameserveru využívat. Dnsmasq je používán zároveň jako DHCP server a je možné omezit povolené adresy pouze na adresy v rozmezí přidělených adres pomocí DHCP.
- Implemetace nabízí možnost rozeznávání tzv. „Bogus Domains“ a tím zabránit útokům typu NXDomain DoS 5.1.

3.5 djbdns

Stručné informace o djbdns

Djbdns je implementace DNS serveru vytvořena jako reakce na bezpečnostní chyby ve starších verzích BIND DNS. Implementace se pyšní bezpečností do takové míry, že autor implementace v roce 2009 nabídl odměnu 1000$ za nalezení jakékoliv bezpečnostní zranitelnosti. Kompletní informace o implementaci včetně bezpečnostních vlastností jsou k dispozici v oficiální dokumentaci[12].

Bezpečnostní vlastnosti djbdns

Djbdns je implementace pro bezpečnostní nadšence. Autor slibuje její absolutní bezpečnost. V sekci návodu, kde autor popisuje její bezpečnost, jsou mimo jiné vypsány následující body:

[7] Kompletní informace na `https://thekelleys.org.uk/dnsmasq/doc.html`.
[8] K dispozici zde: `https://thekelleys.org.uk/dnsmasq/docs/dnsmasq-man.html`

- Skrytá paměť nemá práva na ovlivnění jiných funkcí systému a odstraňuje DNS query, pokud nepřijdou z předem specifikovaných IP adres
- Implementace nepřijímá odpovědi od serverů, na které předtím nebyl poslán dotaz
- DNS používá kryptograficky zabezpečený generátor náhodných čísel pro ID paketů a porty serveru
- Rekurzivní DNS Resolution 2.6 není podporována

3.6 Implementace vybrané pro testování

Pro testování jsem se rozhodl využít všech výše zmíněných implementací kromě djbdns. Všechny ostatní jsou open-source, volně dostupné a jednoduché na zprovoznění.

1. **BIND DNS** verze 9.3. BIND je de-facto standard pro implementaci protokolu DNS a proto předpokládám, že proti běžným útokům bude zabezpečený, nebo zabezpečitelný
2. **KNOT DNS** 3.0.5. Jelikož implementace soupeří o místo na trhu s BIND a výrobek pochází od společnosti, která spravuje doménu `cz`, předpokládám, že tato implementace bude také zabezpečená či zabezpečitelná proti všem běžným útokům
3. **CORE DNS** verze 1.8.3 bez pluginů. Ačkoliv implementace soustředí zabezpečení do rozšíření, aby zachovala svou jednoduchost, minimální obranu proti běžným útokům by mít měla
4. **dnsmasq** verze 2.85. Implementace bude testovaná, jakožto zástupce implementací protokolu DNS pro malé sítě

Veškeré testování proběhne na nameserverech se základním nastavením. Ve výsledcích tento fakt zhodnotím a zanalyzuji veškeré možnosti obrany na útok proběhlý v daném testu.
Implemntace djbdns nebude testovaná, protože se mi jí nepodařilo zprovoznit v testovacím prostředí. Vyžaduje ke korektní funkčnosti nástroj `ucspi-tcp`, který se mi pomocí instalačního návodu autora sice povedlo zprovoznit na svém počítači, ale přenést ho do testovacího prostředí se ukázalo být nemožné. Ke konkrétnímu problému jsem sice nenašel žádné informace, ale domnívám se, že nástroj k funkci potřebuje démona systemd, jehož funkce je uvnitř docker kontejnerů modifikovaná a nástroj tedy nemůže být kompatibilní s mnou navrženým testovacím prostředím.

Kapitola 4

Modelování hrozeb

Modelování hrozeb je proces identifikace slabin systému. Výsledkem modelování hrozeb je seznam hrozeb spolu s jejich vlastnostmi (např. dopad na uživatele) – model hrozeb. Je zvykem modelovat hrozby pomocí metodik, které zaručují kompletní přehled o celém povrchu útoku na systém. Kategorizací hrozeb se zabývá metodika STRIDE, odhadem závažnosti hrozeb se zabývají metodiky DREAD a CVSS.

4.1 STRIDE

STRIDE kategorizuje hrozby do šesti kategorií, které jsou identifikované pomocí jednotlivých písmen jména metodiky. Konkrétně se jedná o[13]:

- **S**poofing of Identity – podvržení identity uživatele
- **T**ampering with Data – neautorizované pozměnění dat
- **R**epudiation – popření transakce uživatelem
- **I**nformation Disclosure – únik informací
- **D**enial of Service – odepření služby
- **E**levation of Privilege – neautorizované zvýšení oprávnění

Hledání hrozeb pomocí metodiky STRIDE znamená zaměřit se na každou kategorii hrozeb, kterou metodika STRIDE popisuje a hledat hrozbu této kategorie v celém systému.
Ke kategorizaci útoků v této práci budu používat metodiku STRIDE. Narozdíl od typické kategorizace pomocí této metodiky si dovolím útoky přiřadit do více kategorií současně. Metodiku jsem si zvolil, protože je jednoduchá, výstižná, známá a mám s ní zkušenosti.

4.2 DREAD

Metodika DREAD hodnotí hrozbu z hlediska pěti kategorií, které jsou podobně jako u STRIDE identifikovány pomocí písmenek jména metodiky. Jedná se o[13].

- **D**amage potential – potenciál škody
- **R**eproducibility – reprodukovatelnost chyby
- **E**xploitability – množství úsilí vynaložené k útoku
- **A**ffected users – počet a důležitost uživatelů, na které má zranitelnost dopad
- **D**iscoverability – odhad toho, jak snadné je zranitelnost najít

Metodika ohodnotí hrozbu v každé kategorii celým číslem v intervalu $(1-10)$ a poté vypočítá vážený průměr všech pěti ohodnocení. Tento průměr vyjadřuje závažnost hrozby.

4.3 CVSS

CVSS je metodika klasifikace závažnosti hrozeb reagující na nedostatky metodiky DREAD. CVSS verze 3.0 z roku 2015 je široce používaný průmyslový standard. Od verze 2.0 se liší převážně upravením škály jednotlivých metrik. Základní metriky CVSS 3.0 mají více-úrovňovou škálu:

- **Attack Vector** popisuje „směr“, ze kterého může útok přijít
  Škála: network/adjacent network/local network/physical access
- **Attack Complexity** udává komplikovanost a spolehlivost útoku
  Škála: low/high
- **Privileges Required** charakterizuje nutná oprávnění útočníka k provedení útoku
  Škála: none/low/high
- **User Interaction** udává, zda úspěch útoku vyžaduje akci uživatele s přístupem do systému
  Škála: none/required
- **Scope** popisuje, zda úspěšný útok ovlivňuje jinou část systému, než je zranitelná část systému
  Škála: unchanged/changed
- **Confidentiality Impact** je metrika popisující, na kolik útok ovlivní důvěrnost dat
  Škála: none/low/high

- **Integrity Impact** popisuje na kolik útok ovlivní integritu dat
  Škála: none/low/high
- **Avaiability Impact** popisuje na kolik útok ovlivní dostupnost služby
  Škála: none/low/high

Metriky i škála jsou převzaté ze slidů přednášky předmětu BI-BEK[13].
Celková závažnost hrozby je stejně jako u metodiky DREAD číslo v intervalu $(1 - 10)$, ale počítá se podle složitějšího vzorce. Pro výpočet výsledku závažnosti hrozby podle CVSS existují veřejně dostupné nástroje[9]. Výsledek se dá upřesnit pomocí časových metrik a metrik prostředí.
Pro modelování závažnosti hrozeb v této práci budu používat metodiku CVSS 3.0, bez upřesnění výsledků pomocí časových metrik a metrik prostředí. CVSS jsem si vybral, protože je považována za průmyslový standard, používá se relativně jednoduše a protože s ní mám zkušenosti.
Verzi CVSS 3.0 jsem si vybral, protože k výpočtu celkové závažnosti hrozby hodlám používat internetový nástroj vydaný organizací "FIRST"[10]. Pro výpočet celkové hrozby pomocí CVSS 3.1 sice také existují nástroje, ale mnou používaný nástroj je jediný, který dovoluje vložit škálu dovnitř hypertextového odkazu a tím ukázat výpočet celkové závažnosti hrozby pomocí pouze odkazu.

[9] např. `https://nvd.nist.gov/vuln-metrics/cvss/v2-calculator?calculator&.0`
[10] `https://www.first.org/cvss/calculator/3.0`

Kapitola 5

Model hrozby DNS nameserverů

Cílem této kapitoly je zanalyzovat známe zranitelnosti systému DNS se zaměřením na DNS nameservery a vytvořit jejich modely hrozeb. Při tvoření tohoto modelu hrozeb se omezím pouze na známé a ověřené zranitelnosti, na které se útočí běžně.
Celkový souhrn hrozeb je na konci kapitoly 5.7.

5.1 Denial of Service (DoS)

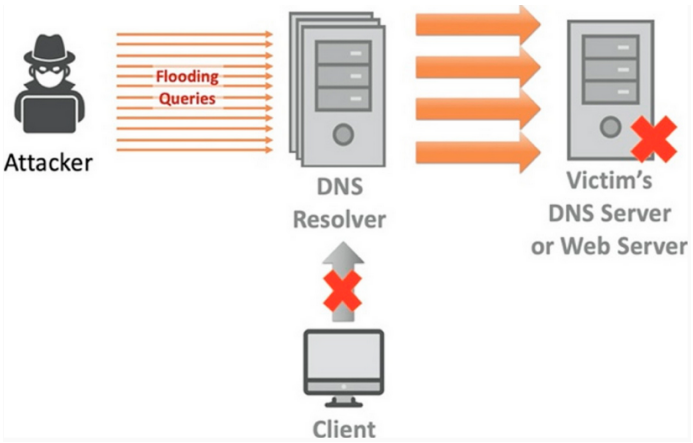
Komponenta je ohrožena hrozbou DoS pokud je možné odepřít uživatelům její funkčnost neoficiální cestou. V případě DNS se dá tvrdit, že nameserver nebo resolver je ohrožen, pokud útočník může odepřít uživatelům službu DNS.

Útok typu DoS a definice zranitelnosti DoS

Útoky typu DoS cílí na odepření služby běžným uživatelům přetížením komponenty. V případě útoky typu DoS na službu DNS se jedná o vytížení nameserveru pomocí jeho zahlcení DNS dotazy. Útok typu DoS nejčastěji bývá následujících typů:

- **Přímý DoS** zahltí server nadměrným množstvím „legitimních“ dotazů, které vytíží veškeré výpočetní kapacity serveru a tím brání zpracování dotazů uživatele
- **Slow/Sloth DoS** útok cílí k odepření služby tím, že zaměstnává všechna dostupná vlákna serveru dotazy, na které trvá dlouho odpovědět
- **Bogus Domain DoS/útok NXDomain** zaplaví DNS server dotazy na překlad neexistujících doménových jmen. Je podobný přímému DoS útoku, ale využívá faktu, že DNS Resolution je rekurzivní, tudíž neexistující doménová jména trvá déle zpracovat[14]

Obrázek 5.1: DoS Attack



- **Phantom Domain Attack** posílá DNS serveru dotazy, jenž k překladu potřebuje nedostupný nameserver (např. útočníkův privátní). Principiálně je podobný Bogus Domain útoku[14]

Obrázek 5.1[15] ilustruje typický průběh přímého útoku DoS na službu DNS.

Model hrozby DoS

Kategorizace DoS pomocí metodiky STRIDE 4.1 není náročná – pro DoS metodika obsahuje vlastní kategorii.
Závažnost hrozby podle metodiky CVSS 4.3 je následující:

- Attack Vector: network
- Attack Complexity: low
- Privileges Required: none
- User Interaction: none
- Scope: unchanged
- Confidentiality impact: none
- Integrity impact: none
- Avaiability impact: low

Celková hodnota CVSS: 5.3[11]

[11] `https://www.first.org/cvss/calculator/3.0#CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:L`

5.2 DNS Amplification

Útok typu DNS Amplification a definice zranitelnosti DNS Amplification

Útok DNS Amplification využívá faktu, že DNS Resolution je rekurzivní proces 2.6. Nameservery mohou být buďto součástí útoku (ať už dobrovolně nebo nedobrovolně), nebo jeho cílem. Pokud jsou součástí útoku, tak se snaží exponencinálně zvětšit velikost dotazů odeslaných útočníkem a tyto zvětšené dotazy následně odeslat na útočníkův cíl[14].. Pokud jsou jeho cílem, tak se nameserveru jeví, že je pod útokem typu DoS. Kromě typického útoku DNS Amplification může nameserver být součástí DNS Reflection útoku, který nezvětšuje velikost síťového provozu, ale má za cíl zamaskovat původní IP adresu útočníka místo zvětšení.
Nameserver je ohrožen hrozbou DNS Amplification, pokud je možné jej nedobrovolně využít ke zvětšení síťového provozu putující k cíli útoku.

Model hrozby DNS Amplification

Pokud DNS server pod naší kontrolou není cílem útoku, není ohrožen. Nejedná se tedy o zranitelnost, která by byla kategorizovatelná, nebo ke které by se dal vytvořit model hrozby pomocí CVSS. Pokud by náš server byl cílem, útok se navenek tváří jako přímý DoS útok a hrozba je tedy stejná jako při napadení přímým útokem DoS 5.1.

5.3 Packet sniffing

Útok typu Packet sniffing a definice zranitelnosti Packet sniffing

Služba DNS není ve výchozím stavu šifrovaná a uživatel si nemůže ověřit integritu dat, ani identitu odesílatele. Správně umístěný útočník může odchytit komunikaci a bez problémů si ji přečíst. Útočník provádějící Packet sniffing data pouze čte, nijak je nemění. Pokud by útočník data měnil, jednalo by se o „Man-in-the-middle“ útok.
DNS nameserver je ohrožený hrozbou Packet Sniffing, pokud data, která odesílá nedokáže ochránit před nechtěným čtením.

Model hrozby packet sniffing

Podle metodiky STRIDE se Packet sniffing očividně řadí do kategorie information disclosure. Závažnost útoku podle CVSS:

- Attack Vector: local network
- Attack Complexity: high
- Privileges Required: none
- User Interaction: none
- Scope: changed
- Confidentiality impact: low
- Integrity impact: none
- Avaiability impact: none

Celková hodnota CVSS: 4.0[12]

5.4 Man-in-the-Middle (MiM)

Útok typu MiM a definice zranitelnosti MiM

Útok typu MiM je principiálně podobná hrozbě Packet sniffing 5.3. Jediný rozdíl nastává v tom, že útočník pozměňuje pakety, které mu projdou pod rukama. Nameserver je tedy zranitelný proti útokům typu MiM pokud nedokáže ochránit odesílaná data proti jejich neautorizovanému pozměnění.
Z definice útoků MiM je Packet Sniffing také součástí této kategorie. Hrozby Packet Sniffing a ostatní MiM jsem rozdělil proto, že obrané mechanismy proti čtení a pozměnění dat mohou být jiné a bylo by dobré ochránit DNS pakety proti čtením i změnám.

Model hrozby MiM

Metodika STRIDE řadí MiM do kategorie tampering with data. Jelikož při uvedení této metodiky 4.1 jsem si dovolil řadit útoky do více kategorií, zařadím MiM také do kategorie Spoofing, kvůli schopnosti útočníka vydávat se za nameserver.
Závažnost útoku podle CVSS:

- Attack Vector: network
- Attack Complexity: high
- Privileges Required: none
- User Interaction: none

[12] `https://www.first.org/cvss/calculator/3.0#CVSS:3.0/AV:N/AC:H/PR:N/UI:N/S:C/C:L/I:N/A:N`

- Scope: changed
- Confidentiality impact: low
- Integrity impact: low
- Avaiability impact: none

Celková hodnota CVSS: 5.4[13]

5.5 DNS Cache Poisoning

Útok typu DNS Cache Poisoning a definice zranitelnosti DNS Cache Poisoning

DNS Cache Poisoning je asi nejznámější zranitelnost DNS resolverů. Využívá faktu, že DNS resolver, který má nějaký záznam ve skryté paměti nijak neověřuje jeho integritu a odesílá ho na opakovaný dotaz znovu. Pokud útočník dostane do skryté paměti zlomyslný záznam, DNS resolver bude tento záznam používat a přesměrovávat uživatele na špatnou IP adresu. DNS resolver je zranitelný, pokud se nedokáže ochránit proti vsunutí nevalidních záznamů do své skryté paměti.
Útočník vloží svůj záznam do skryté paměti tím, že pošle nameserveru zfalšovanou odpověď na právě probíhající DNS Resolution 2.6, která se tváří jako validní a která dorazí dřív, než opravdová odpověď. Cílový resolver si nemůže ověřit, že odpověď nepřišla od tázaného nameserveru, ale útočníka, tudíž ji přijme a uloží si ji do skryté paměti.

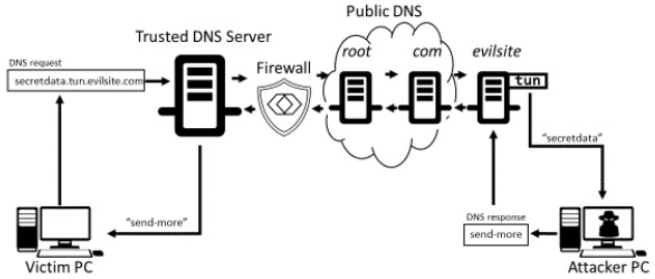
Model hrozby DNS Cache Poisoning

Pomocí metodiky STRIDE se útok řadí do kategorie spoofing. Útočník se při zfalšovaní odpovědi vydává za jiný nameserver. Navíc, do skryté paměti většinou vsune záznam, který odkazuje běžné uživatele na jeho službu, která se za původní službu vydává.
Závažnost útoku podle CVSS:

- Attack Vector: network
- Attack Complexity: high
- Privileges Required: none
- User Interaction: none

[13] `https://www.first.org/cvss/calculator/3.0#CVSS:3.0/AV:N/AC:H/PR:N/UI:N/S:C/C:L/I:L/A:N`

Obrázek 5.2: DNS Tunneling



- Scope: changed
- Confidentiality impact: low
- Integrity impact: high
- Avaiability impact: none

Celková hodnota CVSS: 7.5[14]

5.6 DNS Tunneling

Útok DNS Tunneling a definice zranitelnosti DNS Tunneling

DNS Tunneling nastává, pokud útočník obejde firewall sítě skrz protokol DNS. DNS server je zranitelný, pokud lze s jeho pomocí obejít firewall pro nepovolenou síťovou komunikaci.
DNS tunneling je útok, který využívá důvěryhodnosti služby. DNS dotazy se obvykle mohou volně pohybovat skrz firewall a nejsou nijak kontrolovány. Pro pochopení útoku uvádím jeho typický příklad: útočník si zařídí vlastní DNS server, nad kterým bude mít plnou kontrolu. Následně je nutná spolupráce uživatele – uživatel musí spustit malware, čehož se dá dosáhnout např technikou phishing. Tento malware následně komunikuje s útočníkem pomocí DNS query – posílá zašifrovaná data ve formě DNS query a přijímá data (příkazy) ve formě DNS odpovědi. Tímto způsobem může útočník obejít firewall a získat kompletní nadvládu nad počítačem uživatele.[14].
Útok DNS Tunneling je ilustrován na obrázku 5.2[16].

Model hrozby DNS Tunneling

DNS Tunneling definován na začátku této sekce je pomocí metodiky STRIDE obtížně kategorizovatelné. Kategorie metodiky STRIDE, která se nejvíce blíží

[14] `https://www.first.org/cvss/calculator/3.0#CVSS:3.0/AV:N/AC:H/PR:N/UI:N/S:C/C:L/I:H/A:N`

správné kategorizace DNS Tunneling je dle mého názoru Information disclosure. Ze sítě, jejíž firewall je obcházen sice neunikají data, ale zato útočník narušuje soukromí uživatelů uvnitř.
Závažnost útoku podle CVSS:

- Attack Vector: network
- Attack Complexity: high
- Privileges Required: none
- User Interaction: required
- Scope: changed
- Confidentiality impact: low
- Integrity impact: none
- Avaiability impact: none

Celková hodnota CVSS: 3.4[15]

5.7 Souhrn modelů hrozeb známých útoků

Attack	S	T	R	I	D	E	CVSS
DoS					✓		5.3
Amplification							
Packet Sniffing				✓			4.0
MiM	✓	✓					5.4
Cache Poisoning	✓						7.5
DNS Tunneling					✓		3.4

5.8 Útoky vybrané na testování

Pro účely testování byly vybrány následující útoky:

1. **Přímý DoS** 5.1. Tento útok budu testovat, protože je známý, rozšířený, relativně jednoduchý na testování, lze jej testovat v emulovaném prostředí a každý nameserver by se proti přímému DoS měl umět ubránit

[15] `https://www.first.org/cvss/calculator/3.0#CVSS:3.0/AV:N/AC:H/PR:N/UI:R/S:C/C:H/I:H/A:H`

2. **DNS Amplification** 5.2. V emulovaném prostředí budu testovat, nakolik je nameserver v základním nastavení využitelný pro amplifikaci síťového provozu. Schválně jsem použil slovo "nakolik", protože nameserver v základním nastavení by se teoreticky proti tomuto útoku neměl být schopný bránit. V kapitole zabývající se výsledky 9 experimentů sepíšu, jaké možnosti obrany implementují jednotlivé nameservery proti tomuto útoku a podle toho rozhodnu, nakolik jsou zranitelné.

3. **DNS Cache Poisoning** 5.5. Resolvery jednotlivých implementací nameserverů by měly být schopné odmítat podvržené odpovědi na DNS Resolution. Podobně jako je tomu u DNS Amplification, v základním nastavení v prostředí, které vytvořím pro útok se nameserver teoreticky nemá šanci ubránit. V kapitole zabývající se výsledky experimentů sepíšu, jaké možnosti obrany implementují jednotlivé nameservery proti tomuto útoku a podle toho rozhodnu, nakolik jsou zranitelné.

Ostatní útoky jsou buďto nevhodné pro testování (např. útok DNS Tunneling), nebo obtížně emulovatelné tak, aby prostředí bylo co nejreálnější (např. útok NXDomain).

Kapitola 6

Emulace testovacího prostředí

Primárním cílem této práce je otestovat vybrané implementace DNS nameserverů proti vybraným útokům. K tomu je potřeba prostředí, ve kterém budu jednotlivé útoky simulovat. Tato kapitola popisuje, jak jsem toto testovací prostředí emuloval.

6.1 Emulace počítačové sítě

K emulaci sítě pro účely své práce jsem si vybral nástroj GNS3. GNS3 je open source emulátor síťového softwaru, který dovoluje simulaci malých síťových topologií na uživatelově počítači. Je využíván jak studenty připravující se na zkoušky, tak globálními společnostmi jako je např. NASA[17].
Tento emulátor jsem si vybral převážně kvůli jeho flexibilitě a jednoduchosti používaní. Narozdíl od jiných emulátorů GNS3 poskytuje jak rozsáhlé možnosti pro emulaci koncových bodů sítě a jednoduchou integraci programu Wireshark, tak možnosti tvoření vlastních komponent. Pro testování budu používat verzi 2.2.20, nainstalovanou na distribuci Kali Linuxu verze 2020.4.

6.2 Emulace síťových komponent

Jelikož potřebuji k účelům testování mimořádné komponenty, potřeboval jsem si je vlastnoručně vytvořit. GNS3 podporuje několik možností integrace vlastnoručně vytvořených komponent – přes VirtualBox, VMWare, nebo Docker. Jelikož experimenty budu provádět na operačním systému na základě debianu a všechny komponenty, které si potřebuji vytvořit také potřebují operační systém na základě debianu (konkrétně Ubuntu 20.04), vybral jsem si pro tvorbu komponent nástroj Docker.
Docker je sada produktů, jenž poskytuje jednotné rozhraní pro izolaci aplikací. Poskytuje možnost používat software uvnitř prostředí nazývaného kontejner. Tento kontejner obsahuje pouze aplikace a soubory nutné pro běh soft-

waru, které nejsou obsaženy v uživatelově operačním systému[16]. Při instalaci nástroje na oerační systém Kali Linux jsem postupoval podle oficiálního návodu[17].
Při experimentech jsem Docker využíval pro tvoření virtuálních strojů. Pro tento účel není Docker zamýšlen a tedy budu mnohdy používat nezvyklé praktiky. Např. při vytváření DNS nameserverů nestačí stáhnout si oficiální Docker image, kterou vývojáři nabízí, ale je potřeba vytvořit si virtuální stroj, na kterém zprovozním implementace nameserverů jako na standardním počítači. Pro spuštění docker image v interaktivním režimu lze použít přepínače `-it`, které spustí image v kontejneru s terminálovým přístupem.

Následuje seznam komponent, které jsem při experimentech použil a návod k jejich vytvoření. Všechny vlastnoručně vyrobené komponenty byly vytvořeny ze šablony image Ubuntu 20.04 stažené pomocí příkazu `docker pull ubuntu`. Na touto image jsem si stáhl základní linuxové aplikace pomocí balíčkovacího nástroje `apt-get` – nano, gcc, ifconfig, dnsutils, ping, nano atd. a poté ji rozšiřoval o aplikace nutné k práci dané komponenty. Po vypnutí kontejneru lze jeho obsah uložit jako image pomocí příkazu `docker commit`.

6.2.1 BIND9 DNS nameserver

Vývojáři BIND DNS umožňují stáhnout si implementaci pomocí balíčkovacího nástroje `apt-get`. Konkrétně se jedná o příkaz `apt-get install bind9`.

6.2.2 KNOT DNS nameserver

Vývojáři KNOT DNS také umožňují stáhnout si implementaci pomocí balíčkovacího nástroje `apt-get`. Konkrétně se jedná o příkaz `apt-get install knot`.

6.2.3 CORE DNS nameserver

Narozdíl od předchozích implementací CORE DNS není součástí žádného balíčkovacího systému. Vývojáři poskytují pouze předkompilovaný binární soubor[18], který je k chodu implementace nutný spustit. Tento soubor jsem si stáhl a překopíroval do běžícího kontejneru pomocí příkazu `docker cp`.

6.2.4 dnsmasq nameserver

Dnsmasq je stejně jako BIND DNS a KNOT DNS součástí balíčkovacího nástroje `apt-get` a tudíž lze stáhnout pomocí příkazu `apt-get install dnsmasq`.

[16] Více informací o nástroji Docker je k dispozici zde: `https://www.docker.com`

[17] Ten je dostupný z `https://www.kali.org/docs/containers/installing-docker-on-kali/`.

[18] K dispozici ke stáhnutí zde : `https://github.com/coredns/coredns/releases/tag/v1.8.3`

6.2.5 Switch

Switch používaný pro účely testovaní není vytvořen pomocí nástroje docker, ale využívám ten, který je poskytnut uvnitř GNS3 v základní instalaci.

6.2.6 Router

Router také není vytvořen pomocí nástroje docker. V testech používám model CISCO c7200, jehož image je k volně dispozici ke stáhnutí[19].

6.2.7 Počítač útočníka (ATTACKER)

Jako počítač oběti je použita šablona Ubuntu obohacená o útočný skript konkrétní pro daný útok. Tyto skripty blíže popisuji v kapitole o testovacím prostředí 8.

6.2.8 Počítač oběti (VICTIM)

Jako počítač oběti je použita šablona Ubuntu bez žádných úprav.

6.2.9 Počítač simulující webový server (WEBSERVER)

Jako počítač simulující webový server je použita šablona Ubuntu bez žádných úprav.

[19] Já tento image stáhl zde: `https://www.careercert.info/new-cisco-ios-version-124-collection/`.

Kapitola 7

Nastavení komponent v topologii

V této kapitole popisuji nastavení jednotlivých komponent po jejich vytvoření a zařazení do síťové topologie.

7.1 Nastavení BIND DNS

Implementace DNS nameserveru BIND používá 4 konfigurační soubory v adresáři `/etc/bind` a k tomu několik konfiguračních souborů, které popisují zóny. Jedná se o následující soubory:

- **named.conf** obsahující komentář s odkazem na dokumentaci BIND9 a 3 řádky includů:

  ```
  include "/etc/bind/named.conf.options";
  include "/etc/bind/named.conf.local";
  include "/etc/bind/named.conf.default-zones";
  ```

  Tento soubor je považován za hlavní konfigurační soubor který není doporučeno ho nijak měnit.

- **named.conf.options**, ve kterém se nachází konfigurační možnosti pro službu BIND9. Můj vyplněný `named.conf.options` přikládám v ilustraci 7.1.

- **named.conf.local**, kam vkládáme lokální zóny a určujeme jejich nastavení, bez nastavení samotného překladu. Pro účely testování jsem si v souboru vytvořil následující zónu:

```
zone "local"{
type master;
file /etc/bind/db.local;
};
```

Soubor popisující zónu, konkrétně `/etc/bind/db.local` je potřeba si vytvořit. Jeho tvorbu popíši v následujících odstavcích.

- **named.conf.default-zones** popisuje předem připravené zóny ve stejném formátu jako `named.conf.local`. Pro účely testování není potřeba soubor měnit.

Po správném nastavení všech konfiguračních souborů bylo třeba vytvořit si zónový soubor 2.3 `/etc/bind/db.local`. Zónový soubor implementace bind odpovídá standardnímu zónovému souboru definovaného v RFC 1035[4]. Mé jednoduché vyplnění je ilustrováno na obrázku 7.3.

7.2 Nastavení KNOT DNS

Na rozdíl od implementace BIND, KNOT je znatelně přímočařejší. Používá jeden konfigurační soubor `/etc/knot/knot.conf`, ve kterém se ve výchozím stavu nachází zakomentářovaný náčrt jednoduchého nastavení. Pro své účely jsem tento náčrt promazal a upravil na formu ilustrovanou na obrázku. Poté jsem si vytvořil složku `zones` a v ní soubor `local.zone`, do které jsem vložil stejné nastavení zóny jako do zónového souboru implementace BIND9 7.3 (zónové soubory KNOT DNS také splňují standard definovaný v RFC 1035[4])

Obrázek 7.1: /etc/bind/named.conf.options

```
options {
        directory "/var/cache/bind";

        // If there is a firewall between you and nameservers you want
        // to talk to, you may need to fix the firewall to allow multiple
        // ports to talk.  See http://www.kb.cert.org/vuls/id/800113

        // If your ISP provided one or more IP addresses for stable
        // nameservers, you probably want to use them as forwarders.
        // Uncomment the following block, and insert the addresses replacing
        // the all-0's placeholder.

        // forwarders {   uncomment for regular usage
        //      8.8.8.8
        //      8.8.4.4
        // };

        //========================================================================
        // If BIND logs error messages about the root key being expired,
        // you will need to update your keys.  See https://www.isc.org/bind-keys
        //========================================================================
        dnssec-validation no;
        listen-on { any; };
        listen-on-v6 { none; };
        recursion yes;

        auth-nxdomain no;

};
```

7.3 Nastavení CORE DNS

Ze všech nameserverů je CORE nejoriginálnější. Používá jeden konfigurační soubor, který je potřeba si vyrobit – `Corefile`. Syntaxe tohoto konfiguračního soubory je zcela jedinečná, ale naštěstí plně dostupná v oficiální maunálu CORE DNS[18]. Pro mé testovací prostředí je zapotřebí zcela minimální Corefile:

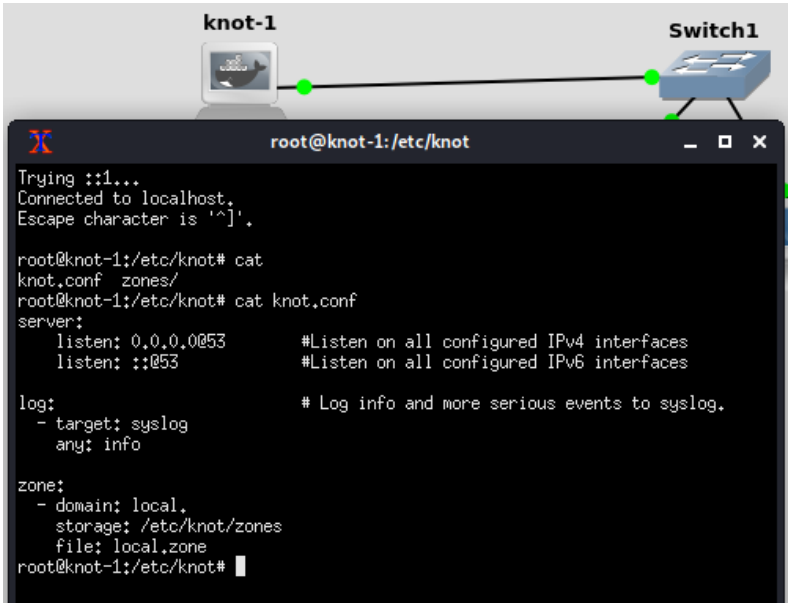
```
. {

    root /etc/coredns      ;dir containing zone files
    file db.local local {  ;zone and zone file
                           ;Syntax: file [filename] [zone name]
    }
}
```

Poté bylo potřeba vytvořit složku `/etc/coredns` a v ní zónový soubor `db.local`, který obsahuje zónovou konfiguraci dle RFC 1035 7.3[4].

7.4 Nastavení dnsmasq

Implementace dnsmasq funguje téměř „out-of-the-box“. Služba DHCP, kterou dnsmasq poskytuje navrch ke službě DNS je ve výchozím stavu deaktivovaná

Obrázek 7.2: /etc/knot/knot.conf



(nemá určený rozptyl IP adres, které by poskytovala). Ke zprovoznění implementace stačí přidat IP adresy lokální sítě do souboru `/etc/hosts` ve formě dvojice `([IP adresa], [doménové jméno])`.

7.5 Nastavení ostatních komponent

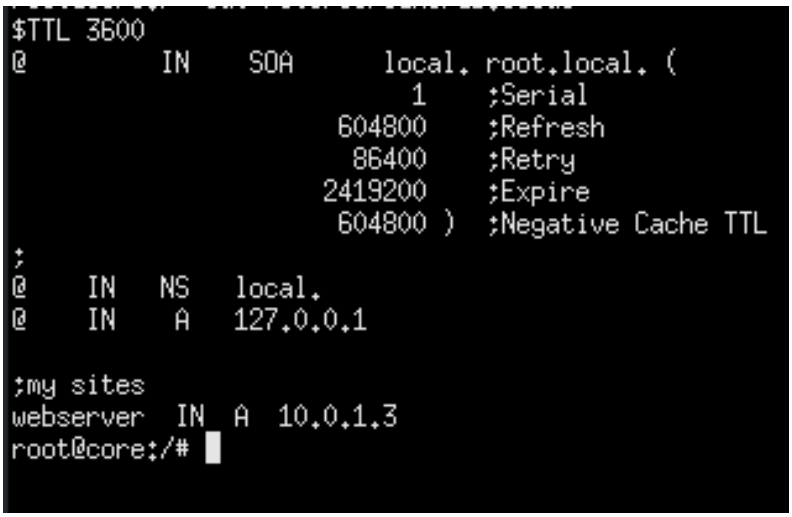
Ostatní komponenty fungují téměř okamžitě. Je potřeba pouze pozměnit nastavení lokálního resolveru 2.1, aby jednotlivé komponenty věděli, kde mají hledat nameserver pro překlad dotazů:

```
echo "nameserver 10.0.1.2 > /etc/resolv.conf"
```

Pokud by bylo potřeba dotazovat se z DNS nameserveru na překlady, resolver by se nastavil následovně:

```
echo "nameserver 127.0.0.1 > /etc/resolv.conf"
```

Obrázek 7.3: DNS zone file (příklad, v jednotlivých experimentech se může lišit)



Kapitola 8

Prostředí pro útoky

V této kapitole je popsáno, jak probíhali jednotlivé útoky – jejich topologie, doplňková nastavení komponent, průběh a podmínky, kdy se nameserver považuje za zranitelný proti útoku.

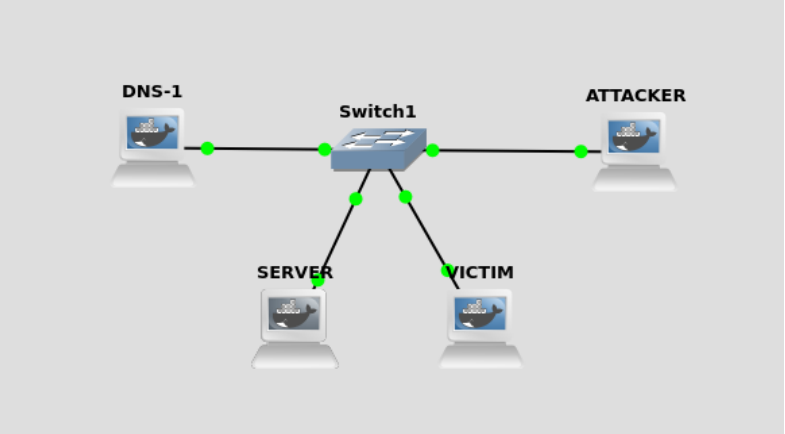
8.1 Přímý DoS

Topologie a popis komponent

Experiment pro přímý DoS bude probíhat na topologii, jenž je ilustrovaná na obrázku 8.1.Jednotlivé komponenty představují:

- **DNS nameserver** simuluje testovaný nameserver
  IP adresa: 10.0.1.2
- **SERVER** simuluje server, na který obsahuje nameserver záznam
  IP adresa: 10.0.1.3

Obrázek 8.1: Prostředí pro útok DoS



- **VICTIM** simuluje oběť
  IP adresa: 10.0.1.4
- **ATTACKER** simuluje počítač útočníka
  IP adresa: 10.0.1.5

Testování funkčnosti topologie proběhlo pomocí nástroje `nslookup` 8.2.
Na počítači útočníka se nachází útočný program `dnsflood`, který neustále odesílá DNS dotazy na předem specifikovaný nameserver[20].
Počítač simulující DNS nameserver má uměle zmenšenou maximální frekvenci procesoru pomocí nástroje `cpufreq`, konkrétně na hodnotu minimální frekvence, která je ještě stále podporována hardwarem mého počítače. Tato frekvence se dá zjistit pomocí stejného nástroje (`cpufreq-info`).

Průběh experimentu

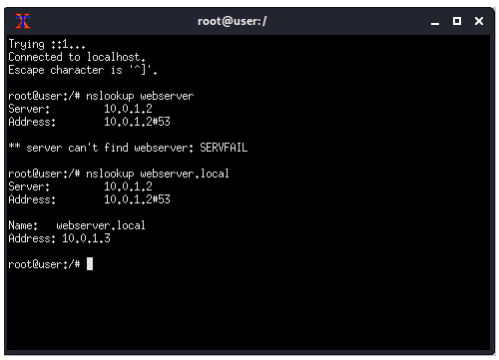
Experiment probíhal následovně:

1. Na nameserveru se vyčistí skrytá paměť
2. Útočník spustí program `dnsflood` s cílem na nameserver
3. Použitím programu Wireshark lze snadno ověřit, zda nameserver přijímá dotazy 8.3
4. Oběť požádá o překlad

Úspěšnost útoku se hodnotí podle míry zpomalení překladu oběti. Pokud bude nameserver mít stále stejnou odezvu před útokem i po útoku, prohlašuji ho za odolný proti přímému útoku DoS. Pokud útok výrazně zpomalí, nebo úplně zastaví odezvu nameserveru, není nameserver odolný proti přímému DoS. Experiment proběhl opakovaně, abych rozšířil velikost vzorku.

[20] Zdrojový kód tohoto programu je volně k dispozici ke stáhnutí zde: `https://github.com/nickwinn/dns-flood`

Obrázek 8.2: Testování funkce nastavení



Obrázek 8.3: DoS program test

No.	Time	Source	Destination	Protocol	Length	Info
90833	11.006135	10.0.1.5	10.0.1.2	DNS	64	Standard query 0xbe06 A test
90834	11.006184	10.0.1.5	10.0.1.2	DNS	64	Standard query 0xa774 A test
90835	11.006232	10.0.1.5	10.0.1.2	DNS	64	Standard query 0xe43f A test
90836	11.006281	10.0.1.5	10.0.1.2	DNS	64	Standard query 0x3996 A test
90837	11.006330	10.0.1.5	10.0.1.2	DNS	64	Standard query 0x9ca2 A test
90838	11.006407	10.0.1.5	10.0.1.2	DNS	64	Standard query 0x0cf8 A test
90839	11.006456	10.0.1.5	10.0.1.2	DNS	64	Standard query 0x4e41 A test
90840	11.006523	10.0.1.5	10.0.1.2	DNS	64	Standard query 0x3412 A test
90841	11.006573	10.0.1.5	10.0.1.2	DNS	64	Standard query 0xd7ff A test
90842	11.006642	10.0.1.5	10.0.1.2	DNS	64	Standard query 0xe7bc A test
90843	11.006710	10.0.1.5	10.0.1.2	DNS	64	Standard query 0x6f24 A test
90844	11.006759	10.0.1.5	10.0.1.2	DNS	64	Standard query 0x9783 A test
90845	11.006811	10.0.1.5	10.0.1.2	DNS	64	Standard query 0xfb74 A test
90846	11.006882	10.0.1.5	10.0.1.2	DNS	64	Standard query 0x6d35 A test
90847	11.006935	10.0.1.5	10.0.1.2	DNS	64	Standard query 0x7120 A test
90848	11.006996	10.0.1.5	10.0.1.2	DNS	64	Standard query 0x54b8 A test
90849	11.007074	10.0.1.5	10.0.1.2	DNS	64	Standard query 0x3d49 A test
90850	11.007142	10.0.1.5	10.0.1.2	DNS	64	Standard query 0xaa69 A test
90851	11.007218	10.0.1.5	10.0.1.2	DNS	64	Standard query 0x80eb A test
90852	11.007267	10.0.1.5	10.0.1.2	DNS	64	Standard query 0x4aa3 A test
90853	11.007375	10.0.1.5	10.0.1.2	DNS	64	Standard query 0xc133 A test
90854	11.007424	10.0.1.5	10.0.1.2	DNS	64	Standard query 0xc5fc A test
90855	11.007472	10.0.1.5	10.0.1.2	DNS	64	Standard query 0x5d19 A test

8.2 DNS Cache poisoning

Topologie a popis jednotlivých komponent

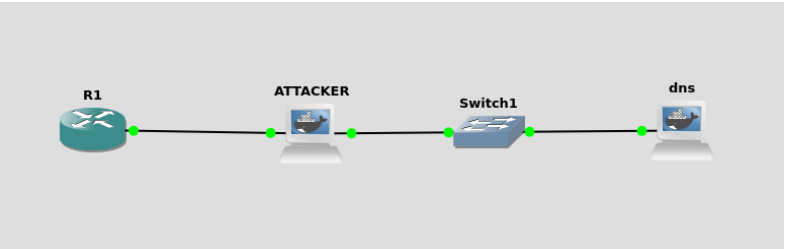
DNS Cache Poisoning bude probíhat na topologii, ilustrované na obrázku 8.4. Jednotlivé komponenty představují následující:

- **Router** simuluje bránu do vedlejší sítě, kde by se teoreticky měl nacházet forwarder testovaného DNS serveru
  IP adresa: 10.1.0.1
- **ATTACKER** simuluje počítač útočníka a zároveň slouží jako jednosměrný most pro přepravu paketů
  IP adresy: 10.1.0.4 na rozhraní vlevo, 10.1.0.5 na rozhraní vpravo
  Výchozí brána: 10.1.0.1
- **DNS** simuluje testovaný nameserver
  IP adresa: 10.1.0.2
  Výchozí brána: 10.1.0.1

Z důvodu zjednodušení experimentu je útočník uprostřed DNS komunikace, aby byl schopen odchytit přesné znění dotazu[21]. To je vyřešeno tím, že na

[21] V reálném prostředí by mohl být téměř kdekoliv.

Obrázek 8.4: Topologie pro DNS Cache Poisoning



počítači útočníka probíhá přemosťování paketů mezi jednotlivými rozhraními pomocí nástroje `tcpbridge`[22]. Před útokem útočník přestane předávat zprávy a začně odpovídat na DNS dotazy svou vlastní odpovědí s podvrženým identifikátorem. Toto by v praxi znamenalo přerušení veškerého síťového provozu, ale pro experiment je to dostačující. Tato situace bude fungovat pouze po dobu, dokud má DNS server uvnitř své ARP tabulky záznam o MAC adrese routeru.
Na počítači útočníka se také nachází útočný skript. Pojem "skript"je pro tuto situaci lehce přehnaný – jedná se o zfalšovanou DNS odpověď, ke které jsem akorát připojil ID DNS transakce a odeslal. Tento skript vypadá následovně:

```
#extract transaction ID
tcpdump -i eth0 -c 1 'port 53' | tr ' ' '.' \
| tr '+' '.' | cut -d. -f15 > temp

#convert transaction ID to hexadecimal
value=`cat temp | tr -d '\n'`
printf "%x" "$value" > response

#merge transaction ID with the rest of the DNS response to be sent
cat fakeDNS >> response

#send the DNS response to destination
cat response | tr -d '\n' | netcat -u 10.1.0.2 53
```

Při vytváření jsem bral v potaz, že předem znám jméno a průběh DNS query. V reálném prostředí bych si zprávu nemohl takto předpřipravit. Zprávu jsem si předpřipravil po jednotlivých bajtech v programu `Midnight Commander`. Proces tvorby dat a jeho výsledek je ilustrován na obrázku 8.5. Pro přehlednost experimentu přikládám popis jednotlivých bajtů na obrázku:

```
Question bytes:
  85 80 - standart response, no error  #DNS resolution return code
  00 01 00 01 - 1 question RR, 1 answer RR
  00 00 00 00 - 0 authority RR, 0 aditional RR
  09 77 65 62 73 65 72 76 65 72 00 - name len, name (webserver), C-string endpoint
  00 01 00 01 - message type A, class IN

Answer bytes:
  C0 0C - name
  00 01 00 01 - type A, class IN
  00 00 00 00 - TTL
  04 0A 01 00 06 0A - data length, data (ip address to insert into cache)
```

Průběh experimentu

Experiment bude probíhat následovně:

1. Na počítači útočníka spustím útočný skript

[22] Vzhledem k neuspokojivému stavu dokumentace nástroje si dovolím přidat přesný příkaz: `tcpbridge -i eth0 -I eth1 &`

Obrázek 8.5: Předpřipravená DNS zpráva pro DNS Cache Poisoning

```
/home/michal/fakeDNS    0x00000000                    0%
00000000 85 80 00 01  00 01 00 00 ........
00000008 00 00 09 77  65 62 73 65 ...webse
00000010 72 76 65 72  00 00 01 00 rver....
00000018 01 C0 0C 00  01 00 01 00 ........
00000020 00 00 00 00  04 0A 01 00 ........
00000028 06 0A                    ..
```

2. Testovaný DNS nameserver odešle dotaz na překlad, který nezná. Tento dotaz odputuje za výchozí bránou a nameserveru se nedostane žádná legitimní odpověď

3. Skript spuštěný na počítači útočníka automaticky odešle zfalšovanou odpověď

4. Útočník pomocí nástroje `nslookup` otestuje, zda jeho odpověď byla akceptována a zda se povedlo dostat škodlivý záznam do paměti.

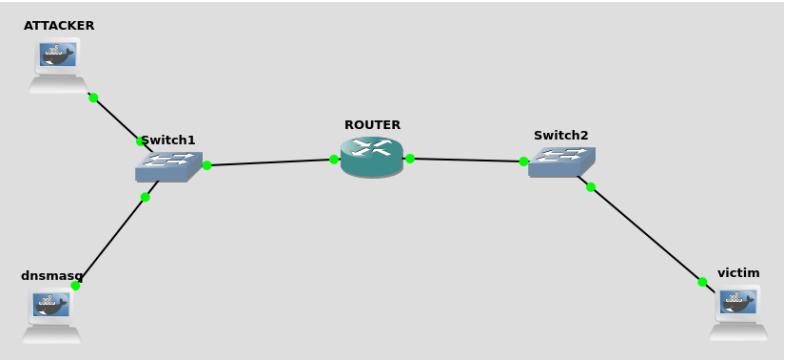
Tento experiment popisuje velice zjednodušenou verzi útoku a měl by ukázat, že každá implementace v základním nastavení je zranitelná. Cílem experimentu je ukázat, že útok je proveditelný a měl by být brán v potaz.
Odolnost jednotlivých implementací hodnotím podle toho, jak obtížné je nameserver ochránit proti DNS Cache poisoning.

8.3 DNS Amplification

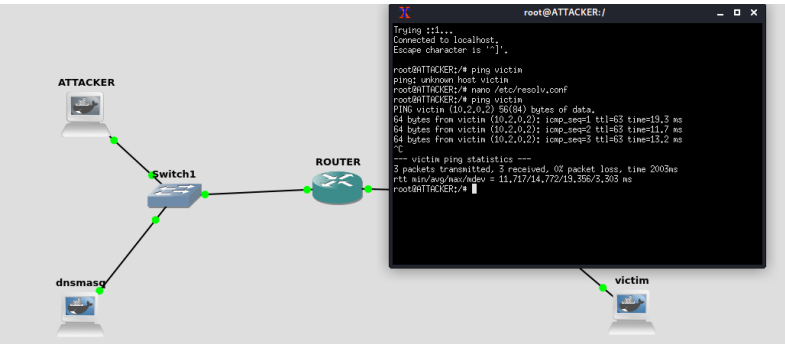
Topologie a popis jednotlivých komponent

Topologie pro útok DNS Amplification lze vidět na ilustraci 8.6. Obsahuje 2 podsítě – síť s adresou 10.1.0.0 nalevo a síť s adresou 10.2.0.0 napravo.

Obrázek 8.6: Prostředí pro útok DNS Amplification



Obrázek 8.7: Prostředí pro útok DNS Amplification



Všechny komponenty v dané podsíti mají nastavenou výchozí bránu na dané rozhraní rozdělujícího routeru. Komponenty sítě jsou následující:

- **ROUTER** tvořící hranice sítí
  IP adresy: 10.1.0.1 (na rozhraní v síti 10.1.0.0) a 10.2.0.1 (na rozhraní v síti 10.2.0.0)
  Směrovací protokol: Routing Information Protocol (RIP)
- **ATTACKER** je počítač simulující útočníka
  IP adresa: 10.1.0.3
- **Testované DNS** je počítač simulující DNS nameserver, který bude testován, na DNS Amplification
  IP adresa: 10.1.0.2
- **VICTIM** je počítač simulující DNS nameserver, na který bude DoS vycházející z DNS Amplification cílen. Tento nameserver neslouží žádnému jinému účelu
  IP adresa: 10.2.0.2

Testování prostředí pomocí nástroje `nslookup` je ilustrováno na obrázku 8.7. Na počítači simulující útočníka se nachází program `dnsamp`, jehož úkolem je odesílání předem připravených dotazů na testované DNS s falešnou IP adresou, tak, aby testované DNS poslalo odpověď na DNS nameserver oběti[23].

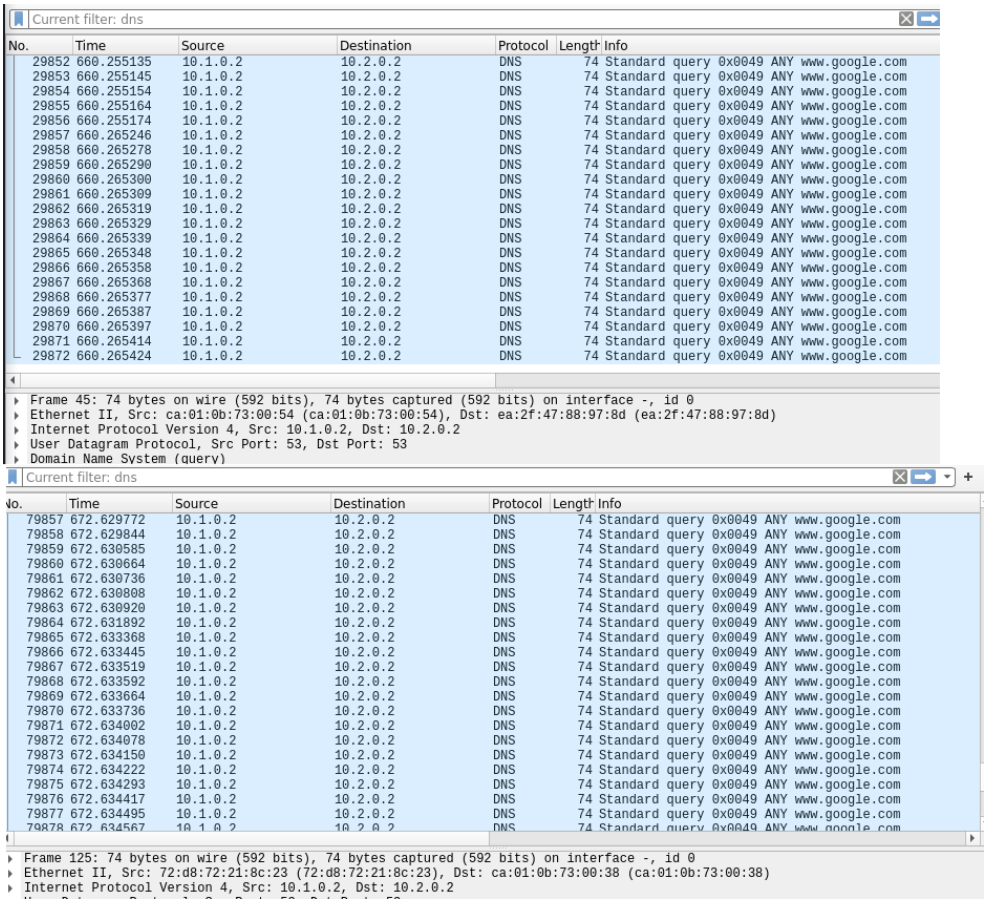
Průběh experimentu

Experiment probíhal následovně:

1. Na testovaném nameserveru se vyčistí skrytá paměť

[23] Zdrojový kód tohoto programu jsem stejně jako v případě přímého DoS nenapsal já. Program, včetně podmínek jeho šíření je k dispozici ke zde: `https://github.com/ethanwilloner/DNS-Amplification-Attack`

Obrázek 8.8: Příklad úspěšné amplifikace DNS provozu



2. Útočník spustí program `dnsamp` se správnými parametry

3. Pomocí programu Wireshark zjistíme rozdíl mezi velikostí dat odeslanými útočníkem a velikostí dat směřující na nameserver oběti 8.8

Podobně jako je tomu u experimentu popisující DNS Cache poisoning, tento experiment také popisuje útok, který by měl uspět u všech jednotlivých nameserverů.
Odolnost nameserveru hodnotím podle toho, jak jednoduché je zařídit, aby velikost amplifikace byla zanedbatelná. Cílem je, aby vnější útočník nebyl schopen využít DNS uvnitř lokální sítě k útoku.

Kapitola 9

Výsledky

Tato kapitola obsahuje výsledky jednotlivých experimentů spolu s analýzou možností obran, které implementace proti útokům poskytuje.

9.1 Výsledky experimentu pro útok přímého DoS

U implementací BIND DNS, KNOT DNS a kupodivu i dnsmasq se neprojevil žádný rozdíl odezvy před a po útoku a tedy jsou proti tomuto útoku odolné. U CORE DNS jsem ve třech z dvaceti případů zaznamenal zpoždění odezvy trvající dobu sotva postřehnutelnou okem. Jelikož se odezva lišila o minimální časový okamžik a nemůžu s jistotou určit, zda se tak stalo kvůli útoku nebo výkonnostní anomálií mého PC (např. vyhledávání aktualizace Windows), prohlašuji nameserver za odolný proti útoku přímého DoS.

Tento výsledek není nijak překvapující. Implementace BIND DNS a KNOT DNS jsou široce používané a na takto primitivní útok musí být připraveny.
Implementace CORE DNS prodělala v roce 2018 bezpečnostní prohlídku od společnosti CURE53[24], při které bylo nalezeno, že implementace má s útoky typu DoS problémy. Dá se předpokládat, že vývojáři tento problém opravily a v budoucnu si na něho dají pozor. Výsledek mého experimentu potvrzuje, že je tomu tak.
Dnsmasq byla jediná implementace, u které jsem se bál, že bude na přímý DoS zranitelná. Verze nižší než 2.81 zranitelné byly poněvadž program obsahoval úniky paměti[25]. Já testoval verzi 2.85, která je již v pořádku jak podle dokumentace, tak podle mých testů.

[24]kompletní info této prohlídky je k dispozici zde: `https://coredns.io/2018/03/15/cure53-security-assessment/`

[25]Detaily o zranitelnosti dostupné zde: `https://www.tenable.com/plugins/nessus/136411`.

9.2 Výsledky experimentu pro útok DNS Cache Poisoning

Při tomto experimentu se dalo očekávat, že všechny implementace v základním budou proti útoku zranitelné. V reálném prostředí je útok provést mnohem těžší. Útočník se např. nemusí nacházet uprostřed útoku a musí tak hádat, kdy má zfalšovanou odpověď odeslat. Také si zprávu nemůže předpřipravit tak jednoduše jako při mém experimentu.

Nejúčinnější metoda ochrany proti tomuto útoku je DNSSEC. Všechny testované implementace nabízí plnou kompatibilitu s tímto rozšířením. Ačkoliv se v této práci zabývám nameservery bez rozšíření DNSSEC, mohu všemi deseti doporučit tuto možnost – je spolehlivá a vyřeší i další problémy kromě DNS Cache poisoning.

Implementace BIND DNS poskytuje možnost detekce zfalšovaných odpovědí pomocí DNS Cookies. V reálném prostředí je zfalšování DNS Cookies prakticky nemožné a implementace se tedy dokáže proti DNS Cache poisoning bránit.
KNOT DNS poskytuje funkci DNS Cookies pouze jako rozšíření. Implementace se ve svém základním nastavení proti DNS Cache Poisoning bránit neumí.
Implementace CORE DNS a dnsmasq se umí proti útoku bránit pouze uvedenými univerzálními metodami a tedy jsou zranitelné proti DNS Cache poisoning.

9.3 Výsledky experimentu pro útok DNS Amplification

Stejně jako při experimentu pro útok DNS Cache Poisoning ukázal tento útok, že při základním nastavení jsou proti tomuto útoku zranitelné všechny nameservery. Toto je mnohem větší problém než při DNS Cache Poisoning, jelikož tento útok je podstatně lehčí na provedení.

Dobré zprávy jsou, že obrana proti zneužití pro DNS Amplification je extrémně jednoduchá. Stačí omezit počet odpovědí, které nameserver může odeslat na IP adresu v určitém časovém úseku. Tato obrana se nazývá Response-rate-limiting (RRL).
Implementace BIND DNS i KNOT DNS poskytují možnost RRL v základním nastavení a jsou tedy odolné proti využití na DNS Amplification.
Implementace CORE DNS poskytuje tuto funkci pouze jako rozšíření, v základním nastavení tedy není odolná proti využití na DNS Amplification.
Implementace dnsmasq možnost RRL neposkytuje vůbec. Jediný argument, který by mohl hrát implementaci dnsmasq do karet je, že tato implemen-

tace poskytuje zároveň se službou DNS službu DHCP a možnost odesílat odpovědi pouze na adresy, které předtím pomocí DHCP přidělila. Pokud je takto nastavená, neposkytuje možnost DNS Amplification na kohokoliv, ale pouze na uživatele v lokální síti. Na druhou stranu, útok je stále možné provést a možnosti omezit IP adresy, na které je možné odesílat odpověď poskytují všechny testované implementace, i když podstatně složitěji. Tento argument není dost na to, abych mohl prohlásit implementaci za odolnou proti využití na DNS Amplification.

9.4 Shrnutí výsledků a porovnání jednotlivých implementací

Experimenty v této práci ukázaly, že jakkoliv bezpečná implementace nameserveru je při nesprávném nastavení zranitelná. Následuje pořadí implementací podle jejich bezpečnosti vycházející z experimentů v této práci:

1. **BIND DNS**
   Implementace BIND DNS ukázala, že je standardem pro službu DNS právem. Ve svém základním nastavení obsahuje možnosti účinné obrany proti všem experimentovaným útokům.

2. **KNOT DNS**
   Jediný důvod, proč je KNOT DNS na až druhém místě je ten, že DNS Cookies ještě nejsou součástí standardní implementace a tedy v úplném základu není schopný se bránit proti DNS Cache Poisoning. Oproti BIND DNS je ale mnohem jednodušší a intuitivnější pro uživatelské používaní.

3. **CORE DNS**
   Implementace CORE DNS bez pluginů je opravdu nezabezpečená. Kdybych testoval verzi s pluginy, nejspíš skončí na prvním místě – v pluginech se nachází možnost prevence pro každý jednotlivý útok. Důvod, proč není poslední je fakt, že uživatelé využívající tuto implementaci alespoň mají možnost pluginy si stáhnout a používat

4. **dnsmasq**
   Implementace dnsmasq bez rozšíření DNSSEC poskytuje naprosto nulové zabezpečení. Jediný důvod, proč je implementace stále používaná, ačkoliv má o mnoho silnější konkurenci je fakt, že je extrémně jednoduchá na používaní a že je využívána v sítích, ve kterých je největším obraným prostředkem její nedůležitost.

Následuje tabulka, která shrnuje, zda se daná implementace dokáže bránit proti danému útoku:[26]

[26] ✓ = implementace se umí bránit

	Přímý DoS	DNS Cache Poisoning	DNS Amplification
BIND DNS	✓	✓	✓
KNOT DNS	✓		✓
CORE DNS	✓		
dnsmasq	✓		

Pro lidi, kteří spravují malou domácí síť a potřebují použít implementaci systému DNS doporučuji BIND DNS nebo KNOT DNS. BIND DNS má nejrozsáhlejší dokumentaci a vede si velmi dobře z bezpečnostního hlediska. Je také rozšiřitelná, takže na internetu existuje mnoho diskuzí ohledně veškerých možných chyb, které mohou uživatele potkat. KNOT DNS je sice méně rozšířená, ale podstatně jednodušší ke spravování. Navíc, s integrací veškerých pluginů se v bezpečnosti vyrovná implementaci BIND DNS.
CORE DNS je implementace, se kterou je spoustu práce. Pro to, aby produkovala stejný výsledek jako jiné implementace je potřeba integrace spousty pluginů. Tuto implementaci mohu doporučit leda síťovým nadšencům.
S implementací dnsmasq není skoro žádná práce, ale v reálném světě kvůli svým omezeným možnostem nemá uplatnění. Jediné, na co bych implementaci mohl doporučit je pro studenty počítačových sítí, pro emulaci a testování topologií je ideální.

Závěr

Na závěr okomentuji míru splnění cílů práce:

- **Analýza systému DNS, včetně používaných implementací nameserverů**
  Na začátku práce jsem zanalyzoval protokol a systém DNS. Vzhledem k široké dostupnosti materiálů na internetu nebyl problém zanalyzovat systém doménových jmen z hlediska jeho principu a funkcionality. Analýza jednotlivých implementací byla náročnější, převážně analýza bezpečnostních vlastností jednotlivých implementací se ukázala být tvrdým oříškem kvůli nekompletní, nebo zcela chybějící dokumentaci. Nakonec jsem dle mého názoru cíl splnil – co jsem nevyčetl z dokumentace, to se buď nacházelo v konfiguračních souborech, nebo jsem zjistil při testování.

- **Analýza známých útoků na službu DNS**
  Jelikož systém DNS je ve své původní formě nezabezpečený a často dostává výjimku v průchodu skrz firewall, není překvapující, že na něho již dlouho existují útoky. Tyto útoky jsou velmi dobře popsané a na internetu jsou často k dispozici amatérské nebo i profesionální programy, které útok realizují. Pro jejich kategorizaci a model hrozeb jsem použil rozšířené metodiky. Cíl byl definitivně splněn.

- **Návrh a postup realizace emulace systému DNS**
  Návrh, který je v práci prezentován se ukázal být mnohem náročnější, než se původně zdál. To převážně proto, že musel obsahovat nezvyklé komponenty, které samy o sobě představovaly komplexní systémy. Po správném nastavení návrh splňoval všechny požadavky, které na něho byly kladeny a cíl tedy byl splněn.

- **Návrh a postup realizace testů odolnosti jednotlivých nameserverů**
  Ačkoliv jsem byl kvůli omezenému výkonu na emulaci prostředí v některých

částech značně omezen, díky extrémně pružnému prostředí emulace systému DNS a existence nespočetně mnoha nástrojů byl návrh a realizace testů jedním z lehčích bodů. Veškeré prováděné testy byly funkční, dle mého nejlepšího svědomí korektní a měly jednoznačné měřítko úspěšnosti. Tento cíl byl také splněn.

- **Srovnání DNS nameserverů podle jejich odolnosti na testované útoky**
  K úplnému splnění tohoto cíle jsem popsal metody, jak se jednotlivé implementace mohou proti útokům bránit. Také jsem přidal svůj osobní uživatelský názor ke všem jednotlivým implementacím.

Rád bych se také vyjádřil ke splnění svých osobních cílů:

1. **Ucelit a rozšířit své teoretické znalosti v oboru počítačových sítí**
   Myslím, že ke splnění tohoto cíle jsem si vybral nejlepší možné téma. Díky studiu protokolu DNS a emulace sítí obsahující nameservery se tento cíl splnil bez toho, abych se na něj musel jakkoliv soustředit.

2. **Rozšířit své povědomí o možnostech útoku na počítačovou síť**
   Splnění tohoto cíle bylo součástí zadání, splněn být tedy musel. Kromě toho jsem získal nové poznatky o standartních metodikách modelování a kategorizace hrozeb.

3. **Získat osobní zkušenost s útoky, o kterých se naučím**
   Veškeré útoky, se kterými jsem experimentoval v emulovaném prostředí se mi povedly provést. V analýze nabízených možností obran jednotlivých nameserverů jsem se dozvěděl, jak se účinně bránit. Při analýze bezpečnostních vlastností nameserverů mi několikrát došlo, jak se proti útokům nebránit. Jsem si jistý, že po této zkušenosti se má schopnost uvažovat nad bezpečnostní několikrát znásobila.

Valnou většinu cílů, které jsem si dal jsem splnil. Tato práce ukazuje průběh jejich plnění a jejich výsledky.
Mrzí mě, že prostředí, ve kterém jsem experimenty prováděl, není kompatibilní implementaci djbdns – byla by příjemná změna pracovat se softwarem, který byl vyvíjen s důrazem na bezpečnost, narozdíl od běžných softwarů, kde bezpečnost je dodělávána až po vývoji, nebo v extrémním případě až po vydání. Emulace této implementace je jediný cíl, který považuji za nesplněný. Celkem práci považuji za úspěch a to primárně kvůli stoprocentnímu splnění mých osobních cílů.

Literatura

[1] Mockapetris, P. V.: RFC882: Domain names - concepts and facilities. November 1983. Dostupné z: `https://tools.ietf.org/html/rfc882`

[2] Mockapetris, P. V.: RFC883: Domain names - implementation and specification. November 1983. Dostupné z: `https://tools.ietf.org/html/rfc883`

[3] Mockapetris, P. V.: RFC1034: Domain names - concepts and facilities. November 1987. Dostupné z: `https://tools.ietf.org/html/rfc1034`

[4] Eastlake, D.: Rfc2535: Domain Name System Security Extensions. March 1999. Dostupné z: `https://datatracker.ietf.org/doc/html/rfc2535`

[5] Root: Root Server Technical Operations Assn. Dostupné z: `http://www.root-servers.org`

[6] Pohořelý, M.: *Detection of DNS Anomalies viaData Mining Analysis of NetworkTraffic*. Diplomová práce, Fakulta Informačních Technologií, ČVUT, 2014.

[7] Mockapetris, P. V.: Rfc1035: Domain names-implementation and specification. November 1987. Dostupné z: `https://www.ietf.org/rfc/rfc1035.txt`

[8] ISC: *BIND 9 Documentation*. Dostupné z: `https://kb.isc.org/docs/aa-01031`

[9] Zhou, T.: Secure Your BIND DNS Service. *ITPro Today*, 10 2001. Dostupné z: `https://www.itprotoday.com/security/secure-your-bind-dns-service`

[10] CZ.NIC: *KNOT DNS 2.9.9 Documentation*. Dostupné z: `https://www.knot-dns.cz/docs/2.9/singlehtml/`

[11] Zorz, Z.: Dnsmasq vulnerabilities open networking devices, Linux distros to DNS cache poisoning. *Help Net Security*, 1 2021. Dostupné z: https://www.helpnetsecurity.com/2021/01/19/dnsmasq-vulnerabilities/

[12] Bernstein, D. J.: *djbdns: Domain Name System tools*. Dostupné z: https://cr.yp.to/djbdns.html

[13] Kokeš, J.: Úvod do bezpečného kódu, modelování hrozeb, 2021. Dostupné z: https://kib-files.fit.cvut.cz/bi-bek/recordings/2021-pred-01.mp4

[14] David Doucette, G. R.: DNS Security. 08 2018. Dostupné z: https://www.f5.com/content/dam/f5/corp/global/pdf/agility/agility2018/dns_security.pdf

[15] Kim TH, R. D.: A survey of domain name system vulnerabilities and attacks. *J Surveill Secur Saf*, 2020. Dostupné z: http://dx.doi.org/10.20517/jsss.2020.14

[16] Lifinski, R.: How hackers use DNS Tunneling to own your network. *Cynet*, 06 2020. Dostupné z: https://www.cynet.com/attack-techniques-hands-on/how-hackers-use-dns-tunneling-to-own-your-network/

[17] Galaxy Technologies: *Getting started with GNS3 — GNS3 documentation*. Dostupné z: https://docs.gns3.com/docs/

[18] *KNOT DNS 2.9.9 Documentation*. Dostupné z: https://www.knot-dns.cz/docs/2.9/singlehtml/

Příloha **A**

Seznam použitých zkratek

DNS Domain Name System

DHCP Dynamic Host Configuration Protocol

MiM Man in the Middle

ARP Address Resolution Protocol

RRL Response Rate Limiting

Příloha **B**

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu flash disku
- `sim` ..... adresář obsahující soubory simulace
  - `dos.gns3` ..... soubor obsahující prostředí pro útok DoS
  - `amp.gns3` ... soubor obsahující prostředí pro útok DNS Amplification
  - `cachepoison.gns3` . soubor obsahující prostředí pro útok DNS Cache Poisoning
- `src` ..... adresář obsahující zdrojové kódy práce
  - `thesis.tex` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... adresář obsahující text práce
  - `thesis.pdf` ..... text práce ve formátu PDF