FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

Zadání diplomové práce

Název:	Prostředí pro demonstraci síťových útoků
Student:	Bc. Jan Suchara
Vedoucí:	Ing. Jiří Dostál, Ph.D.
Studijní program:	Informatika
Obor / specializace:	Počítačová bezpečnost
Katedra:	Katedra informační bezpečnosti
Platnost zadání:	do konce letního semestru 2022/2023

Pokyny pro vypracování

Seznamte se s různými útoky typu Denial of Service (DoS), Distributed Denial of Service (DDoS), Spoofing nebo chybami v implementacích protokolů rodiny TCP/IP. Popiště jejich principy a nejčastější způsoby provedení. Uveďte možnosti jejich detekce a obrany proti nim. Na základě předchozí analýzy navrhněte strukturu testovacího prostředí (například několik virtuálních strojů reprezentujících klienty, útočníka a oběť) tak, aby bylo možné vybrané útoky prakticky vyzkoušet v laboratorních podmínkách. Následně testovací prostředí zprovozněte a útoky demonstrujte. Ověřte a vyhodnoťte metody prevence a jejich účinnost.

Elektronicky schválil/a prof. Ing. Róbert Lórencz, CSc. dne 21. února 2022 v Praze.

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Diplomová práce

Prostředí pro demonstraci síťových útoků

Bc. Jan Suchara

Katedra informační bezpečnosti
Vedoucí práce: Ing. Jiří Dostál, Ph.D

5. května 2022

Poděkování

Děkuji svému vedoucímu, panu Ing. Jiřímu Dostálovi, Ph.D., za odborné vedení, cenné rady a připomínky v průběhu psaní této práce. Současně chci poděkovat svojí rodině a svým blízkým, kteří mi byli oporou během celého studia.

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 5. května 2022 .....

Odkaz na tuto práci

Suchara, Jan. *Prostředí pro demonstraci síťových útoků.* Diplomová práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2022.

Abstrakt

Cílem této práce je navrhnout a vytvořit prostředí pro demonstraci vybraných síťových útoků v laboratorních podmínkách. Na základě provedené řešerše nejčastějších útoků jsou některé z nich vybrány k podrobnější analýze. Ta se zaměřuje na jejich dopady, možné způsoby detekce a metody obrany proti nim. Získané výsledky jsou použity k odvození nejvhodnější struktury demonstračního prostředí a výběru nástrojů, kterými lze útoky replikovat. Výstupem práce je prostředí realizované pomocí virtualizačního nástroje VirtualBox a programu Vagrant.

Klíčová slova Síťové útoky, IP spoofing, DoS, DDoS, DNS amplifikace, SYN flood, Snort

Abstract

The aim of this thesis is to design and create an environment for demonstration of selected network attacks in laboratory conditions. Based on the research of the most common attacks, some of them are selected for more detailed analysis. The analysis focuses on impacts of those attacks, possible detection techniques and means of defence. Obtained results are used to derive the most suitable structure of the demonstrational environment and to select tools

for replication of the attacks. The output of this thesis is an environment implemented using the virtualization tools VirtualBox and Vagrant

Keywords Network attacks, IP spoofing, DoS, DDoS, DNS amplification, SYN flood, Snort

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Díky technologickému boomu v 90. letech minulého století došlo k masivnímu rozšíření výpočetní techniky mezi běžné uživatele. Ruku v ruce s ní se do domácností a firem dostal i internet, který se během posledních 20 let stal nedílnou součástí moderní společnosti. Z několika málo tisíc připojených zařízení se za tu dobu rozrostl na globální síť s přibližně 4,5 miliardami aktivních uživatelů a několika desítkami miliard připojených zařízení [1]. S pokračujícím pokrokem a nástupem technologií jako IoT je jisté, že tato čísla nadále porostou. Ve světle nedávných událostí, kdy byla část společnosti nucena pracovat z domova, se role internetu ukázala být ještě důležitější.

Vzhledem k těmto faktům není překvapivé, že se uživatelé internetu i připojená zařízení stávají čím dál častěji terčem útoků jednotlivců nebo organizovaných skupin. Jejich motivace je různorodá - snaha o cenzuru, vydírání, konkurenční boj, či může jít o prostou zvědavost a snahu zviditelnit se. K dosažení svých cílů útočníci využívají rozličné metody. Pomocí nástrojů sociálního inženýrství, jako je phishing nebo vishing, se zaměřují na jednotlivé uživatele. Snaží se je zmanipulovat k instalaci škodlivých programů nebo vyzrazení citlivých informací. Dále jsou zde pokročilejší a technicky mnohonásobně složitější typy útoků. Ty cílí na samotná zařízení a využívají chyb v konfiguraci, v komunikačních protokolech nebo aplikacích.

Nezanedbatelná část síťových protokolů vznikala během překotného vývoje internetu v 90. letech a jsou stále jeho fundamentální součástí. Za normálních okolností jich většina plní svoji roli správně a spolehlivě. Někdy však při jejich návrhu nebo samotné implementaci nebyl kladen dost velký důraz na bezpečnost nebo autoři podcenili vynalézavost útočníků. Ti byli schopni velice rychle odhalit a využít bezpečnostní mezery ke škodlivé činnosti. Problémem je, že často uplyne velmi dlouhá doba mezi odhalením chyby a okamžikem, kdy se ji podaří opravit alespoň na podstatném procentu zařízení. To je zapříčiněno především velikostí, do které se internet rozrostl, a počtem jednotlivých typů zařízení a jejich provozovatelů. Tento fakt zároveň zapříčiňuje pomalou adap-

taci nových, bezpečnějších protokolů, nahrává hackerům a má za následek to, že některé zranitelnosti působí škody i mnoho let po jejich odhalení.

Některé chyby, které se podařilo odhalit a zjednat jejich nápravu, se ovšem mohou opět vracet. Tento jev je často spojen s nástupem internetu věcí, kdy se vývojáři pouštějí do opětovného implementování ověřených protokolů a aplikací, které na klasických zařízení prošly mnoha lety vývoje. Snaha o co nejjednodušší návrh a nepoučení se z předchozích zkušeností, často vede k návratu útoků, které by jinak byli považovány za vyřešené. Několik takových případů bude dále rozebráno v následujících kapitolách.

Další kategorií hrozeb jsou *denial-of-service* (DoS) útoky a především jejich distribuované varianty *distributed denial-of-service* (DDoS). Ty se v poslední době objevují čím dál častěji. Současně s navýšením frekvence jejich výskytu byl v nedávných letech zaznamenán i růst rozsahu těchto útoků. Obě metriky dosahují meziročního nárůstu, který se pohybuje v řádech stovek procent [2]. Největší poskytovatelé cloudových služeb ve svých zprávách informují dokonce o exponenciálním růstu velikosti DDoS útoků proti jejich infrastrukturám. Objemy dat generované těmito útoky během několika let narostly z desítek, maximálně stovek gigabitů, do řádu několika terabitů za sekundu [3] [4]. Tyto útoky jsou často spojovány s rozšířením tzv. botnetů – sítí zařízení kompromitovaných speciálním druhem škodlivého softwaru. Některým z těchto útoků lze relativně snadno zabránit. Jiné je však složité identifikovat a astavit, aniž by došlo k omezení služeb pro běžné uživatele, protože útočící zařízení nejde na první pohled rozlišit od těch legitimních.

Kapitola 1

Cíle práce

Většina útoků, se kterými se v reálném světě setkáváme nejčastěji, je velmi dobře popsána. Bohužel ale není mnoho zdrojů, které by se věnovali konkrétnímu postupu jak dané útoky replikovat v laboratorních podmínkách. Čtenáři jsou nuceni dohledávat kusé informace z mnoha zdrojů a často narážejí na problémy s nastavením systémů, zastaralými nástroji a jejich vzájemnou nekompatibilitou.

Tato práce si klade za cíl navrhnout a vytvořit testovací prostředí, které jeho uživatelům umožní snadno demonstrovat a zanalyzovat vybrané síťové útoky. Prostředí by mělo být jednoduše spustitelné a přenositelné mezi různými platformami. To vše bez nutnosti dodatečné konfigurace. Současně by mělo poskytovat všechny potřebné nástroje pro vykonání útoků, zhodnocení jejich dopadů a otestování možných způsobů detekce i obrany.

K dosažení vytyčených cílů je třeba nejprve splnit následující dílčí úkoly:

1. Seznámit se s vybranými typy síťových útoků, s jejich principy a protokoly, které využívají.
2. Některé z popsaných útoků podrobit detailnější analýze a zhodnotit jejich dopady. Následně otestovat vybrané metody obrany a detekce těchto útoků.
3. Na základě provedené rešerše navrhnout vhodnou strukturu demonstračního prostředí a odvodit nastavení aplikací nebo nástrojů, které jsou nutné k provedení útoků.
4. Získané poznatky využít k implementaci samotného prostředí, popřípadě dalších podpůrných nástrojů.

Výstupy práce umožní čtenářovi redukovat čas potřebný k nastavení všeho potřebného a provedení vybraných útoků. To v konečném důsledku povede k zefektivnění procesu analýzy jednotlivých útoků. Demonstrační prostředí poslouží například studentům předmětů zaměřujících se na síťovou bezpečnost.

Kapitola 2

Vybrané síťové útoky

V této kapitole jsou představeny některé vybrané síťové útoky. Úvodní část se věnuje nejznámějším protokolům rodiny TCP/IP. Ty jsou nedílnou součástí moderních sítí a znalost jejich základních principů je nezbytná k porozumění fungování útoků popisovaných v dalších částech práce.

2.1 Rodina protokolů TCP/IP

Během rozšiřování počítačových sítí na přelomu 70. a 80. let minulého století začalo vznikat mnoho různých komunikačních protokolů, které byly často velmi odlišné. Objevují se proto snahy o standardizaci základních principů síťové komunikace. Výsledkem bylo vytvoření referenčního modelu ISO/OSI, který je abstraktním popisem toho, jak by měla komunikace v telekomunikačních a počítačových sítích probíhat. Tento model se však ve své originální podobě neujal a byl vytlačen rodinou protokolů TCP/IP, který z původního ISO/OSI vychází. V současnosti jde o nejrozšířenější skupinu protokolů v síti internet.

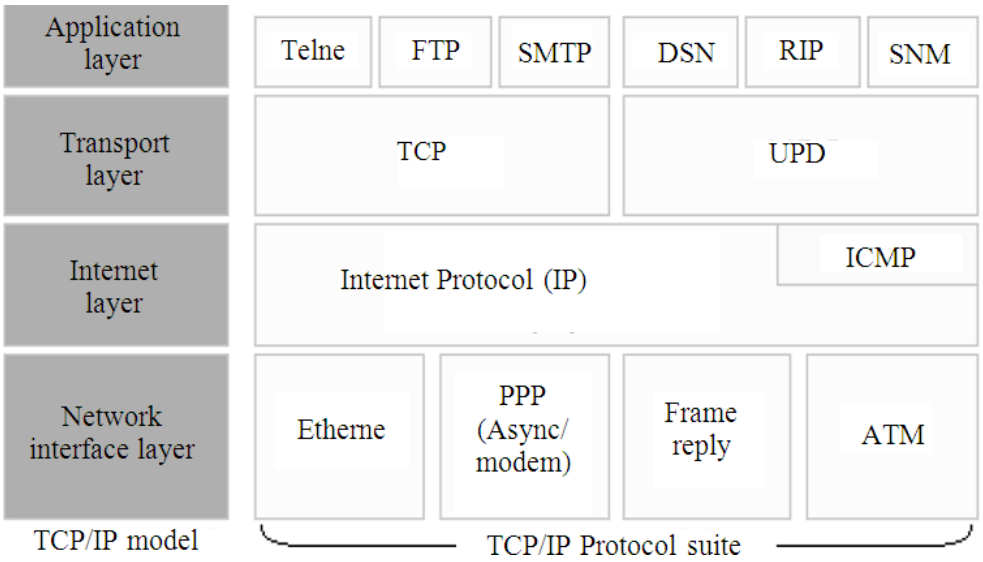
Tok a zpracování dat je v TCP/IP modelu rozdělen mezi 4 hierarchicky uspořádané, funkčně disjunktní vrstvy:

1. linková
2. síťová
3. transportní
4. aplikační

Každá vrstva využívá služeb nižší vrstvy a zároveň poskytuje služby té nad sebou. Při předávání dat mezi jednotlivými vrstvami dochází k jejich zapouzdření neboli enkapsulaci. Odesílaná data jsou na každé vrstvě opatřena hlavičkou, která obsahuje informace specifické pouze pro danou vrstvu. Teprve

poté dojde k jejich předání vrstvám nižším a nakonec k odeslání. Po přijetí dat druhou stranou se proces otáčí. Data prostupují hierarchií směrem vzhůru, přičemž každá vrstva zkontroluje platnost informací v hlavičce. Pokud je vše v pořádku, hlavička je odstraněna a data se předávají vyšší vrstvě.

Každá ze zmiňovaných vrstev je v praxi reprezentována pomocí konkrétního protokolu. Samotný název rodiny TCP/IP je odvozen od dvou jejích nejpoužívanější protokolů – Transmission Control Protocol (TCP) a Internet Protocol (IP).



Obrázek 2.1: Přehled jednotlivých vrstev rodiny TCP/IP a jejich nejznámějších protokolů [5]

2.1.1 Protokol IP

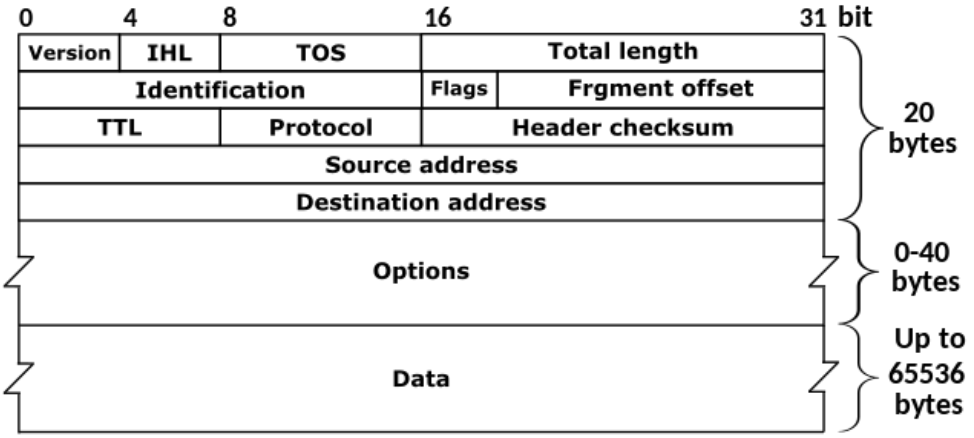
Nejdůležitějším protokolem síťové vrstvy v modelu TCP/IP je Internet Protokol. Internet je tvořen velkým počtem sítí, které jsou vzájemně propojeny. Úkolem tohoto protokolu je zajištění přenosu dat mezi nimi.

Každá veřejně dostupná síť a zařízení v ní jsou identifikovány pomocí unikátních adres. Formát této adresy zároveň určuje použitou verzi IP protokolu. V současnosti existují 2 verze – IPv4 a její nástupce IPv6. V případě IPv4 je adresa 32bitové číslo reprezentované v člověkem čitelné podobě jako 4 dekadická čísla od 0 do 255 oddělená tečkami, např. `142.251.37.100`. Standard IPv6 používá 128bitové adresování zapisované až osmi skupinanmi hexadecimálních čísel, které jsou odděleny dvojtečkami, např. `2001:4860:4860::8888`.

O přenášených datech se v kontextu tohoto protokolu mluví jako o paketech nebo IP datagramech. Každý paket je opatřen IP hlavičkou, která společně s informací o velikosti paketu vždy obsahuje i zdrojovou a cílovou

adresu. Další obsah je závislý na verzi protokolu. Obrázek 2.2 zachycuje strukturu hlavičky IPv4, která zahrnuje například informace o fragmentaci paketu nebo kontrolní součet. Tyto položky byly s novější verzí IPv6 odstraněny z důvodu zvýšení výkonu.

Pole s adresami jsou nastavována na zařízení, které data odesílá, a zpravidla se po cestě k cíli již nemění (výjimkou může být použití NAT[1]). Pokud se cílové zařízení nachází v jiné síti, musí projít skrze směrovač – zařízení ležící na pomezí dvou sítí. Tento směrovač příchozí paket zachytí, přečte cílovou adresu v IP hlavičce a na základě konzultace se směrovací tabulkou rozhodne, jakým směrem data přeposlat.



Obrázek 2.2: Diagram ukazující strukturu hlavičky protokolu IPv4[6]

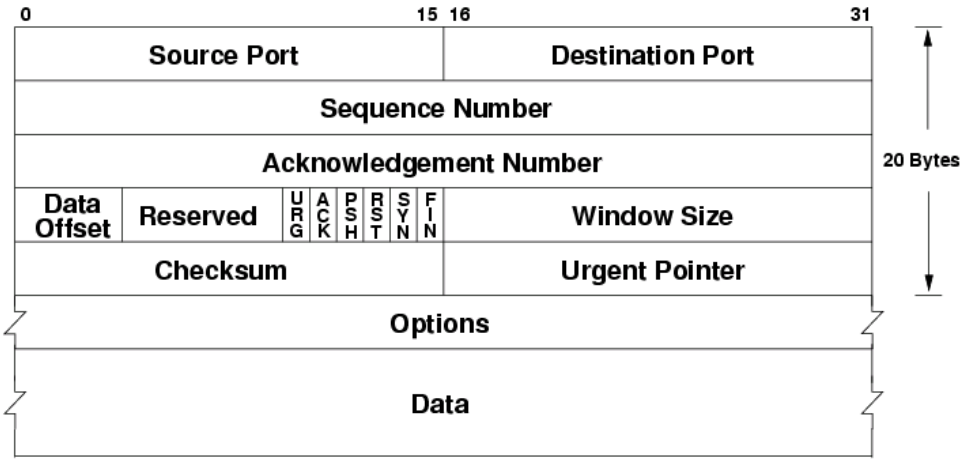
2.1.2 Protokol TCP

Transmission Control Protocol spadá do transportní vrstvy zajišťující přenos dat mezi koncovými uzly. Mezi nejdůležitější vlastnosti TCP protokolu patří:

- je spojově orientovaný
- point-to-point komunikace
- je plně duplexní

První vlastnost značí, že k zahájení výměny dat je nejprve potřeba navázat spojení mezi komunikujícími stranami. Point-to-point vyjadřuje fakt, že jde o spojení pouze dvou zařízení. Poslední vlastnost zajišťuje, že server i klient mohou přes stejné spojení zároveň odesílat i přijímat data. TCP se navíc stará i o přeposílání poškozených nebo ztracených paketů a tzv. congestion control, tedy prevenci zahlcení linky [7].

[1] překlad síťových adres (Network Address Translation)



Obrázek 2.3: Diagram ukazující strukturu hlavičky TCP protokolu [6]

Navázání TCP spojení spočívá v provedení takzvaného handshake, někdy také označovaného jako třícestný (*three-way handshake*). Jak jméno napovídá, skládá se ze tří kroků:

1. Stanice, která chce přes síť navázat spojení s jinou, nejprve odešle inicializační TCP segment. Tento segment má ve své hlavičce nastavený speciální příznak SYN a dále již neobsahuje žádná aplikační data.

2. Naslouchající zařízení na opačné straně segment zpracuje a odešle zpět klientovi segment s příznaky SYN a ACK (označavaný jako SYN+ACK nebo SYNACK).

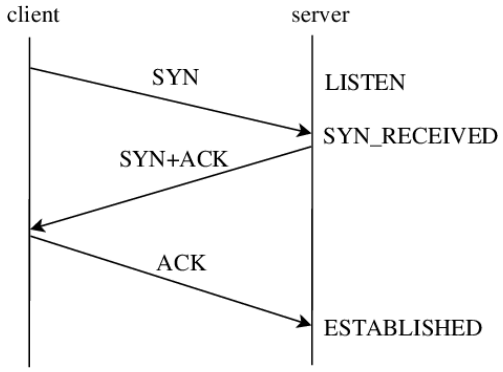
3. Iniciátor spojení po přijetí SYN+ACK segmentu odpovídá dalším s příznakem ACK. Pokud tento segment k serveru dorazí, jde o signál, že navázání spojení proběhlo v pořádku a komunikace mezi oběma uzly může začít.

Kromě výše popsaného scénáře může během handshake každá ze stran reagovat navíc tím, že odpoví segmentem s příznakem RST nebo požadavek ignoruje.

- Segment RST vede k okamžitému ukončení komunikace. Bývá obvykle odesílán v případě, že dojde k pokusu o připojení k portu, na kterém žádná aplikace nenaslouchá. Dalším možným důvodem může být nedostatek volných prostředků na straně serveru nebo ukončení spojení zásahem firewallu.

- V případě, že cíl neodpovídá, bude protistrana reagovat opakovaným odesíláním posledního segmentu až do vypršení časového limitu (time-

out). Tento scénář může být způsoben i tím, že pakety nejsou doručeny kvůli problémům v síti.



Obrázek 2.4: Úspěšný TCP handshake [8]

2.1.3 Protokol UDP

User Datagram Protokol (UDP) se stejně jako TCP nachází na transportní vrstvě TCP/IP modelu. Stejně tak používá pro rozlišení jednotlivých služeb, které běží na počítači, síťové porty. Hlavním rozdílem oproti TCP je, že se jedná o bezspojový (connectionless) protokol. Komunikaci pomocí UDP tedy nepředchází žádné navazovaní spojení a datagramy jsou rovnou odesílány na adresu cílového zařízení. Hlavička přenášených UDP datagramů obsahuje pouze 4 pole:

- zdrojový port
- cílový port
- velikost datagramu
- kontrolní součet

Oproti TCP není zaručeno, že data do své destinace dorazí. Nedochází ani k opakovanému odesílaná datagramů nebo ověřování, zda dorazily ve správném pořadí. UDP se oproti TCP vyznačuje vyšší rychlostí, protože tolik nezatěžuje síťovou infrastrukturu. To je ovšem vykoupeno nižší spolehlivostí a nutností ošetření chyb vzniklých při přenosu v samotných aplikacích, které tento protokol používají.

Protokol UDP nalézá uplatnění především v oblastech, které nejsou citlivé na ztráty jednotlivých paketů. Jedná se zejména o telefonii, streamování videa nebo počítačové hry.

2.2 Spoofing

2.2.1 IP spoofing

Podvržení IP adresy (IP spoofing) je relativně často využívaná technika, která umožňuje útočníkům zamaskovat svojí identitu, popřípadě i obejít některá bezpečnostní opatření v počítačových systémech. Jak již bylo řečeno, zdrojová adresa IP paketu je nastavena automaticky odesílajícím zařízením. Avšak samotné operační systémy poskytují nízkoúrovňové rozhraní pro přístup k síťovým kartám pomocí tzv. `raw soketů`. Díky nim je možné přímo upravovat data odesílaná do sítě a to včetně IP hlavičky. V současnosti existuje mnoho knihoven, které práci s těmito rozhraními usnadňují, například Scapy[2], libtins[3] nebo libcap[4].

Útočníkům tedy stačí tyto nástroje použít k vytvoření paketů se změněnou zdrojovou IP adresou. Směrovače v základním nastavení nijak nekontrolují odkud pakety pocházejí. Zajímají se pouze o cílovou adresu. V důsledku toho pakety dorazí ke svému cíli, který je zpracuje a považuje je za legitimní, protože nemá jak ověřit jejich autenticitu. Tímto způsobem je možné obejít například restrikce v systému, které umožňují komunikovat pouze s vybranými IP adresami.

Základní branou proti podvržení IP adresy je filtrování příchozích a odchozích paketů. Tato řešení je nejefektivnější, pokud je aplikováno na směrovačích ležících na tzv. poslední míly. Jde o routery, které se nacházejí nejblíže koncovým uživatelům a jsou většinou provozované poskytovateli internetového připojení. Další možností obrany je nasazení protokolu IPsec. To ovšem vyžaduje složitější konfiguraci a není možné jej nasadit univerzálně ve všech případech.

2.2.2 Únos TCP spojení

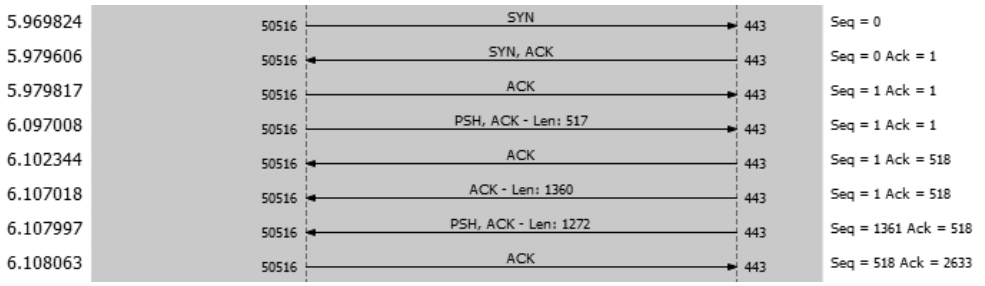
Únos TCP spojení (TCP hijacking) je útok, který umožňuje útočníkovi získat kontrolu nad již navázaným spojením mezi dvěma stranami. Toho docílí tak, že směrem k oběti odesílá segmenty s podvrženou IP adresou, která odpovídá adrese zařízení, se kterým oběť již komunikuje.

Útočník musí zajistit, aby se zdrojová IP adresa a zdrojový a cílový port shodovali s těmi, které využívá napadané spojení. Kromě toho je zapotřebí zvolit i správnou hodnotu 32bitového sekvenčního (*sequence*) a potvrzovacího (*acknowledgement*) čísla, která se taktéž nacházejí v TCP hlavičce.

Tato pole slouží k ověřování odeslaných dat mezi uzly. Sequence number udává pořadí posledního bajtu odeslaných dat. Každá strana eviduje aktuální hodnotu svého sekvenčního čísla a navyšuje jej o počet oktetů, které odeslala. Díky tomu je na straně příjemce možné seřadit data v případě, že dorazila ve

[2] https://scapy.net/
[3] https://libtins.github.io/
[4] https://www.tcpdump.org/

špatném pořadí. Acknowledgement number dává protější straně vědět, jaké pořadové číslo měl poslední bajt, který byl přijat. To umožňuje detekovat ztrátu dat během přenosu a zajistit jejich opětovné odeslání. Obrázek 2.5 zachycuje průběh TCP spojení společně s hodnotami sekvenčního a potvrzovacího čísla, které se mění s počtem odeslaných dat.



Obrázek 2.5: Navázání TCP spojení a následná komunikace zobrazená programem Wireshark

Obě čísla jsou navíc využívána právě k obraně před útoky proti TCP protokolu. Počáteční hodnotu sekvenčního čísla si zvolí klient i server v SYN, resp. SYN+ACK, segmentu během navazování spojení. Tato hodnota se označuje jako Initial Sequence Number (ISN). Pokud se v průběhu následující komunikace objeví segment, v němž se jedna či druhá hodnota výrazně liší od té očekávané, bude celý segment zařízením ignorován. V praxi to znamená, že pro sekvenční číslo seq_{max} v útočníkem podvrženém datagramu musí platit:

$$ack_{last} < seq_{max} \leq ack_{last} + W_{size}$$

kde ack_{last} je nejvyšší hodnota acknowledgement number, kterou oběť zatím odeslala, a W_{size} je aktuální velikost TCP okna pro dané spojení. Útok je tedy efektivní zejména proti dlouhotrvajícím spojením, přes která není odesíláno velké množství dat.

Situace je pro útočníka jednodušší v případě, kdy může odposlouchávat komunikaci mezi jeho oběťmi, například pokud se nachází ve stejném síťovém segmentu nebo komunikace probíhá přes ním kontrolovaný uzel. V tomto případě může přímo pozorovat hodnoty ISN obou komunikujících stran i porty, které využívají.

Pokud ovšem tuto možnost nemá, může se stále pokusit o *off-path* TCP hijacking. V této situaci nemá útočník přístup k segmentům odesílaných mezi zařízeními a musí výše zmiňované hodnoty „uhádnout“. Proto je velice důležité, aby byla hodnota ISN pro nová spojení bez znalosti vnitřního stavu systému nepředvídatelná.

V minulosti bylo objeveno několik zranitelností, které umožňovali takovéto útoky provádět. Hlavní příčinou byly předvídatelné hodnoty sekvenčních čísel generované některými operačními systémy. Pokud se útočníkovi podařilo od-

vodit, přes jaký port komunikace probíhá, stačil mu odeslat pouze relativně malý počet paketů k úspěšnému provedení útoku.

2.3 Útoky (D)DoS

Následující sekce je věnována dělení a jednotlivým variantám útoků anglicky označovaných jako Denial-of-Service – odepření služby. Principiálním cílem těchto útoků je znepřístupnit nebo zcela vyřadit z provozu konkrétní služby, servery nebo celé počítačové sítě.

2.3.1 Klasifikace

Existuje několik definic těchto útoků, které jsou často velmi podobné. Jako příklad můžeme uvést tuto obecnou definici, která byla formulována autory v knize [9]:

Definice 2.3.1 (Denial-of-service): *Útok denial-of-service je činnost, která vede k zabránění legitimního přístupu k prostředkům, zpomalení časově kritických operací, nebo znemožňuje fungování nějaké části informačního systému.*

Zaměříme-li se na útoky prováděné přes počítačové sítě, lze definici formulovat takto:

Definice 2.3.2 (Síťový denial-of-service): *„K síťovému útoku typu denial-of-service dochází, když nějaká skupina síťových entit záměrně používá síťové služby s cílem spotřebovat síťové zdroje nebo je poškodit takovým způsobem, že přístup k jinak dostupným síťovým službám je pro nějakou jinou skupinu síťových entit degradován nebo zpožděn tak, že jsou služby učiněny nepoužitelnými.*[5]*“ [10] (překlad autora)*

V praxi dosáhnou útočníci uvedených cílů vyčerpáním výpočetních prostředků oběti nebo zařízení v síťové infrastruktuře, která k ní zprostředkovává přístup. Z pohledu běžného uživatele jsou ovšem tyto možnosti v konečném důsledku identické, neboť obě vyústí v nedostupnost služby nebo výraznému snížení její schopnosti reagovat na požadavky.

Existuje několik možností jak klasifikovat DoS útoky. Základní dělení rozlišuje útoky podle způsobu provedení a to na:

- lokální – obvykle vyžadující přístup k cílovému stroji
- vzdálené – prováděné přes počítačovou síť

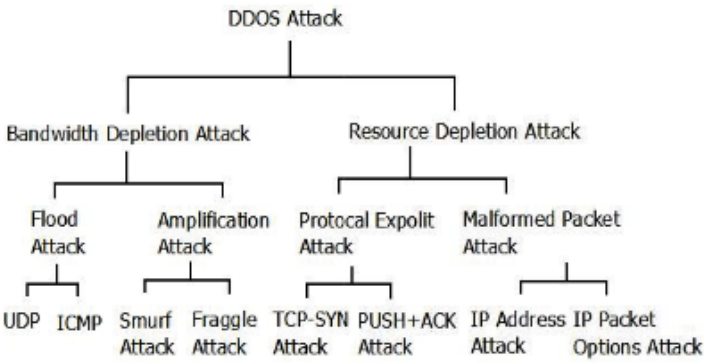
[5] A network denial-of-service attack occurs when some set of network entities intentionally uses network services with the goal and effect of causing consumption or corruption of network resources in such a way that some other set of network entities have their ability to access otherwise usable network services degraded or so delayed as to render them unusable.

Tato práce se bude dále zabývat pouze útoky vzdálenými, protože ty v současnosti nad lokálními DoS útoky převažují. Lokální útoky často vyžadují, aby měl útočník předem získanou nějakou formu přístupu ke svému cíli, například ovládal uživatelský účet a měl oprávnění ke spouštění různých programů. Toho je obvykle mnohem komplikovanější dosáhnout. Zároveň existují účinnější prostředky obrany. Administrátoři mohou například nastavit kvóty pro výpočetní prostředky nebo úplně zamezit přístupu z daného uživatelského účtu.

Další rozdělení navrhované autory článku [11] se zaměřuje na použitou techniku a to, zda-li dojde k odmítnutí služby v důsledku vyčerpání prostředků nebo šířky pásma síťového připojení. Zmiňovaný článek se zabývá DDoS útoky, ale tuto klasifikaci lze použít i pro nedistribuovanou variantu denial-of-service.

1. vyčerpání prostředků
    - zneužití návrhu čí konkrétní implementace protokolu
    - manipulace s packety
2. vyčerpání šíře pásma
    - záplavové útoky
    - reflexivní útoky
    - amplifikační útoky

Je potřeba uvést, že toto dělení není vždy jednoznačné a uvedené kategorie se mohou překrývat. Útočníci navíc v reálném světě často během jednoho útoku využívají kombinaci několika metod.



Obrázek 2.6: Klasifikace DDoS útoků společně s jejich nejznámějšími zástupci [11]

2.3.2 Záplavové útoky

Záplavové útoky neboli tzv. *flood útoky* jsou v současnosti nejméně sofistikovaným, ale zároveň nejvíce rozšířeným typem DoS útoků. Jejich základním principem je odesílání velkého množství požadavků na cílový server, což vede k jeho přetížení nebo zahlcení linky.

Na internetu je dostupné velké množství nástrojů, které umožňují provádět tyto útoky i bez potřeby hlubších technologických znalostí. Vysoký výkon dnešních počítačů a prvků sítové infrastruktury však snižuje pravděpodobnost, že by jediný útočník mohl být ve své snaze úspěšný. Proto je nejčastěji využívána jeho distribuovaná varianta, kdy je cíl koordinovaně napaden tisíci zařízeními, která bývají součástí botnetu.

Obrana proti tomuto druhu útoků nemusí být jednoduchá a v případě použití jeho distribuované varianty je dokonce značně složitá. Největší problém spočívá v rozlišení legitimních požadavků od těch škodlivých.

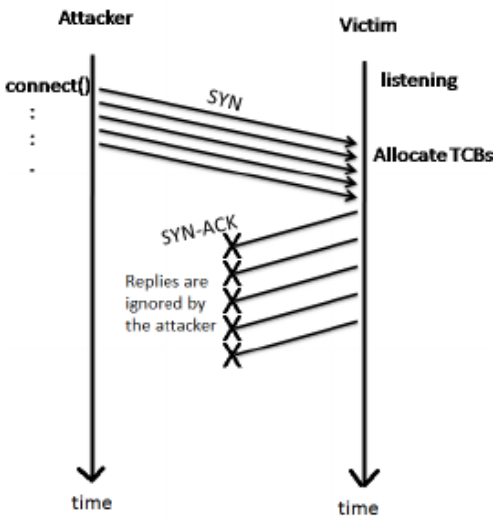
2.3.2.1 SYN flood

Tato metoda byla poprvé popsána již v první polovině 90. let [12], ale o její rozšíření se nejvíce zasloužil článek v magazínu Phrack, který vyšel v roce 1996 [13]. SYN flood útoky využívají toho, jakým způsobem operační systémy nakládají s TCP spojením, které ještě nebylo zcela navázáno.

Server po obdržení SYN segmentu alokuje prostředky potřebné pro budoucí komunikaci. Jedná se o speciální strukturu Transmission Control Block (TCB), jejíž implementace se liší systém od systému. Vždy ale obsahuje informace o vstupních a výstupních bufferech pro data, stav spojení, hodnoty čítačů, čísla portů a další [14]. Současně slouží k udržování aktuálního stavu daného připojení. V této fázi bude stav nastaven na `SYN_RECEIVED` a celé TCB se přesune do tzv. SYN queue – datové struktury sloužící k evidenci spojení čekajících na potvrzení. V momentě, kdy server obdrží finální ACK segment, nastaví stav na `ESTABLISHED` a přemístí TCP ze SYN queue do ACCEPT queue, která je určena pro již plně navázaná připojení.

Pokud server po odeslání SYN+ACK neobdrží od klienta ACK paket, dojde k‘opětovném odesílání SYN+ACK. Konkrétní počet opakování a prodleva mezi nimi se řídí nastavením systému, resp. *exponenciálním back-off* algoritmem. Po dobu čekání na odpověď jsou alokované prostředky stále blokovány. Tím, že budeme generovat velké množství paketů s příznakem SYN a současně ignorovat přicházející odpovědi, dojde k vyčerpání volných kapacit pro spojení s klienty na straně serveru. V konečném důsledku tím budou ostatním uživatelům znepřístupněny služby, které na něm běží.

Útok se v této základní podobě nazývá přímý (direct attack) a existuje proti němu relativně jednoduchá obrana. Pokud útočník využívá pouze jednu stanici a nijak nemaskuje svojí IP adresu, stačí na straně oběti odfiltrovat



Obrázek 2.7: Záplavový útok pomocí SYN segmentů z jednoho zdroje bez použití podvržené adresy [15]

provoz z tohoto konkrétního uzlu. Navíc server udržuje pouze jedno spojení a možnost vyčerpání prostředku je tedy velmi nízká [15].

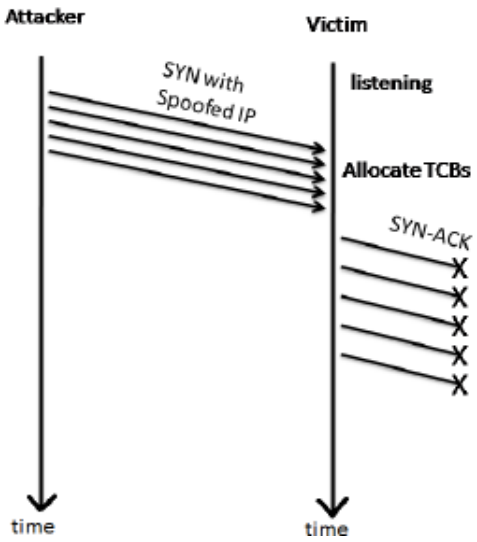
Proto se v praxi tento přístup kombinuje s IP spoofingem. Kombinace obou těchto metod velmi znesnadňuje obranu, protože nelze jednoduše určit, které adresy jsou legitimní a které ne.

Útočník obvykle generuje SYN segmenty s podvrženými zdrojovými adresami. To má za následek, že server pro každý takový segment alokuje nové spojení a na tuto adresu odesílá SYN+ACK. Pokud je host na podvržené adrese dostupný, odpoví segmentem RST, protože spojení neočekává. V ten moment server uvolní alokované prostředky a spojení ukončí. Ovšem v případě, že na dané adrese žádné zařízení neodpovídá, server pokus o spojení opakuje a dochází k situaci popsané výše.

Efektivní obranou proti tomuto útoku je tzv. SYN cookie. Jejím učelem je zabránění vyčerpání prostředků pro přicházející spojení tím, že se nealokuje TCB pro připojení, která ještě nebyla zcela navázána (jsou ve stavu SYN_RCVD). Namísto toho jsou základní informace o tomto spojení zakódována do 32bitového čísla (SYN cookie), které se pak použije jako ISN v segmentu SYN+ACK. Toto číslo se vypočítá následovně:

$$ISN = M + F(IP_{local}, PORT_{local}, IP_{remote}, PORT_{remote}, secret\_key)$$

kde M je monotónní rostoucí funkce, F je hashovací funkce a IP_{local}, $PORT_{local}$, IP_{remote}, $PORT_{remote}$ jsou postupně lokální IP, lokální port, vzdálená adresa, vzdálený port a $secret\_key$ je tajný klíč. M obvykle vrací hodnotu čítače, který se inkrementuje v pravidelných intervalech několika mikrosekund. F by měla být kryptograficky bezpečná hashovací funkce a hodnota $secret\_key$ by



Obrázek 2.8: Útočník odesílá SYN segmenty s podvrženými adresami. Cíl útoku na tyto adresy adpovídá SYN+ACK segmenty [15]

měla být dostatečně velkým náhodným číslem. Tento klíč se mění při restartu systému, po uplynutí určité doby, nebo po předem definovaném počtu použití.

Jakmile server obdrží ACK, opět přepočítá SYN cookie a ověří, že je shodná s hodnotou acknowledgement number v přijatém segmentu. Dále se zkontroluje, o kolik se zvětšila hodnota funkce M. Pokud se hodnoty liší nebo stáří cookie překročilo maximální povolenou dobu, je spojení ignorováno.

Nevýhodou tohoto řešení je jeho nekompatibilita s některými funkcemi protokolu TCP a jeho rozšířeními. Počáteční SYN paket může obsahovat různé informace v poli *Options*, které slouží například ke zlepšení kvality služeb. Pokud jsou používány SYN cookies, dojde ke ztrátě těchto informací, protože je server neuloží ani nezakóduje v odeslaném segmentu. Dalším možným opatřeními je omezení počtu opakovaného odesílání SYN+ACK segmentů nebo zvýšení kapacity SYN queue.

2.3.2.2 UDP flood

Jak název napovídá, jedná se o útok využívající protokol UDP. Během UDP zaplavení je odesláno velké množství UDP paketů buď na náhodné nebo specificky určené porty na systému oběti. Jako příklad služeb, které se v minulosti často stávali terčem UPD flood útoku, můžeme uvést ECHO a CHARGEN protokoly běžící na portech 7 a 19. Pokud klient pošle data na serverový ECHO port, budou mu poslána bez jakýchkoliv úprav zpět. Na portu č. 19 je protokol CHARGEN. Tento protokol přijme doručená data, ale odesílateli jsou vrácena data náhodná.

```
GET /index.html HTTP/1.1
Host: localhost
User-Agent: curl/7.71.1
Accept: text/html
Accept-Language: en-GB
Accept-Encoding: gzip, deflate
```

Ukázka 2.1: HTTP požadavek

2.3.2.3 HTTP flood

Hypertext Transfer Protocol (HTTP) vznikl na přelomu 80. a 90. let minulého století. Slouží primárně k přenášení hypertextových dokumentů, které jsou nejčastěji ve formátu HTML a obsahují text, obrázky nebo odkazy na další dokumenty. Základem protokolu je klient-server architektura. To znamená, že dokumenty jsou uloženy na serveru a klient na něj zasílá HTTP požadavky. Server požadavky zpracuje a vrátí HTTP odpověď.

Protokol je navržen jako bezstavový. Server tedy není povinen ukládat informace o stavu minulých požadavků. Každý nově příchozí požadavek je zpracováván, jako by byl první. HTTP operuje na aplikační vrstvě modelu TCP/IP a ke svému fungování vyžaduje spolehlivý transportní protokol – nejčastěji TCP. Ve výchozím nastavení server naslouchá na portu 80 [16].

V protokolu existuje několik metod, kterými klienti říkají serveru, jakou akci chtějí provést. Dvěma nejběžnějšími jsou:

- GET – klient od serveru požaduje vrácení dokumentu na konkrétní adrese
- POST – klient na server odesílá data a požaduje jejich uložení nebo zpracování

Server ve své odpovědi zahrnuje i *stavové kódy*, které klienta informují o výsledku jeho požadavku a případných chybách. Klient musí na tyto kódy patřičně reagovat.

HTTP požadavek i odpovědi ve svých hlavičkách obsahují i tzv. *header fields*, které obsahují dodatečné informace například o typu klienta nebo serveru, očekávaném formátu dat, souborech cookies a podobně. Ukázka 2.1 zachycuje strukturu HTTP požadavku, kde na prvním řádku vidíme zvolenou metodu, cestu k dokumentu (URI), verzi HTTP protokolu a pole *User-Agent*, které obsahuje identifikaci použitého HTTP klienta. Následující položky uvádějí podporovaný jazyk a kódování obsahu. Na ukázce 2.2 je vidět navrácená hlavička HTTP odpovědi. Na prvním řádku vidíme opět verzi protokolu následovanou stavovým kódem 200 a jeho textovou interpretací.

Princip tohoto útoků je stejný jako v předchozích případech – zahltit server odesíláním velkého množství HTTP požadavků. Požadavky se na první

```
HTTP/1.0 200 OK
Server: nginx
Date: Tue, 01 Mar 2022 14:36:14 GMT
Content-type: text/html
Content-Length: 162
```

Ukázka 2.2: HTTP odpověď

pohled jeví jako legitimní a nelze je snadno rozeznat od ostatních. Útočníka však prozradí kvantita dotazů, které generuje. Jde o velice primitivní techniku, která nevyužívá žádnou zranitelnost nebo nedokonalost protokolu, jako tomu bylo v předchozích případech. To činí tyto útoky oblíbené i mezi lidmi bez technického vzdělání nebo tzv. *script kiddies*, a to hlavně díky volně dostupným nástrojům, které je umožňují spustit pomocí několika kliknutí. Nejznámějšími z těchto programů jsou Low Orbit Ion Cannon, aktualizovaná verze High Orbit Ion Cannon nebo HTTP Unbearable Load King (HULK). Jsou často využívány i „hacktivistickými" skupinami, jako je Anonymous, k znepřístupnění webových stránek organizací nebo státních orgánů [17].

Obrana proti těmto útokům spoléhá hlavně na správné nastavení pravidel firewallu, kterými lze omezit počet přijímaných požadavků od jednoho klienta za určitý časový úsek. Další řešení vychází z pozorování, že uživatel v daný moment využívá k prohlížení webu pouze jeden prohlížeč. Výše uvedené nástroje nastavují v hlavičce HTTP požadavku pole *User-Agent*, zmiňované v ukázce 2.1. Obvykle je jeho hodnota náhodně vybrána z předdefinovaného seznamu, který vytvořili autoři programu. Lze tedy sledovat, zda-li z dané adresy nepřicházejí požadavky s často se měnící identifikací klienta. Zde ovšem může nastat problém v případě, kdy je za jednou adresou několik klientů, jako například při použití NAT nebo VPN.

2.3.2.4 ICMP flood

Tato metoda je založena na protokolu ICMP (Internet Control Message Protocol), který slouží jako podpůrný protokol pro IP. Využívají ho síťová zařízení k diagnostikování různých problémů nebo předávání informací o stavu sítě. Přenášené datagramy obsahují ve své hlavičce 3 bajty, které kódují typ zprávy, stavový kód a kontrolní součet pro ověření integrity. Následují pole, jejichž obsah závisí na typu zprávy [18].

Jednou ze základních funkcí ICMP je možnost provádět tzv. *ping*. Za normálních okolností slouží tento nástroj k ověření dostupnosti cílového systému. Ze zdroje je vyslána zpráva Echo Request s velikostí obvykle nepřesahující několik desítek bajtů. Od cílového zařízení je pak očekávána odpověď Echo Replay, která má stejný obsah. Tento proces je například využíván programy *ping* a *tracert* ke zjištění latence připojení, dostupnosti zařízení a počtu směro-

vačů mezi nimi. Tento útok se také nazývá *ping attack*, protože využívá právě zmiňované funkcionality ICMP protokolu.

Útočník při něm odesílá velké množství Echo Request zpráv o maximální velikosti dovolené IPv4 nebo IPv6 pakety. Tím může dosáhnout spotřebování značného množství výpočetní kapacity oběti a také mezilehlé síťové infrastruktury. Tento útok je možné využít jak proti konkrétním počítačům, tak proti směrovačům a jiným sítovým prostředkům, protože podpora ICMP je povinná pro každou implementaci IP protokolu [19].

Tento útok ale vzhledem k výkonu současné infrastruktury a počítačů nepředstavuje vážnou hrozbu. Výjimkou by mohla být situace, kdy špatně nakonfigurovaný směrovač nebo přepínač odpovídá a přeposílá Echo Request pakety směrující na všesměrové adresy v síti.

Obranou proti ICMP flood je dočasná deaktivace ICMP na napadeném zařízeni nebo filtrování příchozích ICMP paketů z vnější sítě. Toto řešení ale bude mít dopad na možnosti diagnostiky sítě.

2.3.3 Reflexivní a amplifikační útoky

Během reflexivních útoků jsou proti obětem využívány tzv. reflektory. Reflektor je počítač, který na příchozí požadavek reaguje odesláním odpovědi zpět na zdrojovou IP adresu. Typickým příkladem reflektoru je například DNS, emailový server či jiná síťová služba. Principem útoku je vyslání požadavku s podvrženou zdrojovou IP adresou k těmto reflektorům. Podvržená adresa je v tomto případě shodná s adresou oběti. Na stroj oběti poté přicházejí nevyžádané odpovědi , které musí vyřizovat. Současně dochází i k zahlcení linky.

Amplifikační útoky využívají principu zesílení. Směrem ke zranitelné nebo chybně nakonfigurované službě se odešle spouštěcí paket (*trigger packet*), který obsahuje jen minimum informací a je tedy velmi malý. Zranitelná služba na něj poté reaguje odesláním odpovědi, jejíž velikost často několikanásobně převyšuje velikost dotazu a tím dochází k amplifikaci. Míru amplifikace jednotlivých útoků můžeme vyjádřit metrikou nazývanou jako *amplifikační faktor*. Pro výpočet amplifikačního faktoru A_f platí vztah:

$$A_f = \frac{size_{resp}}{size_{req}}$$

kde $size_{resp}$ je velikost odezvy a $size_{req}$ je velikost odpovědi.

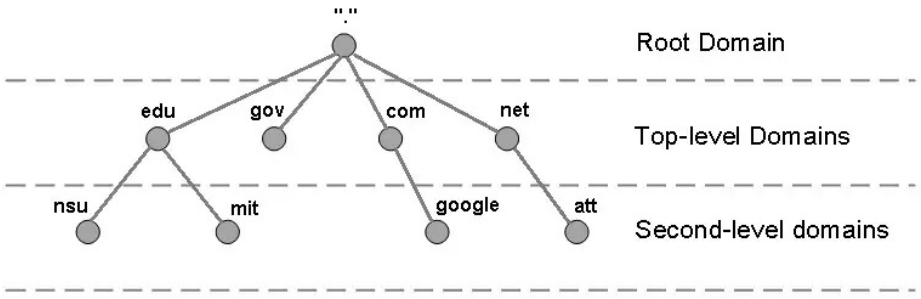
Pomocí snadno dostupných nástrojů může útočník odeslat mnoho tisíc těchto spouštěcích paketů. Při vysokém faktoru zesílení mohou amplifikační útoky dosahovat velikosti až stovek gigabitů za vteřinu a zahltit tak infrastrukturu i velkých poskytovatelů internetových služeb.

V praxi se tyto útoky často kombinují, což lze pozorovat například u DNS flood, který je popsaný níže. V tomto případě jde o využití hned 3 metod – amplifikace, reflexivního útoku a podvržení IP adresy. Co dělá tuto kombinaci

útoků tak nebezpečnou, je fakt, že jej mohou provádět jak běžné počítače, tak i méně výkonné IoT zařízení. Na první pohled nemusí ani vykazovat náznaky kompromitace, protože k útoku jsou použity standardní protokoly Internetu.

2.3.3.1 DNS flood

S rozmachem Internetu a především WWW (World Wide Web) vyvstala potřeba zavést systém, který by uživatelům zjednodušil používání těchto adres. Za tímto účelem vznikl Domain Name System (DNS). Jde o službu umožňující překlad názvů domén na konkrétní IP adresy a naopak. Jedná se o distribuovaný, hierarchický systém, který ke svému fungování využívá stejnojmenný protokol.



Obrázek 2.9: Hierarchie DNS domén [20]

Hierarchie je zajištěna uspořádáním do stromové struktury, v jejímž kořeni leží tzv. kořenová doména. V další hladině se nacházejí domény nejvyšší úrovně (TLD), pod nimi domény druhé úrovně a níže následují autoritativní jmenné servery. Kořenová doména je spravována neziskovou organizací ICANN a aktuálně ji obsluhuje 13 serverů. Tyto servery udržují informace o státních TLD (ccTLD), jako .cz nebo .us, a generických TLD (gTLD), například .org nebo .edu. TLD servery pak mají na starost odpovídající státy nebo organizace. Následující autoritativní servery jsou zodpovědné za subdomény konkrétních domén [21].

O převod doménového jména na IP adresu se nejčastěji starají *rekurzivní resolvery*. Jde o lokální DNS servery, které jsou provozovány v lokálních sítích organizací nebo poskytovatelů internetového připojení (ISP). Slouží k obsloužení požadavků klientů a také jako vyrovnávací paměť. Na obrázku 2.10 je znázorněn překlad probíhající v několika krocích:

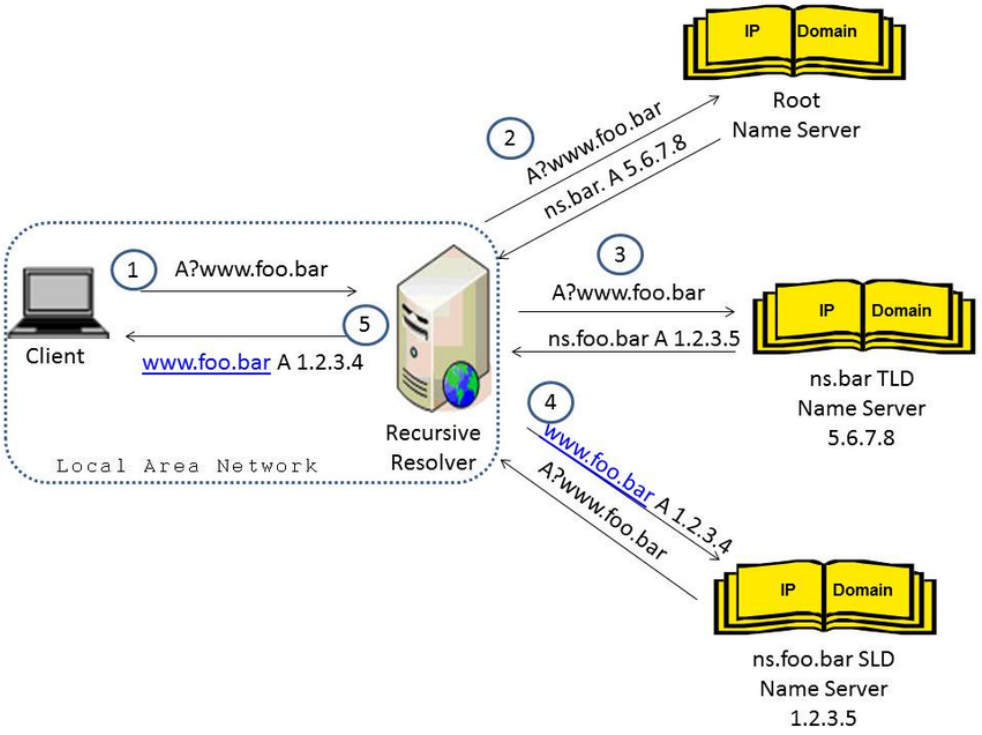
1. Uživatel se snaží přistoupit k serveru s doménovým jménem *www.foo.bar*. Předpokládejme, že odpovídající adresa není uložena ve vyrovnávací paměti počítače a proto je nucen dotázat se rekurzivního resolveru. Odešle na něj tedy DNS dotaz, tzv. DNS query.

2. Resolver nejprve zkontroluje svojí vyrovnávací paměť. Pokud překlad není přítomen, kontaktuje kořenový server a požádá o informace o TLD serveru, který spravuje zónu *.bar*.

3. Po obdržení odpovědi se resolver obrátí na daný server a požádá o adresu autoritativního serveru pro doménu *foo.bar*.

4. Následně se tohoto serveru dotáže na adresu náležící *www.foo.bar*. Autoritativní server vrátí DNS záznam typu A, resp. AAAA. Záznam obsahuje IPv4, resp. IPv6, hledaného webového serveru společně s informací o tom, jak dlouho může resolver uložit překlad do vyrovnávací paměti předtím, než bude muset dotaz opakovat.

5. V posledním kroku DNS resolver odešle hledanou IP adresa počítači, který o ni zažádal.



Obrázek 2.10: Proces překladu doménového jména na IP adresu [22]

DNS protokol, používaný při výše popsaném procesu, nejčastěji využívá port 53 a protokol UDP. Jak již bylo popsáno výše, UDP nevyžaduje žádné předchozí navázání spojení. Právě této vlastnosti útočníci zneužívají. Útočníci nejprve cíleně vybírají domény, jejichž jmenné servery vracejí velmi velké odpovědi na různé typy DNS dotazů. V praxi se využívají dotazy typu `ALL` nebo

`TXT`. První jmenovaný slouží k získání veškerých údajů spojených s danou doménou, které má server k dispozici. Dotaz typu `TXT` pak vrací tzv. `TXT` záznamy. Ty slouží administrátorům pro uložení doplňujících informací nebo k verifikaci vlastníka domény. Velikost těchto dat může být i několik kilobajtů a umožňují tím útočníkům dosáhnout velkého amplifikačního faktoru.

Po vybrání vhodné domény stačí, pokud naleznou DNS resolver nakonfigurovaný tak, že odpovídá na dotaz jakýmkoli klientům v Internetu. Následně na něj zašlou DNS dotaz, který vyústí v odpověď s největší velikostí, a zamění jeho zdrojovou adresu za adresu cíle útoku. Resolver poté odešle výsledek na IP adresu oběti [23].

Zmíněná metoda má však tu nevýhodu, že útočníci musejí najít doménu a odpovídající autoritativní servery, které mají vhodné vlastnosti. To může být pracné a zdlouhavé. Tento problém mohou vyřešit tak, že si zaregistrují vlastní doménu a zprovozní vlastní autoritativní server, který jí bude spravovat. Výsledkem je dosažení úplné kontroly nad vracenými záznamy. Dalším krokem bývá nastavení vysoké hodnoty `TTL` pro danou doménu, který určuje, jak dlouho mohou být získané informace drženy ve vyrovnávací paměti DNS resolverů. V důsledku docílí toho, že resolver použitý k útoku pravděpodobně kontaktuje jejich server pouze jednou a poté bude na podvržené dotazy vracet výsledky uložené ve své paměti. Útočníci tedy musejí udržovat autoritativní server dlouho a sníží tak svoje případné náklady a především pravděpodobnost jejich dopadení, která je ovšem stále vyšší než v případě využití již existující domény. Amplifikační faktor je při této metodě relativně vysoký a tím činí útok velmi efektivní [24].

2.3.4 DDoS útoky

DDoS útoky se prakticky neliší od DoS útoků, kterým byla věnována předchozí kapitola. DDoS využívají stejné principy a můžeme je rozdělit i do stejných kategorií. Jediný praktický rozdíl mezi nimi je ten, že u DoS útoků je útočící zařízení pouze jedno. Naproti tomu do DDoS útoků se zapojuje zařízení více. Množství zúčastněných počítačů může být i dosti velké. V některých případech se jedná o koordinované útoky tisíců až stovek tisíc počítačů.

Tento fakt s sebou nese několik výhod pro útočníka a zároveň i komplikací pro oběť. Vzhledem k výkonu dnešních počítačů je pravděpodobnost úspěchu DoS útoku relativně malá a k jeho zastavení postačuje zablokovat pouze jeden stroj či adresu. Pokud ale útočí velké množství počítačů najednou, šance na úspěch vzroste. To je způsobeno velkým nárůstem objemu odesílaných dat a současně nutností identifikovat mnohem větší počet zdrojů útoku, což nemusí být triviální. Často je totiž obtížné rozlišit legitimní provoz od toho škodlivého. Následkem filtrování nesprávných požadavků pak může dojít i k znepřístupnění služby pro běžné uživatele a tedy neúmyslnému naplnění cílů útočníků.

Motivaci pro tyto útoky můžeme rozdělit do několika kategorií [25]:

- **Finanční nebo ekonomická** – Příkladem může být konkurenční boj internetových obchodů, kdy se jeden subjekt pokusí vyřadit webové stránky ostatních a tím získat část jejich klientely. Strůjcem útoku může být i jednotlivec nebo skupina, která provozovatele takového obchodu vydírá a výměnou za ukončení útoku požaduje výpalné.

- **Pomsta** – Cílem takového útoku je odplata za nějakou reálnou či domnělou újmu způsobenou firmou, státní institucí nebo i jednotlivcem. Typickými obětmi jsou školy, bankovní domy nebo novináři.

- **Ideologické přesvědčení** – Jde o takzvaný „hacktivismus“, kdy útočníci dávají najevo nesouhlas s názory oběti. Často se jedná o politicky nebo nábožensky motivované útoky směřující proti státním podnikům nebo orgánům veřejné moci.

- **Demonstrace schopností** – Snahou útočníků je prokázat své dovednosti, zviditelnit se nebo získat uznání.

- **Kybernetická válka** – V případě kybernetické války jsou útoky vedeny přímo státními složkami nebo státem sponzorovanými skupinami. Cílem je poškodit kritickou infrastrukturu protivníka a získat nad ním strategickou výhodou

DDoS útoky mohou vznikat z iniciativy běžných uživatelů tak, jako tomu bylo například v reakci na konflikt na Ukrajině. Během několika dnů vznikly webové stránky, které jejich návštěvníkům umožňovali zapojit se do útoku tím, že v prohlížeči spustili skript odesílající požadavky na servery oběti. Přestože jde o činnost, která je nelegální, zapojilo se do ní během prvních dnů několik desítek tisíc lidí [26].

Faktem ale zůstává, že v drtivé většině případů jsou DDoS útoky prováděny prostřednictvím sítí počítačů infikovaných malwarem. Tyto sítě ovládané organizovanými skupinami hackerů se nazývají *botnety*. Je proto vhodné uvést definici botnetu a stručný popis jeho fungování.

2.3.4.1 Botnet

Botnet je síť zařízení infikovaných malwarem, které se nazývají *boti* nebo také zombie. Takto infikovaná zařízení jsou vzdáleně ovládána *botmasterem* – operátorem botnetu. Hlavní charakteristikou těchto sítí je schopnost komunikace s tzv. *command-and-control* (C&C) servery, které jsou pod kontrolou právě botmastera. Z těchto serverů jsou jim rozesílány povely, které mají vykonávat. V praxi existuje mnoho komunikačních kanálů, pomocí kterých boti s C&C komunikují. Nejčastěji se můžeme setkat se řízením botů pomocí protokolu IRC, HTTP nebo SMB [27]. Existují ovšem i komplexnější způsoby řízení botnetů, které budou představeny později.

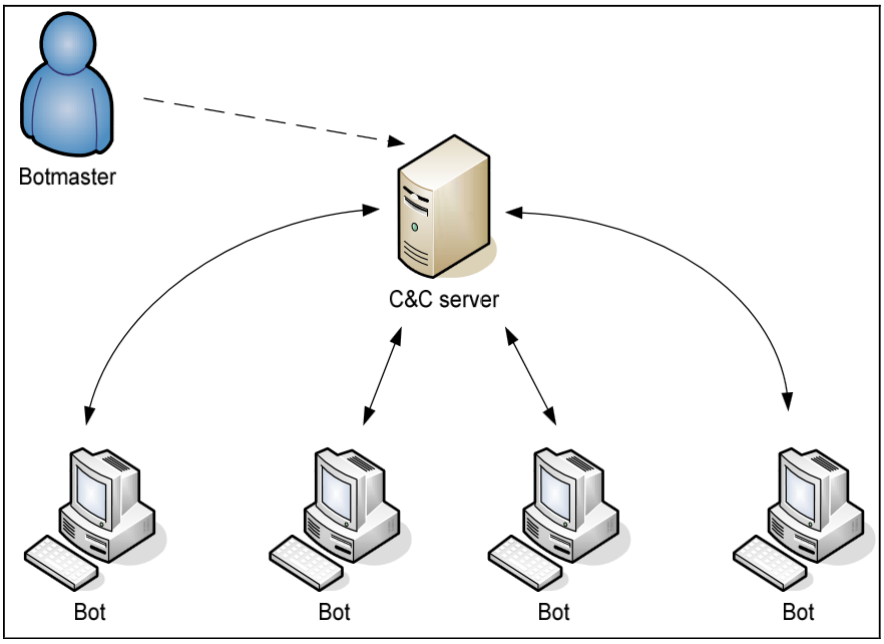
Takto vytvořené sítě mohou zahrnovat až stovky tisíc botů, díky čemuž dosahují obrovského kombinovaného výpočetního výkonu a šířky pásma. Participující zařízení jsou často různých typů a nacházejí se v několika zemích, což v důsledku znesnadňuje jejich detekci. Motivací pro vytváření těchto sítí je několik:

- rozesílání spamu
- distribuce dalšího malwaru
- těžba kryptoměn
- sběr uživatelských dat (hesla, klíče kryptopeněženek atd.)
- vykonávání DDoS útoků

Command-and-control servery jsou zároveň slabým článkem botnetů. Pokud jsou C&C vyřazeny z provozu, přestává de facto existovat i samotná síť botů, protože je nelze dále ovládat. V anglické terminologii se pro tento případ používá výraz single point of failure – jediné místo selhání. Důsledkem toho je, že se v poslední době čím dál častěji setkáváme s botnety s hybridní nebo decentralizovanou topologií. Ty se částečně nebo úplně zbavují závislosti na C&C serverech díky využívání *peer-to-peer* (P2P) komunikačních protokolů. Jednotlivé topologie byly detailně rozebrány v článku [28], ze kterého čerpá následující stručný popis každé z nich.

Centralizované botnety

Nejjednodušší architekturou pro ovládání botnetů je architektura centralizovaná, kdy boti s C&C servery komunikují pomocí metody klient-server. Roli serveru zde nejčastěji zastává IRC kanál nebo webový server. Infikovaní klienti se k danému serveru pravidelně připojují a kontrolují, jestli jim byl udělen nějaký příkaz botmasterem. V poslední době se můžeme setkat i s případy, kdy k řízení botů slouží profily na sociálních sítích, na kterých botmaster zveřejňuje příkazy maskované jako příspěvky nebo komentáře. Jak již bylo řečeno, slabinou toho přístupu je fakt, že k znefunkčnění celé sítě stačí vyřadit z provozu jeden nebo několik málo C&C serverů.



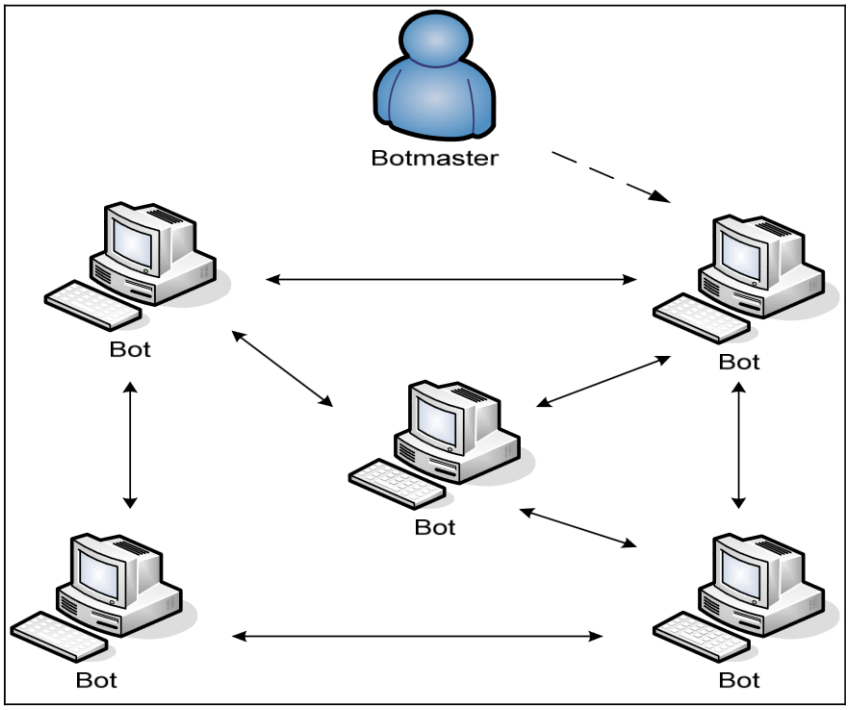
Obrázek 2.11: Architektura centralizovaného botnetu [29]

Decentralizované botnety

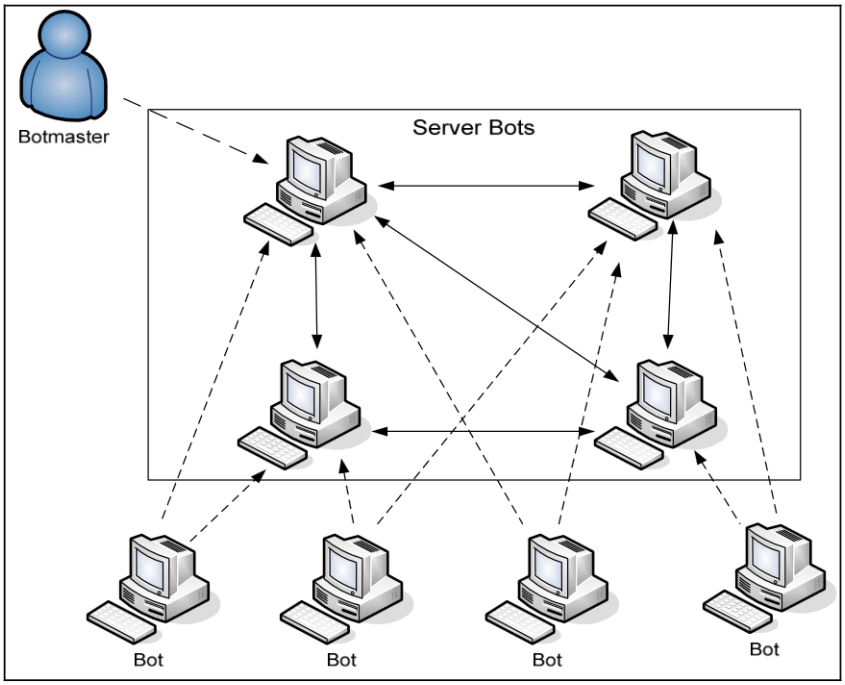
Decentralizované botnety fungují na principu peer-to-peer sítě, kde každý klient komunikuje s několika dalšími. Každý bot může zároveň plnit funkci klienta i serveru současně. Neexistuje zde tedy žádný řídící server, který by byl botům nadřazený. Botmasterovi tak stačí přístup k jakémukoli nakaženému zařízení v botnetu, pomocí kterého rozešle příkaz mezi ostatní. Na implementaci takového botnetu je botřeba vynaložit větší úsilí, které je ale kompenzováno větší robustností takovéto sítě. Ovšem i v tomto případě existují slabiny, které mohou vyřadit botnet z provozu, např. chyby v komunikačním protokolu, které umožní převzetí kontroly pomocí jediného bota.

Hybridní botnety

Hybridní botnety kombinují benefity obou předchozích řešení. Řadoví boti nekomunikují přímo s C&C serverem, ale využívají mezivrstvu speciálně určených botů, které jsou propojeni P2P sítí. Povely od botmastera jsou zadávány právě těmto *server botům* nebo *proxy*, kteří je mezi sebou rozdistribuují. Teprve od nich putují k „obyčejným“ botům.



Obrázek 2.12: C&C architektura decentralizovaného botnetu [29]



Obrázek 2.13: C&C architektura hybridního botnetu [29]

Kapitola 3

Detekce síťových útoků

K detekci síťových útoků a jejich eliminaci je možné použít speciální síťové prvky a detekční systémy. Tato kapitola se věnuje stručnému představení některých z nich.

3.1 Firewall

Firewall je síťový prvek sloužící k zabezpečení provozu na síti. Jde o software nebo samostatné zařízení nejčastěji nasazené na rozhraní dvou sítí s různou úrovní zabezpečení a důvěryhodnosti. Jeho funkce spočívá v monitorování datového provozu a rozhodování o jeho legitimnosti na základě sledovaných vlastností. Tento prvek dohlíží na dodržování bezpečnostních politik definovaných v síti, kterou chrání. Tyto politiky jsou definovány pomocí jednotlivých pravidel, která jsou do firewallu vložena. Jednotlivé pakety nebo celé jejich toky jsou při vstupu z vnější sítě podrobeny analýze a na jejich základě je rozhodnuto, zda budou přeposlány dále, nebo dojde k jejich zahození. Firewally prošly během svojí existence několika fázemi vývoje a lze je rozdělit do následujících generací:

- **Nestavový firewall** – Nestavový firewall nebo také paketový filtr je nejstarší variantou těchto zařízení. Jeho fungování je založeno na statické analýze paketů podle definovaných pravidel. Ta firewallu říkají, mezi kterými dvěma adresami a porty může probíhat komunikace a mezi kterými ne. Nestavový firewall tedy operuje na síťové a transportní vrstvě modelu TCP/IP. Výhodou těchto zařízení je jejich jednoduchost a vysoká propustnost. Naopak jejich slabinou je neschopnost kontrolovat data ve vyšších vrstvách nebo ve složitějších protokolech.

- **Stavový firewall** – Tento firewall umožňuje filtrovat pakety dynamicky a rozlišit, zda pakety patří k již navázanému spojení, nebo zda-li jde o inicializaci nového spojení. V této generaci se již přechází na pravidla

soustřeďující se na vlastnosti celých toků a nejen paketů. Díky tomu mohou snadněji detekovat anomálie v chování protokolů, sledovat délku jednotlivých relací a k jednotlivým požadavků přiřadit související odpovědi.

- **Aplikační firewall** – Aplikační firewall neboli aplikační brána v sobě kombinuje vlastnosti obou předchozích generací. Jde o řešení umožňující hloubkovou kontrolu komunikace a protokolů aplikační vrstvy, jako jsou HTTP, Telnet nebo FTP. K dosažení těchto vlastností je často nutné, aby firewall sloužil jako prostředník, tzv. proxy, při komunikaci mezi dvěma zařízeními. Pokud se například klient pokouší navázat spojení se serverem, brána vstoupí do procesu inicializace a klient se nejprve připojí k ní. Teprve poté dojde k vytvoření spojení mezi bránou a samotným serverem. Veškerá komunikace oběma směry tak prochází nejprve skrze aplikační bránu. Tento proces je většinou pro klienta i server transparentní [30].

- **Next-generation firewall** – Jde o poslední generaci firewallu, která v sobě zahrnuje i systémy IDS a IPS, které jsou detailněji popsány v následující části práce.

3.2 Intrusion Detection System

Intrusion Detection System (IDS) je systém umožňující sledování a analýzu síťového provozu. Jedná se o software nebo dedikované síťové zařízení, které slouží jako doplněk k dalším prvkům zajišťujícím bezpečnost, jako je například firewall. Role IDS je pasivní a spočívá v upozornění na možný probíhající útok nebo podezřelou aktivitu. Síťový provoz je takovým systémem monitorován, ale nijak do něj nezasahuje a neupravuje ho. Pokud IDS zaznamená potenciální hrozbu, ohlásí tuto skutečnost společně s dodatečnými informacemi definovaným způsobem. Nejčastěji jde o zápis do logovacího souboru, který je zpracováván dalšími systémy, jako například SIEM (Security Information and Event Management), které jsou schopné výstupy podrobněji analyzovat. IDS systémy můžeme rozdělit podle místa jejich nasazení do následujících kategorií [31]:

- **Host-based** IDS (HIDS) – Jedná se o IDS nasazený na koncové stanici. Jeho úkolem je kontrola provozu směřující k nebo pocházející z daného počítače a žádného jiného. HIDS ale může současně sloužit i k monitorování konfigurace samotného systému, na kterém je nasazen. Jde především o sledování logovacích souborů, změn souborového systému a přístupů k nastavení OS nebo administrátorských účtů.

- **Network-based IDS** (NIDS) – NIDS je na rozdíl od HIDS umístěn nejčastěji na rozhraní vnitřní a veřejné sítě stejně tak, jako firewall. Ob-

vykle se jedná o softwarové řešení s úzce specializovanými funkcemi, protože role takového IDS spočívá v monitorování veškerého provozu mezi sousedními sítěmi. Z těchto důvodů jsou zároveň kladeny větší požadavky na výkon zařízení, na kterém systém běží.

- **Distributed IDS** (DIDS) – Distribuovaný IDS často kombinuje oba předchozí způsoby nasazení a svoje uplatnění nachází v rozlehlejších sítích. DIDS je tvořen několika jednotlivými systémy, které vzájemně spolupracují, nebo jsou řízeny nadřazeným centrálním systémem, který agreguje jejich hlášení a poskytuje ucelený přehled o událostech v rámci monitorované infrastruktury.

K detekování jednotlivých událostí je využíváno analyzování obsahu jednotlivých paketů, nebo vlastností celých síťových toků neboli „flows“. Mezi sledované parametry nejčastěji patří frekvence jednotlivých paketů, chování stavových protokolů nebo příznaky v hlavičkách paketů. Podle způsobu, který IDS využívá k identifikaci podezřelých událostí, můžeme tyto systémy rozdělit do 3 skupin.

3.2.1 Detekce pomocí signatur

Způsob detekce pomocí signatur (signature-based detection) je nejzákladnější avšak velice účinný. Je založen na porovnávání obsahu komunikace s databází, obsahující signatury nebo také podpisy již známých hrozeb. Příkladem takové signatury může být textový řetězec v URL adrese, binární data obsažená v paketu nebo název či přípona souboru v příloze emailu. Tento přístup je nejčastěji implementován tak, že zmíněná databáze obsahuje soubor pravidel, které IDS instruují k tomu, jaké přesné vlastnosti provozu má sledovat. Každý paket je pak kontrolován, a pokud porovnávací pravidlo nalezne shodu, dojde k vygenerování události.

Výhodou této metody detekce je jeho jednoduchost a přesnost v případě již prozkoumaných a dobře popsaných hrozeb. Ovšem velice záleží na kvalitě dostupné databáze signatur a její pravidelné aktualizaci. Nejvýznamnější slabinou je fakt, že systém není schopen rozpoznat nové hrozby, jejichž signatury ještě nebyly do databáze zahrnuty. Dalším možným úskalím je to, že i samotní útočníci mohou mít přístup jak ke komerčním, tak i k veřejně dostupným databázím, které tyto systémy používají. To jim teoreticky umožňuje upravit útok takovým způsobem, že se jeho signatura změní a sníží se tak pravděpodobnost jeho odhalení [32].

3.2.2 Detekce pomocí anomáliích

Detekce založená na hledání anomálií (anomaly-based detection) rozpoznává hrozby za pomoci statistické analýzy síťového provozu. Systém srovnává charakteristiky komunikace s referenčním statistickým modelem, který popisuje

chování za normálních podmínek. Referenční model může vznikat dvěma způsoby – staticky a dynamicky.

V prvním případě jsou analyzovány vlastnosti provozu pozorovaného během určitého časového úseku, tzv. *training period*. Po jeho skončení dojde na základě získaných informací k odvození vlastností síťové komunikace, které bude IDS považovat za běžnou. Tento model pak zůstává neměnný. Problémem statického modelu tkví v tom, že se vlastnosti síťové komunikace mohou v čase měnit, a to například změnou zařízení nebo služeb ležících ve sledované sítí. V takovém případě pak dochází k falešným hlášením o hrozbách nebo naopak k jejich nedetekování.

Dynamický model je oproti tomu obnovován v určitých intervalech a je tak schopný se přizpůsobit potenciálním změnách charakteristik prostředí, ve kterém je nasazen.

Společným slabým místem obou těchto přístupů je zmíněná training period. Pokud je totiž útočníkovi alespoň přibližně známá doba, kdy dochází k prvotnímu trénování nebo obnovování modelu, může svými aktivitami do procesu zasáhnout. V případě, že se mu podaří ovlivnit statistické vlastnosti provozu v tomto období, bude systém považovat takový provoz za legitimní a nepodaří se mu odhalit ani budoucí aktivity útočníka.

Výhodou tohoto přístupu je jeho účinnost při detekci jak známých, tak i dosud neobjevených hrozeb. Na rozdíl od detekce pomocí signatur je navíc schopen odhalit i hrozby skryté v šifrovaném přenosu. To je ovšem vykoupeno složitější implementací společně s vyššími nároky na výkon a čas potřebný pro samotné uvedení do provozu [32].

3.2.3 Detekce pomocí analýzy stavových protokolů

Analýza stavových protokolů (stateful protocol analysis) porovnává pozorované chování protokolů s chováním, které je obecně považované za legitimní, a snaží se nalézt odchylky indikující možný útok. Referenční chování protokolů je obvykle popsáno v tzv. profilech popisujících standardní průběh komunikace pomocí daného protokolu. Profily jsou zpravidla vytvářeny autory konkrétních protokolů.

Metoda umožňuje lépe pracovat se stavovými protokoly transportní, síťové i aplikační vrstvy, protože rozeznává v jakém stavu se aktuální spojení nachází. Z toho plyne velmi důležitá vlastnost – schopnost párování požadavků a odpovědí. Díky tomu IDS detekuje mnohem širší škálu útoků zahrnující reflexivní a další útoky používající podvržené adresy.

Nevýhodou podobných systémů je jejich závislost na použitých profilech popisujících běžné chování protokolů. Některé protokoly nemusejí mít přesně definované chování v mezních situacích a závisí pak na konkrétní implementaci. Současně je potřeba profily aktualizovat s každou změnou v konkrétním protokolu. Další stinnou stránkou jsou i vyšší nároky na výpočetní zdroje

z důvodu nutnosti sledování a analýzy stavu velkého množství současně běžících spojení [32].

3.3 Intrusion Prevention System

Systémy Intrusion Prevention System (IPS) v sobě zahrnují funkcionality IDS a dále ji rozšiřují o možnost reagovat na detekované hrozby a aktivně zasáhnout do provozu v síti. Po zachycení podezřelé aktivity dochází ke spuštění předem definované akce, která může spočívat ve:

- vygenerování výstrahy stejně jako v případě IPS
- úpravě dat v paketu
- zahození podezřelého paketu
- okamžitém přerušení spojení
- ignorování události na základě výjimky

Podobné systémy nabývají na oblíbenosti díky jejich všestrannosti. Jelikož se jedná o jistou nadmnožinu detekčních systémů, můžeme se často setkat i s označením IDPS (Intrusion Detection and Prevention System). Ostatní aspekty, jako místo nasazení nebo způsoby detekce, jsou identické s těmi popsanými výše.

3.4 Snort

Snort patří mezi nejrozšířenější NIDS řešení současnosti, které je momentálně vyvíjené společností Cisco. Jedná se o open-source nástroj pro operační systémy Windows, UNIX a Linux, který umožňuje analyzování síťové komunikace v reálném čase. Díky tomu že je jeho zdrojový kód otevřený, jde o velmi rychle se rozvíjející IDPS s pravidelnými aktualizacemi a mnoha doplňky, které rozšiřují jeho funkce. Těmi základními jsou odposlech paketů, logování paketů a IDS funkcionalita. Detekce hrozeb je prováděna na základě signatur.

Pravidel pro rozpoznávání signatur existuje velké množství, protože je vytváří jak početná komunita uživatelů za účelem následného volného šíření, tak i komerční společnosti, které je prioritně zpřístupňují platícím zákazníkům. Snort je proto schopný detekovat rozsáhlou škálu podezřelých činností a útoků, jako například přetečení bufferu, (D)DoS útoky, skenování portů nebo aktivitu malwaru [33].

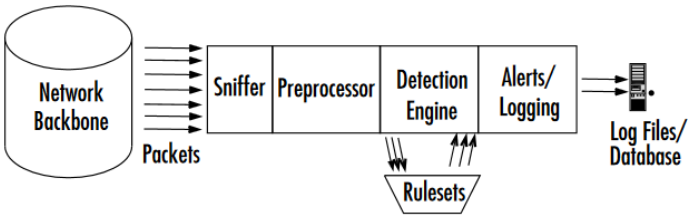
Pakety Snort analyzuje pomocí různých modulů v několika fázích, jejichž diagram je na obrázku 3.1. Funkce jednotlivých komponent jsou následující:

- **Sniffer** – V této komponentě dochází k zachytávání jednotlivých paketů ze sítě. Novější verze programu disponují abstraktní vrstvou, která umožňuje využití různých modulů, tzv. Data Acquisition Modules, operujících s fyzickými i softwarovými rozhraními. Pro snadnější testování pravidel a vývoj rozšíření je taktéž možné načítat pakety ze souborů PCAP.

- **Preprocessor** – Předzpracování slouží k dekódování a identifikaci jednotlivých vrstev a protokolů obsažených v paketech. Snort podporuje kromě mnoha protokolů linkové vrstvy také protokoly tunelovací, jako GRE, MPLS nebo IP in IP. Výstupem této fáze je datová struktura reprezentující obsah paketu, která je zpracovávána navazujícími moduly.

- **Detection Engine** – Tato komponenta je srdcem celého detekčního systému. Slouží ke kontrole obsahu paketů a hledání signatur podezřelého provozu podle pravidel z databáze. Samotná pravidla jsou jednoduché textové řetězce skládající se ze 2 částí. První z nich je hlavička, která určuje pro jaké IP adresy, protokoly a porty se pravidlo vztahuje a jaká akce se má provést při nalezení shody. Následuje tělo pravidla obsahující kontrolovaná kritéria, klasifikaci události a další doplňující informace.

- **Logging Component** – Zde jsou zpracovávána všechna upozornění generovaná detekčními moduly. Obvykle dochází k zaznamenání události do souboru, systémového logu nebo databáze. Snort podporuje několik formátů včetně JSON a binárního Unified2.



Obrázek 3.1: Diagram zobrazující architekturu programu Snort a fáze zpracování paketů [33]

Pravidla jsou důležitou součástí NIDS Snort a v dalších částech práce budou využívána k detekci vybraných útoků. Proto je vhodné se seznámit s jejich strukturou a některými základními parametry. Příklad jednoduchého pravidla, které upozorní na přicházející TCP spojení, je zachycen v ukázce 3.1. Význam jednotlivých částí jeho hlavičky a těla (v závorce) je následující:

1. `alert` – Při shodě se pouze ohlásí detekce, ale k jiné akci nedojde

2. `tcp` – Značí, že se pravidlo týká TCP protokolu

3. `any $EXTERNAL_NET -> $HOME_NET 80` – Kontrolován je provoz z adres mimo chráněnou síť a portů, který směřuje na port 80 adresy v proměnné HOME_NET. Proměnné HOME_NET a EXTERNAL_NET lze nastavit v konfiguračním souboru.
4. `sid:1000000` – Unikátní identifikátor pravidla
5. `flags: S` – Kontrolované segmenty musejí mít nastavený příznak SYN
6. `msg` – Zpráva zobrazená ve výsledném varování
7. `flow: stateless` – Nezáleží na aktuálním stavu TCP spojení

Parametrů pravidel existuje mnohem více a některé budou představeny a vysvětleny v dalších částech práce.

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 80 (
        sid:1000000;
        msg:"Incoming TCP connection";
        flow:stateless;
        flags:S;
)
```

Ukázka 3.1: Příklad pravidla pro systém Snort

Slabinou Snortu oproti jiným IDS je především to, že nepodporuje běh ve více vláknech. Jeho pravidla současně nemohou být příliš komplexní a musejí být upravena na míru prostředí, ve kterém je systém nasazen. V opačném případě dochází ke generování velkého množství falešně pozitivních detekcí.

Kapitola 4

Analýza útoků

Tato kapitola se zaměřuje na analýzu vlastností a dopadů vybraných útoků, které byly popsány v předchozích kapitolách. Jmenovitě jde o útoky odepření služby pomocí DNS amplifikace a útoku pomocí záplavy SYN segmenty.

4.1 DNS Amplifikace

Jedním z cílů této práce je demonstrovat jednotlivé útoky. Aby bylo možné DNS útok co nejpřesněji replikovat, bylo nejdříve nutné stanovit, jaké jsou parametry dosažitelné v reálném prostředí. Myšlenka spočívala v provedení analýzy zaměřené na to, jak vypadá běžné chování serverů v internetu. Z výsledků by pak bylo možné odvodit jejich možné nastavení. To přímo určuje maximální šířku pásma a amplifikační faktor potenciálního útoku. Takto zjištěné údaje by pak posloužily ke konfiguraci serveru v laboratorním prostředí a replikaci útoku, který by se svými vlastnosti blížil těm v reálném světě. Podobnému výzkumu se věnují autoři článku [34]. Jedná se však o práci z roku 2014 a proto neposkytuje přehled o aktuální situaci. V mezidobí bylo navíc zveřejněno několik návrhů na úpravu DNS protokolu například RFC8482, která je popsána dále v textu.

Byl uvažován především případ, kdy útočníci zneužijí již existující domény a odpovídající autoritativní servery namísto toho, aby vytvářeli domény vlastní. Pro tyto účely bylo nutné vytvořit katalog domén a odpovídajících serverů. Nalezené servery pak byly žádány o informace ohledně konkrétních domén a jejich odpovědi byly následně analyzovány.

4.1.1 Metodika sběru dat

Jako zdroj doménových jmen byl využit seznam Cisco Umbrella 1 Million. Jde o soupis 1 milionu nejnavštěvovanějších domén zvěřejňovaný společností Cisco, který je denně aktualizován. Jeho autoři ho sestavují na základě in-

formací získaných z cloudového bezpečnostního řešení Cisco Umbrella[6], které společnost provozuje. Jednou z jeho základních funkcí je právě poskytování služeb DNS překladu a detekce bezpečnostních hrozeb. Na rozdíl od ostatních veřejně dostupných seznamů domén, jako je třeba Alexa Top 500[7], je o poznání rozsáhlejší. Dalším rozdílem je i metodologie sběru dat a určování návštěvnosti jednotlivých domén. Ostatní projekty spoléhají na scripty umístěné majiteli na jejich stránky, popřípadě na uživatele, kteří se rozhodnou do projektu zapojit a nainstalují si do webového prohlížeče doplněk odesílající informace o jejich aktivitách na webu. Výsledkem je, že takto vytvořené katalogy domén zohledňují pouze HTTP provoz. Cisco Umbrella 1 Million je oproti tomu sestaven na základě DNS požadavků generovaných zařízeními, která využívají cloudové řešení Umbrella nebo veřejně dostupné služby OpenDNS. Tyto požadavky vznikají v důsledku síťové komunikace různých aplikací a nejsou tedy spojeny pouze s prohlížením webových stránek.

Zvolený dataset bylo potřeba před použitím předzpracovat. Šlo zejména o odstranění neplatných záznamů, protože jsou v něm zahrnuty i adresy spadající pod vyhrazené TLD, které nejsou dostupné z internetu, např. `.localhost` nebo `.home`. Výskyt takovýchto položek v seznamu je důsledkem způsobu sběru dat. Nezaručuje totiž, že dotazy odesílané na servery provozovatele infrastruktury jsou validními doménovými jmény. K identifikaci takovýchto položek posloužil seznam aktuálně platných TLD, který je poskytován organizací IANA[8]. Z původního počtu 1 000 000 položek jich bylo odstraněno 275 180. Posléze došlo k extrahování unikátních domén druhého řádu. Celkový počet takto získaných domén byl 156 795. Z praktických důvodů byl pro další analýzu vybrán náhodný vzorek 10 000 z nich.

Další krok spočíval ve vyhledání autoritativních serverů spravujících záznamy vybraných domén. Cílem bylo následně nashromáždit informace o příslušných serverech a určit, zda-li by bylo možné je potenciálně zneužít pro DNS útok. Sledované vlastnosti byly následující:

- podepsání domény pomocí DNSSEC
- maximální velikost odesílaných odpovědí při komunikaci pomocí UDP
- nakládání s dotazy typu ANY podle normy RFC8482
- velikost odpovědí na různé typy dotazů

Norma RFC8482 byla zavedena v roce 2019 a motivací k jejímu vzniku bylo zneužívání DNS dotazů typu `ANY` při amplifikačních útocích. Její autoři v ní navrhují změnu chování serveru a umožňují administrátorům omezit počet záznamů vrácených při tomto typu dotazu. Doporučení autorů je odpovědět

[6] https://umbrella.cisco.com/
[7] https://www.alexa.com/topsites
[8] https://data.iana.org/TLD/tlds-alpha-by-domain.txt

pouze záznamem typu `HINFO` obsahujícím textový řetězec „RFC8482“. Tím je do značné míry omezen amplifikační potenciál těchto dotazů při případném útoku [35].

Někteří správci však volí odlišný postup a vracejí jinou sadu záznamů obsahující např. pouze seznam jmenných serverů nebo některé záznamy spojené s DNSSEC. U serveru s popsaným chováním je pak složitější rozpoznat, zda danou normu podporuje. Proto se pro účely této práce za dodržování normy RFC8482 považuje navrácení záznamu HINFO tak, jak je v ní doporučeno.

Při plánování postupu sběru dat bylo nejprve uvažováno o využití DNS resolverů poskytovalete internetového připojení nebo veřejně dostupných alternativ, poskytovaných společnostmi jako je Google. Takový přístup byl ale vyhodnocen jako nevhodný. V obou případech hrozilo, že vzhledem k počtu dotazů a především jejich typů, které jsou využívány při útocích, by aktivita mohla být vyhodnocena jako pokus o zneužití služby. Tato teorie byla potvrzena v průběhu experimentu s malým počtem domén v případě Google Public DNS.

Pokud bylo na servery odesláno větší množství požadavků, vrácené odpovědi se významně lišily. Nejdramatičtější rozdíly byly pozorovány u dotazů ANY. První výsledek vrácený resolverem byl shodný s tím od autoritativních serverů. Při opakovaném dotazování ale začal vracet pouze zlomek záznamů. U ostatních, více obvyklých dotazů, např. typu A, byly odpovědi konzistentní. Popsané chování je pochopitelné, jelikož DNS resolvery společnosti Google jsou jedny z nejpoužívanějších, a proto musejí disponovat vysokým výpočetním výkonem i síťovou propustností. Jejich zneužití k potenciálnímu útoku by vedlo k citelným následkům. Provozovatelé si toho jsou vědomi a postoupili patřičné kroky k eliminaci rizik.

Pro účely sběru dat bylo vytvořeno vlastní řešení, které tvořily dva programy v jazyce Python, používající rozhraní knihovny `dnspython` pro pokročilou práci s DNS dotazy a odpověďmi. První z těchto programů sloužil k nalezení autoritativních DNS serverů spravujících zmíněné domény. Jeho funkce spočívala v rekurzivním procházení hierarchie DNS od kořenové domény, přes TLD, až po hledanou doménu. Současně zajišťoval překlad adres jednotlivých serverů, které se nacházely v jiné zóně.

IP adresy takto nalezených serverů pak byly ukládány do SQLite databáze. K zefektivnění celého procesu může program používat více vláken. Při hledání autoritativních serverů byly nejprve všechny domény vloženy do fronty reprezentované strukturou `Queue`. Jednotlivá vlákna z ní poté získávají jednu položku za druhou a provádějí průchod DNS stromem a překlad adres. Pro účely další optimalizace jsou nalezené zóny a odpovídající servery ukládány do vyrovnávací paměti. Před odesláním DNS dotazu je tato paměť konzultována a v případě shody vrátí adresu serveru zodpovědného za danou doménu. Tím je předcházeno nadbytečnému procházení DNS hierarchie.

Druhý program zajišťoval odesílání DNS dotazů a vyhodnocování korespondujících odpovědí. Adresy jednotlivých serverů jsou nejdříve načteny z da-

tabáze vytvořené v předchozí fázi. Opět je užita fronta úloh a jejich paralelního zpracování. Každá položka ve frontě odpovídá jedné DNS doméně a s ní spojených autoritativních serverů. Každé vlákno programu v rámci jedné úlohy odešle celkem 4 dotazy na každý uvedený server. Tyto dotazy jsou následujících typů:

- **ANY** – Dotaz na veškeré dostupné informace ohledně dané domény, který je při amplifikačních útocích využíván nejčastěji.
- **TXT** – Požadavek na TXT záznamy, které mohou obsahovat libovolné informace zadané správcem domény. V dnešní době jsou často využívány internetovými vyhledávači k verifikaci vlastníka domény.
- **NS** – Jeho výsledkem je výčet všech autoritativních DNS serverů, pod které daná doména spadá. Takovýchto serverů bývá obvykle několik, aby byla zajištěna redundance
- **RRSIG** – Výsledná odpověď by měla obsahovat záznamy RRSIG s kryptografickými podpisy jednotlivých sad záznamů stejných typů. Jde zároveň o indikátor toho, zda-li daná doména používá rozšíření DNSSEC

Program dále eviduje velikost odesílaných a přijímaných paketů, protože jde o faktor, který přímo určuje dopady případného útoku. Po obdržení odpovědi dojde k jejímu vyhodnocení a výsledky jsou uloženy do lokální paměti. Po zpracování všech úloh dojde k zapsání dat do souboru ve formátu CSV pro další zpracování statistickým softwarem.

4.1.2 Vyhodnocení výsledků

Tabulka 4.1 obsahuje výsledky první fáze sběru dat. Z celkového počtu 10 000 domén se podařilo nalézt a získat informace o 9 745 z nich. U zbylých 225 domén došlo k chybě během komunikace s DNS servery. Z toho 155 případů bylo zapříčiněno navrácením odpovědi s příznakem `NXDOMAIN`, resp. `SERVFAIL`, který indikuje neexistenci domény, resp. jinou nespecifikovanou chybu. V případě 76 záznamů nebylo možné kontaktovat ani jeden autoritativní server a to z důvodu vypršení časového limitu pro odpověď (timeout), který byl nastaven na 3 sekundy. Celkový počet zjištěných DNS serverů byl 8 199, což je méně něž celkový počet domén. Uvedený nepoměr v naměřených datech je způsoben faktem, že dnes většina služeb využívá cloudová řešení. V praxi je obvyklé, že každá doména má přiděleno několik autoritativních serverů. V cloudové infrastruktuře, ale každý z těchto serverů může spravovat několik desítek až stovek dalších domén.

Po rozdělení domén do skupin podle jejich nadřazené TLD bylo zjištěno, že průměrně 43.62% všech domén druhého řádu používá DNSSEC. Tabulka 4.3 obsahuje přehled o 10 TLD doménách, které se ve výsledcích vyskytovaly nejčastěji. Můžeme zde zpozorovat, že nejrozšířenější TLD domény mají

Záznamů celkem	Funkčních	Neexistujících	Vypršení spojení	Unikátních serverů
10 000	9 745	185	70	8 643

Tabulka 4.1: Výsledky sběru dat o doménách

TLD	Počet domén 2. řádu	DNSSEC
com	5 690	2 084 (36.63%)
net	619	187 (30.21%)
org	304	123 (40.46%)
io	195	80 (41.03%)
de	194	37 (19.07%)
xyz	133	109 (81.95%)
ru	102	23 (22.55%)
co	92	46 (50.00%)
it	71	29 (40.85%)
nl	68	17 (25.00%)

Tabulka 4.2: Přehled 10 nejčastějších TLD ve zkoumaném vzorku domén

podprůměrný počet subdomén užívajících DNSSEC. Tento výsledek však nemusí být odrazem skutečnosti, protože dataset obsahoval mnoho TLD s pouze jednou nebo několika málo doménami druhého řádu, které DNSSEC implementovali. To mohlo zapříčinit navýšení celkového průměru.

Další analýza se soustředila na chování jednotlivých serverů a potenciální amplifikační faktor různých typů dotazů. Tabulka obsahuje počty celkem provedených dotazů, počet dotazů, u kterých vypršel časový limit pro spojení, a nakonec počet těch, které skončily chybou. Během experimentu bylo zjištěno, že některé servery odpovídaly na validní dotazy chybovou hláškou. Takováto reakce vedla k vyvolání výjimky v knihovně dnspython, která zapříčinila, že nebylo možné získat informace o velikosti obdržené zprávy. S takovými případy bylo nakládáno tak, že ve výsledné statistice byla velikost odpovědi nastavena na hodnotu 0. Stejný postup byl zvolen v situaci, kdy server neodpověděl v průběhu zvoleného časového limitu. Zvolený postup měl za následek, že amplifikační faktor takovýchto dotazů vycházel nulový. Dané záznamy byly při dalším zpracování odstraněny, aby nezkreslovaly výsledná data.

Nejdříve byly pro jednotlivé typy dotazů zjištěny průměrná a maximální velikost v bajtech, značené $\overline{R}$, resp. R_{max}, a průměrný a maximální amplifikační faktor značené $\overline{A_f}$ a $A_{f_{max}}$. Naměřené hodnoty jsou k vidění v tabulce 4.4.

Celkem dotazů	Vypršení spojení	Chyba
28 095	1 548	282

Tabulka 4.3: Souhrn provedených DNS dotazů

Typy dotazu	$\overline{R}[B]$	$R_{max}[B]$	$\overline{A_f}$	$A_{f_{max}}$
ANY	209,38	4 062	4,97	113,86
TXT	341,44	3 150	8,11	80,77
NS	152,70	580	3,62	13,63
RRSIG	98,07	3 957	2,33	105,14

Tabulka 4.4: Výsledky měření velikosti odpovědí a amplifikačního faktoru

ANY

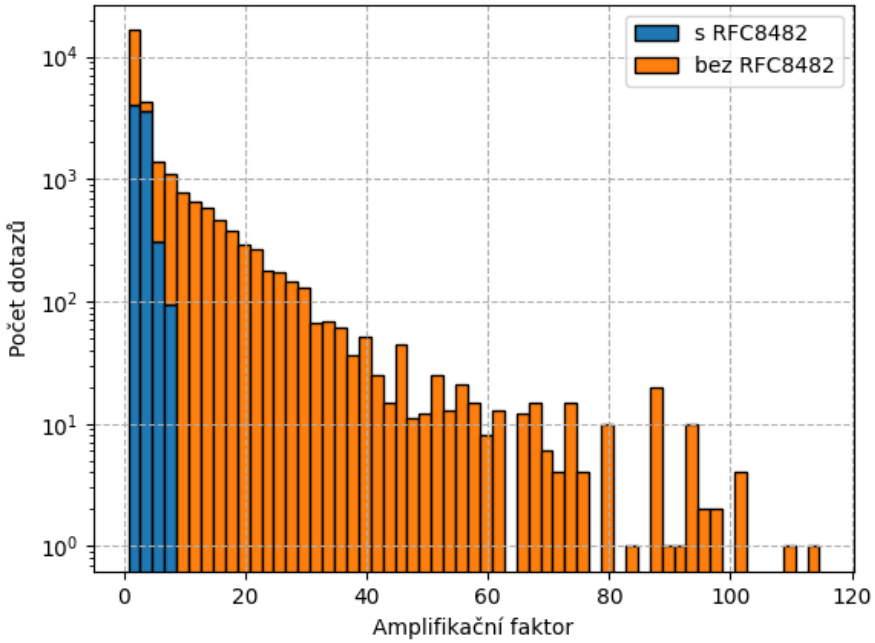
Nejvyššího amplifikačního faktoru bylo podle očekávání dosaženo dotazy ANY. Průměrná hodnota A_f je však výrazně nižší. Příčina tkví v tom, že relativně velké množství serverů se chová podle normy RDF8482 a odpovídá pouze minimálním množstvím informací, resp. dotazy odmítne zpracovat. Konkrétně bylo detekováno 7 953, resp. 264 takových odpovědí, což dohromady představuje 30,95% z celkového počtu odpovědí vztahujících se k danému typu dotazu. Průměrný A_f vztažený k ANY byl u těchto serverů 2,84. Naproti tomu u ostatních dosahoval hodnoty 7,08.

Histogram na obrázku 4.1 zachycuje četnosti velikostí odpovědí na požadavky ANY v závislosti na tom, zda daný server podporoval RFC8482. Šířka intervalů je rovna dvěma a osa y je pro lepší přehlednost v logaritmickém měřítku k počtu dotazů. Stejné parametry jsou zvoleny i u dalších grafů. Z výsledku je patrné, že servery s RFC8482 prakticky eliminují amplifikační potenciál těchto dotazů při potencionálním útoku.

TXT a NS

Následující měření se soustředila na amplifikační faktor u dotazů TXT a NS. Výsledná data jsou k vidění na obrázku 4.2. V případě NS dotazů byl vyvrácen jejich potenciál k možnému zneužití během útoku. Maximální A_f byl v jejich případě pouze 13,63 a průměr dosahoval 3,62. DNS záznamy tedy mohou obsahovat několik informací o mnoha serverech, ale jejich celková velikost je malá a útočníci mají k dispozici efektivnější způsoby jak DNS odpovědi zvětšit.

Příkladem jsou dotazy TXT, u kterých amplifikační faktor přesahoval hodnotu 8 a zastínil tak ostatní analyzované typy. Příčinou je to, že na rozdíl od typu ANY nebyly dotazy TXT většinou serverů nijak omezovány.

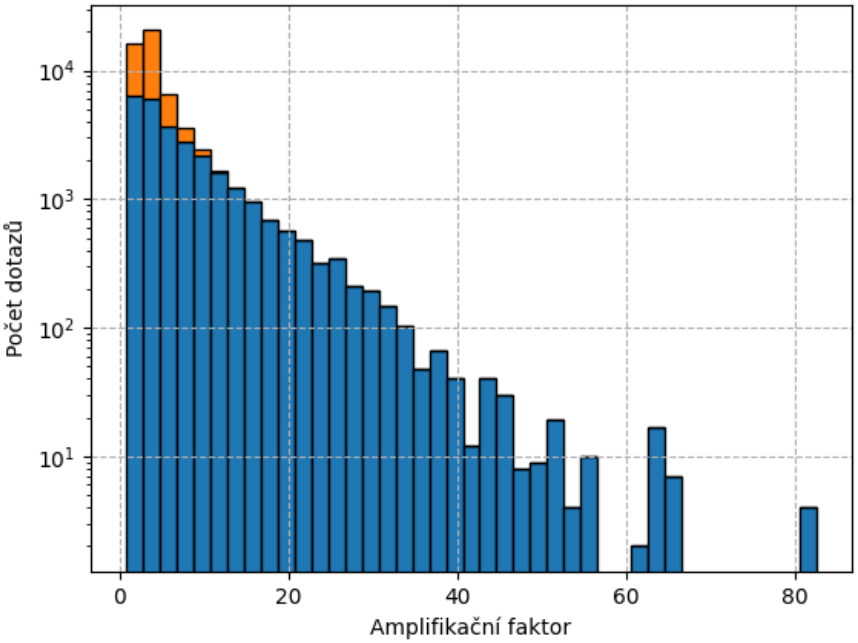


Obrázek 4.1: Amplifikační faktor dotazů ANY v závislosti na podpoře RFC8482

Vliv DNSSEC

Další snaha směřovala k ověření vlivu DNSSEC na amplifikační faktor dotazů ANY. Autoritativní servery, které zodpovídají za domény autentizované pomocí DNSSEC, musejí udržovat několik dodatečných typů záznamů. Jde především o záznamy DNSKEY, DS a zmíněný RRSIG. Tento fakt vede k nárůstu velikosti výsledné odpovědi.

Výsledek měření je možné vidět na obrázku 4.3. Graf zahrnuje pouze ty odpovědi, které nebyly servery nijak zkráceny či upraveny. Je patrné, že amplifikační faktor u domén zabezpečených pomocí DNSSEC je vyšší než u ostatních. Průměrná velikost A_f dosahovala u domén s DNSSEC hodnoty 32,89. V případě domén bez této funkcionality se rovnala pouze 5,51. Pozorovaný jev je v souladu s očekáváním a jeho příčinou jsou zejména položky RRSIG, jejichž velikost dosahuje i stovek bajtů (v závislosti na použitém podpisovém algoritmu). Tím výrazně zvyšují potenciál zneužití podobných záznamů k útoku.

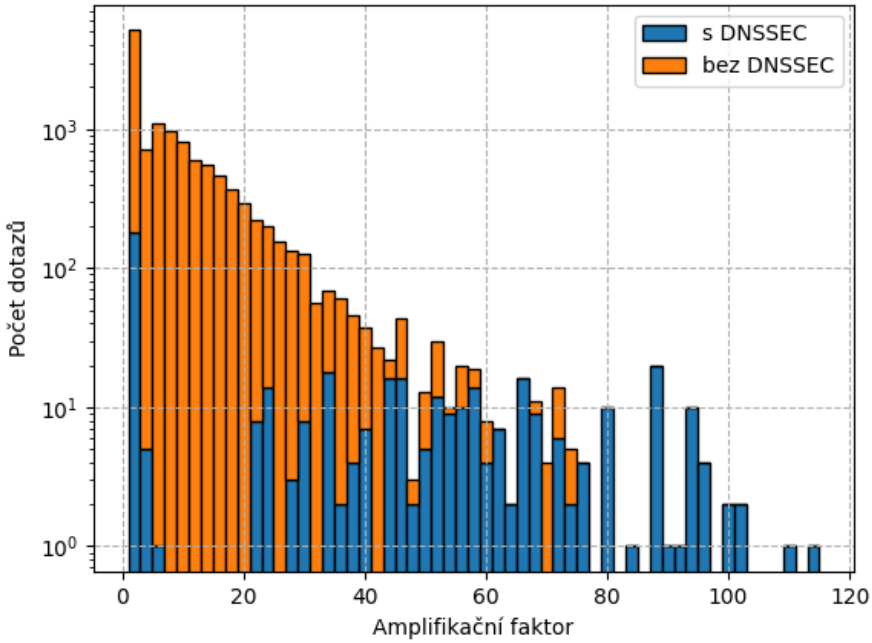


Obrázek 4.2: Amplifikační faktor dotazů TXT a NS

Odlišné nastavení serverů

Při analýze odpovědí autoritativních DNS serverů bylo dále pozorování i nekonzistentní chování různých serverů spadajících pod stejné domény. Byly zaznamenány různě veliké odpovědi na dotazy typu ANY. Vyvstala tím otázka, zda-li jsou všechny jmenné servery spravující tyto domény nakonfigurovány stejně. Metoda sběru dat umožnila servery s odlišným chováním identifikovat a podrobněji prozkoumat jejich odpovědi. Následně bylo odhaleno 652 domén, které měly dva a více autoritativních jmenných serverů, a alespoň jeden z nich poskytoval odlišné odpovědi než ostatní.

K nalezení příčiny bylo nutné porovnat výsledky stejných dotazů odeslaných na všechny autoritativní servery dotčených domén. Rozborem dat vyšlo najevo, že příčinou těchto odchylek bylo odmítnutí některých serverů odeslat požadované informace pomocí protokolu UDP. Toto chování bylo pravděpodobně způsobeno rozdílnou konfigurací EDNS na jednotlivých serverech. Rozšíření Extension Mechanisms for DNS (EDNS) zavádí, mimo jiné, podporu odesílání paketů větších než původních 512B definovaných standardem. Maximální velikost pak může dosáhnout až 4 096 bajtů. V případě, že paket překročí velikost definovanou v konfiguraci serveru, dojde v odpovědi k nastavení příznaku `TC` (truncated). Ten klientovi dává vědět, že zpráva byla oříznuta a měl by se pokusit dotaz opakovat přes protokol TCP.



Obrázek 4.3: Amplifikační faktor dotazů ANY v závislosti na podpoře DNSSEC (bez RFC8482)

Předpokládanými příčinami odlišného nastavení mohou být:

- Rozdílné parametry – Jednotlivé servery jsou kvůli zajištění vysoké dostupnosti často provozovány na oddělených počítačích a v různých sítích. Rozdílné nastavení tedy může reflektovat to, na jakém stroji server běží nebo jaké jsou parametry připojené sítě. Další možností je, že jde o méně výkonné servery, které slouží jako záložní a nejsou určeny pro odesílání velkého množství dat.
- Ponechání výchozího nastavení – Další možností je, že jde o pochybení správců, kteří ponechali servery ve výchozím nastavení. Například DNS server BIND9 v základní konfiguraci odesílá odpovědi velké maximálně 1 432 nebo 1 232 bajtů v závislosti na tom, zda-li je použit protokol IPv4 nebo IPv6. Důvodem je snaha předejít případné fragmentaci paketů.

4.1.3 Replikace útoku

Díky poznatkům získaným v předchozí části práce bylo možné určit vhodnou konfiguraci, která by nejlépe reflektovala možnosti útočníků v reálném světě, a útoky replikovat v laboratorním prostředí. Cílem bylo ověřit, jaké budou

pozorované dopady útoku na oběť a samotný DNS resolver. Pro tyto účely nebylo možné použít virtualizovaných strojů a to z důvodu možného zkreslení výsledků. Bylo tedy nutné navrhnout prostředí tvořené fyzickými počítači, propojenými pomocí ethernetového přepínače.

Testovací prostředí bylo tvořeno celkem 3 počítači s nainstalovaným operačním systémem Ubuntu. Podrobné parametry každého z použitých počítačů jsou v tabulce 4.5. Počítač v roli oběti disponoval integrovanou 100Mbit síťovou kartou. U ostatních byl použit USB ethernetový adaptér o stejné rychlosti. K propojení jednotlivých stanic byl použit router Compal CH7465 s vestavěným gigabitovým přepínačem.

Role	Operační systém	Procesor	Paměť	Síťové rozhraní
Oběť	Ubuntu 18.04.6 LTS	Intel i7-2600K	8 GB	100Mbit
DNS server	Ubuntu 18.04.6 LTS	Intel i7-6700HQ	16 GB	100Mbit
Útočník	Ubuntu 18.04.6 LTS	Intel i7-6700HQ	16 GB	100Mbit

Tabulka 4.5: Parametry a role počítačů v testovacím prostředí

Pro vykonání útoku bylo nutné připravit program, který slouží k odesílání paketů s DNS dotazem a podvrženou zdrojovou adresou. Z počátku se jako nejvhodnější programovací jazyk pro jeho implementaci jevil jazyk Python a to z důvodu existence mnoha uživatelsky přívětivých knihoven, které jsou k manipulaci s pakety určené. Zejména šlo o knihovnu `Scapy`[9]. Při testování, prováděném na počítači označeném jako „DNS resolver“ v tabulce 4.5, však vyšlo najevo, že toto řešení podávalo nedostatečný výkon a zvládlo generovat pouze okolo 800 paketů za sekundu. Další volba tedy padla na jazyk C a knihovnu `libnet`[10]. Výsledný program byl schopen stabilně odesílat přes 120 000 paketů za vteřinu a téměř zcela vytížit dostupné síťové rozhraní.

V dalším kroku byl zprovozněn rekurzivní DNS resolver. K tomu byl využit DNS server BIND9[11], který je jedním z nerozšířenějších open-source DNS řešení současnosti. Server byl nakonfigurován tak, aby odpovídal na dotazy z jakékoli zdrojové adresy a choval se tedy jako „open resolver“. Konfigurační soubor je v ukázce 4.1. Ostatní parametry zůstaly ve výchozím nastavení.

Během analýzy chování autoritativních serverů v předchozí části práce byl identifikován dotaz, který měl největší amplifikační faktor. Šlo o dotaz typu ANY, na který daný server odpověděl 4 062 bajty. Stejný dotaz byl tedy odeslán i na lokální resolver, aby ho vyhodnotil. Tím bylo zajištěno uložení výsledné odpovědi do vyrovnávací paměti. Server tak po dobu platnosti záznamu nemusel kontaktovat autoritativní server dané domény a na dotazy klientů odpovídal daty z cache. Prostředí pak bylo odizolováno od

[9] https://scapy.net/
[10] https://github.com/libnet/libnet
[11] https://www.isc.org/bind/

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

        dnssec-validation auto;

        auth-nxdomain no;

        listen-on port 53 { any; };

        allow-query { any; };
        allow-recursion { any; };
        recursion yes;

        edns-udp-size 4096;
        max-udp-size 4096;
};
```

Ukázka 4.1: Konfigurační soubor DNS serveru

sítě internet vypnutím příslušného síťového rozhraní. Tím bylo zabráněno neplánované interakci s vnější sítí.

K monitorování vytížení síťových rozhraní a procesorů byl využit nástroj `sar`[12], který běžel na všech počítačích a sbíral informace v intervalu jedné sekundy během celého experimentu. Útok trval 30 vteřin a byl opakován celkem desetkrát. Výsledné hodnoty byly zprůměrovány.

Naměřené hodnoty odeslaných a přijatých paketů, množství přijímaných a odesílaných dat v kilobajtech za vteřinu a celkového vytížení procesorů je možné vidět v tabulce 4.6. Grafy na obrázcích 4.4 a 4.5 zachycují počty odesílaných a přijímaných paketů na všech 3 počítačích v průběhu jednoho vybraného opakování experimentu.

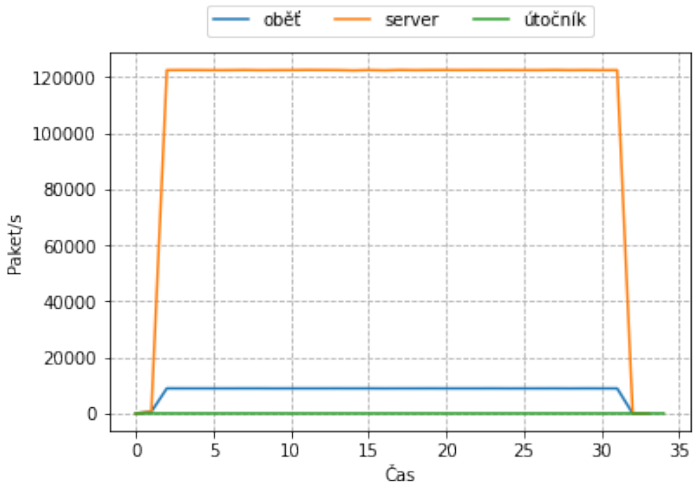
Z daných dat je patrné, že během experimentu došlo k vytížení téměř celé kapacity 100Mbit linky. Bližší prozkoumání zátěže procesorů na všech zúčastněných zařízeních ukázalo velký nárůst zejména na straně DNS serveru, který dotazy zpracovával. Průměr vytížení CPU po dobu testu u něj dosahoval 57%. Na straně oběti bylo dosaženo vytížení pouze 1,91%, což nepředstavuje výraznější nárůst a mohlo být způsobeno procesy na pozadí. Z měření v daných podmínkách vyplývá, že největší zátěži je při tomto druhu útoku vystavena síťová infrastruktura mezi obětí útoku a DNS resolverem a pak resolver samotný.

[12] https://github.com/sysstat/sysstat

Počítač	CPU	Pkt_{in}/s	Pkt_{out}/s	kB_{in}	kB_{out}
Oběť	1,91 %	8 933,72	1,38	11 998,79	0,58
DNS server	57,00 %	122 519,05	8 934,05	9 332,71	11 998,79
Útočník	5,17 %	0,76	122 527,85	0,06	9 333,60

Tabulka 4.6: Výsledky měření zatížení CPU a síťových prostředků během amplifikačního DNS útoku

Obrázek 4.4: Počet odeslaných paketů za sekundu



Obrázek 4.5: Počet přijatých paketů za sekundu

4.1.4 Obrana

Předchozí analýza ukázala, jak je při vhodných podmínkách možné konkrétními typy DNS dotazů dosáhnout relativně vysokého amplifikačního faktoru. Nyní je vhodné představit nejčastější metody obrany proti takovýmto útokům. Jde o relativně jednoduchá opatření, která jsou však velmi účinná, a lze díky nim útoku předejít nebo výrazně omezit jeho dopady. Tyto možnosti můžeme rozdělit do dvou kategorií podle místa, kde mohou být nasazeny. Těmi jsou:

- síťová infrastruktura
- dotčené DNS servery a resolvery

První kategorie se zaměřuje především na základní mechanizmus umožňující amplifikační DNS útoky, kterým je IP spoofing. Pokud nemůže útočník podvrhnout zdrojovou adresu v paketu s DNS požadavkem, je riziko podobného útoku prakticky eliminováno. Jak již bylo zmíněno v kapitole 2.2.1, nejefektivnější obranou je zavedení *ingress filtering* na okrajových směrovačích jednotlivých poskytovatelů internetového připojení. Ač jde o efektivní metodu, je nutné, aby ji aplikovala drtivá většina ISP, jinak svoji účinnost postrádá. Další možností je nasazení pokročilejších metod filtrování provozu na úrovni autonomních systémů za využití dodatečných informací poskytnutých protokolem BGP [36].

V neposlední řadě se nabízí nasazení firewallu, který může blokovat určité typy DNS dotazů jako ANY, které nejsou pro fungování protokolu příliš podstatné. Nejedná se však o efektivní řešení, protože útočníci mohou zvolit jiné dotazy jako TXT nebo RRSIG s podobným efektem. Kýženého výsledku lze dosáhnout užitím stavového firewallu, který je, navzdory bezstavovosti UDP, schopný udržovat informace o stavu jednotlivých spojení. Může tak odfiltrovat DNS odpovědi, u kterých nebyl zaregistrován žádný předchozí požadavek.

Opatření na úrovni samotných serverů spočívají především ve správné konfiguraci. V případě autoritativních DNS serverů jde například o omezení počtu záznamů vracených na dotazy ANY. Předchozí část práce ovšem ukázala, že se nejedná o univerzální řešení, které by se s danou hrozbou spolehlivě vyrovnalo. Dalším obecným doporučením je pak aplikace tzv. *rate limiting*. Jde o omezení maximálního povoleného počtu dotazů od jednoho klienta za určitý časový úsek. Toto řešení s sebou však při nasazení na autoritativním serveru přináší další hrozbu v podobě možného zneužití k jinému DoS útoku. Útočníci při něm mohou odesílat na server požadavky s podvrženou IP adresou, která odpovídá adrese legitimního DNS resolveru, patřícího například ISP. Server by pak požadavky o samotného resolveru ignoroval a tím omezil dostupnost služeb pro jeho uživatele. V případě samotných resolverů je důležité nastavit je tak, aby se nechovaly jako „otevřené“, tedy omezit rozsah IP adres, od kterých budou přijímat DNS dotazy.

Pozornost byla následně upřena na otestování efektivity možných způsobů obrany nastíněných v předchozí části textu. Konkrétně se jednalo o zavedení omezení počtu dotazů za sekundu. U použitého serveru Bind9 lze tuto funkci aktivovat pomocí parametru `responses-per-second` v souboru `named.conf.options`. Povolené rozmezí nastavení je 0 (bez limitu) až 1 000. Proto byly pro testování zvoleny hodnoty 1 000, 800, 600 a 400. Celá série měření byla opakována stejným způsobem, jako při předchozích experimentech.

Naměřené hodnoty při omezení na 1 000 požadavků za sekundu jsou v tabulce 4.7. Výsledky pro ostatní zkoumané hodnoty se od těchto nijak významně nelišily a proto nejsou uvedeny.

Z dat je patrné mírné snížení celkového vytížení DNS resolveru. Počet paketů odesílaných směrem k oběti byl několikrát vyšší, avšak došlo k markantnímu snížení jejich velikosti. Analýza paketů v programu Wireshark ukázala i příčinu tohoto jevu. DNS resolver požadavky stále zpracovává a odesílá odpovědi, které ale obsahují pouze původní dotaz a doplňující informace vztahující se k EDNS. Velikost odpovědí je tak zredukována na několik desítek bajtů.

Počítač	CPU	Pkt_{in}/s	Pkt_{out}/s	kB_{in}	kB_{out}
Oběť	1,70 %	32 985,57	1,07	2 567,01	0,73
DNS server	50,63 %	122 549,11	33 303,67	9 331,10	2 536,80
Útočník	6,92 %	0,97	122 502,70	0,35	9 327,34

Tabulka 4.7: Výsledky měření při zavedení omezení maximálního počtu požadavků na straně DNS serveru

4.1.5 Detekce

Pro detekování amplifikačního DNS útoku byl zvolen zmiňovaný NIDS Snort. V ideálním případě by bylo možné útok detekovat sledováním UDP sezení. Pokud by byla zaznamenána přicházející DNS odpověd bez předchozího dotazu, mohl by program s vysokou mírou pravděpodobnosti ohlásit potenciální útok nebo chybné užití protokolu.

Úskalím Snortu je, že neumožňuje párovat DNS požadavky a odpovědi používající UDP protokol. Preprocesor `stream5`, který je dodáván s nejnovějšími verzemi programu, sleduje TCP sezení a dává pravidlům možnost reagovat na jejich aktuální stav. Stream5 lze nastavit i na sledování sezení u protokolu UDP, ale pouze pro interní účely. Aby tedy bylo možné aplikovat výše navržený postup, musel by být implementován nový preprocesor s danou funkcionalitou.

Útok ale lze stále detekovat i pomocí pravidel, které budou sledovat frekvenci přijímaných DNS paketů a jejich obsah. Víme, že během útoku k serveru

směřuje velký počet DNS odpovědí a v každé z nich je zahrnut i původní dotaz. Navrhované pravidlo tedy bude kontrolovat:

- Zda-li jde o DNS odpověd nebo dotaz. K rozeznání těchto případů slouží příznak ve 3. bajtu v hlavičče DNS protokolu. Pokud je nastavený na 1, jedná se o odpověď.
- Typ původního dotazu. Každá odpověď obsahuje i dotaz, ke kterému se vztahuje. Lze tedy odvodit, jestli se jednalo o typ spojovaný s podobnými útoky.
- Počet přijatých paketů z konkrétních adres a se zdrojovým portem 53.

Výsledné pravidlo je v ukázce 4.2. O rozpoznání DNS odpovědi se stará direktiva `byte_test`. V tomto případě vezme 1 bajt na pozici 2 (počítano od 0) a provede logický AND s hodnotou `0x80`. Pokud je výsledek nenulový, test je splněn a s určitostí víme, že šlo o DNS odpověď.

V případě odvození typu původního datazu, je situace složitější. Pozice kontrolovaného bajtu leží za dotazovaným doménovým jménem a tudíž se mění s jeho délkou. Snort ovšem umí v paketu vyhledávat binární nebo textové řetězce, a to i od určité pozice. Původní dotaz vždy začíná za DNS hlavičkou, tedy na 12. bajtu, přičemž jeho maximální délka je omezena na 255 znaků. Jméno domény má proměnlivou délku, ale vždy je ukončeno nulovým bajtem `0x00`. Následující 2 bajty reprezentují typ dotazu a první z nich je opět nulový. Další určuje typ dotazu a za ním se nacházejí bajty `0x00 0x01`, které značí nejpoužívanější třídu dotazu IN. Pokud tedy chceme identifikovat odpovědi na dotaz ANY, značený bajtem s hodnotou `0xff`, musíme hledat posloupnost `0x00,0x00,0xff,0x00,0x01` na pozici 12 az 273.

Měření počtu přijatých paketů je zajištěno parametrem `detection_filter`, který ohlásí varování při přijetí 20 paketů z jedné IP adresy za dobu 1 vteřiny.

Použité pravidlo správně detekovalo probíhající útok za použití dotazů ANY a jednoduchou úpravou ho lze upravit i pro ostatní varianty. Vygenerované varovaní je k vidění v ukázce 4.3.

```
alert udp any 53 -> $HOME_NET any (
        sid: 1000001;
        classtype: attempted-dos;
        msg: "Possible DNS amplification (ANY)";
        byte_test: 1, &, 0x80, 2;
        content:
                "|00 00 ff 00 01|",
                offset 12,
                depth 261;
        detection_filter:
                track by_src,
                count 20,
                seconds 1;

)
```

Ukázka 4.2: Pravidlo pro detekci příchozích DNS odpovědí na dotaz ANY

```
{
   "seconds":1651274887,
   "action":"allow",
   "class":"Attempted Denial of Service",
   "dir":"C2S",
   "dst_addr":"192.168.0.10",
   "dst_port":39603,
   "iface":"enp7s0",
   "msg":"Possible DNS amplification attack (ANY)",
   "pkt_len":1171,
   "priority":2,
   "proto":"UDP",
   "rule":"1:1000002:0",
   "sid":1000002,
   "src_addr":"52.224.90.37",
   "src_port":53,
   "udp_len":1151,
   "timestamp":"04/30-01:28:07.309984"
}
```

Ukázka 4.3: Upozornění programu Snort no probíhající amplifikační útok

4.2 SYN flood

V rámci demonstrace toho útoku byl ověřován jeho dopad na dostupnost webových stránek poskytovaných serverem, na který útok cílil, a účinnost některých způsobů obrany.

4.2.1 Replikace útoku

Testovací prostředí se skládalo ze stejných počítačů a přepínače jako v předchozím experimentu. Změnily se pouze jejich role, které jsou pro přehlednost uvedeny v tabulce 4.8.

Role	Operační systém	Procesor	Paměť	Síťové rozhraní
HTTP server	Ubuntu 18.04.6 LTS	Intel i7-2600K	8 GB	100Mbit
Útočník	Ubuntu 18.04.6 LTS	Intel i7-6700HQ	16 GB	100Mbit
Měřící stanice	Ubuntu 18.04.6 LTS	Intel i7-6700HQ	16 GB	100Mbit

Tabulka 4.8: Parametry a role počítačů v testovacím prostředí

Jako webový server byl použit Apache HTTP Server verze 2.4.29. Testovaná webová stránka byla generována pomocí redakčního systému Wordpress 5.9.3, který doplňovala relační databáze MySQL ve verzi 5.7.37. Volba konfigurace využívající redakční systém s databází byla upřednostněna před statickou webovou stránkou, protože se jedná o velmi často používanou kombinaci v reálném prostředí. Redakční systém i webový server byl ponechán v základním nastavení bez instalace dodatečných rozšíření.

Další stanice sloužila k měření dostupnosti a výkonu webových stránek. Ke sběru dat byla použita aplikace JMeter[13], která umožňuje spouštět předdefinované testovací scénáře a v rámci nich sledovat různé výkonnostní metriky. Pro účely experimentu byl vytvořen scénář, který pomocí odesílání HTTP požadavků napodoboval interakci uživatelů s webem. Konkrétně bylo simulováno 1000 přístupů na domovskou stránku redakčního systému a to v průběhu 20 sekund.

Během experimentů byla sledována průměrná doba odezvy webových stránek a její směrodatná odchylka, počet HTTP požadavků, které skončily chybou, průměrné vytížení CPU serveru a počet příchozích a odchozích paketů na straně serveru. Každé měření bylo opakováno celkem desetkrát a výsledné hodnoty byly průměrovány.

K simulaci útoku byl použit program `hping3`[14], který byl nakonfigurován parametry uvedenými v ukázce 4.4. Přepínač `S` nastaví SYN příznak v odchozích segmentech, `p` určuje port, na který útok směřuje. V tomto případě

[13] https://jmeter.apache.org/
[14] http://www.hping.org/

```
hping3 -S -p 80 --flood --rand-source SERVER_IP
```

Ukázka 4.4: Parametry programu hping3

byl nastaven na stejný port, na kterém naslouchal webový server. Přepínač `flood` zajistí, že jsou jednotlivé segmenty odesílány bez jakékoliv prodlevy mezi nimi. Argument `rand-source` způsobí, že jsou zdrojové adresy v segmentech nastaveny náhodně. Poslední parametr určuje cílovou adresu útoku.

V předchozí části práce, věnující se popisu principu SYN flood útoku, byly zmíněny 3 metody aplikovatelné na straně serveru, které mohou omezit následky útoku. Jednalo se o povolení SYN cookies, zvětšení SYN queue a omezení počtu opakovaného odesílání SYN+ACK segmentů. V operačním systému Linux je možné ovlivnit tyto vlastnosti pomocí konfiguračních proměnných

- `net.ipv4.tcp_syncookies`
- `net.ipv4.tcp_max_syn_backlog`
- `net.ipv4.tcp_synack_retries`

V rámci demonstrace útoku byly na serveru SYN cookie deaktivovány. Ostatní vlastnosti systému byly ponechány ve výchozím nastavení s hodnotami `tcp_synack_retries` = 5 a `tcp_max_syn_backlog` = 512.

Experiment měl určit chování serveru během dvou situací. Jedna odpovídala běžnému provozu, kdy server nebyl vystaven žádnému útoku. Posléze bylo měření opakováno s tím rozdílem, že na server směřoval záplavový útok. Ve druhém případě byl konkrétní postup takový, že nejprve došlo k aktivaci programu hping a následně byl spuštěn testovací program v aplikaci JMeter.

Naměřené hodnoty jsou k vidění v tabulce 4.9. Můžeme vidět, že během útoku došlo k úplnému znepřístupnění webových stránek pro běžné uživatele, protože chybovost požadavků vzrostla na 100%. Všechny chyby byly způsobeny vypršením maximálního limitu pro navázání spojení. Doba odezvy byla počítána pouze pro úspěšné HTTP požadavky a proto v tomto případě není její hodnota uvedena.

Všechna tato pozorování jsou očekávaným důsledkem záplavového SYN útoku, protože byl server zahlcen podvrženými SYN segmenty, jejichž množství řádově převyšovalo to generované legitimním provozem. Příčinou pozorovaného snížení celkové zátěže procesoru na straně serveru byl fakt, že žádný z požadavků klientů se nedostal k samotné aplikaci Apache, která by je mohla obsloužit.

4.2.2 Obrana

V rámci analýzy jednotlivých nastíněných metod obrany byla pozornost nejdříve upřena na efekt zvětšení fronty pro příchozí spojení. Snahou bylo zod-

Stav	Doba odezvy [ms] Průměr	Odchylka	Chybovost	CPU	Pkt_{in}/s	Pkt_{out}/s
Bez útoku	27,38	3,32	0 %	21,53 %	454,16	911,65
Při útoku	–	–	100 %	1,22 %	148 011,63	110,03

Tabulka 4.9: Výslekdy měření dopadů útoku na server ve výchozím nastavení

povědět otázku, zda může tato technika sama o sobě zmírnit nebo dokonce eliminovat dopady útoku.

Testované velikosti fronty byly 4096, 8192, 16384, 32768 a 65536. Počet opakovaného odesílání SYN+ACK segmentů zůstal nezměněn, tzn. na hodnotě 5. Během měření ovšem vyšlo najevo, že ani jedno z těchto nastavení nezvrátilo následky záplavového útoku. Výsledky byly prakticky identické jako ty získané při replikaci útoku ve výchozím nastavení serveru. Chybovost požadavků byla ve všech případech stoprocentní a nedošlo ani k výrazné změně zátěže procesoru.

Dalším logickým krokem tedy bylo pokusit se nalézt kombinaci nastavení velikost SYN fronty a počtu znovuodeslání SYN+ACK segmentu. Navyšování hodnoty `tcp_synack_retries` by vzhledem k principu fungování útoku nemohlo přinést žádný pozitivní efekt. Pouze by prodloužilo dobu, po kterou systém čeká na odpověď. Byly tedy ověřovány pouze hodnoty 5 a méně. Postup byl takový, že pro každou hodnotu nastavení `tcp_synack_retries` byly znovu testovány velikosti fronty tak, jako v případě zmíněném výše.

`syn_backlog`	Doba odezvy [ms] Průměr	Odchylka	Chybovost	CPU	Pkt_{in}/s	Pkt_{out}/s
4 096	1 718,33	809,47	99,85 %	0,23 %	148 306,66	194,52
8 192	7 980,08	4 793,35	99,75 %	0,31 %	148 008,88	188,93
16 384	4 925,00	6 283,04	99,81 %	0,68 %	148 196,62	183,95
32 768	7 547,25	6 101,81	99,80 %	1,11 %	148 433,62	336,53
65 536	6 783,19	5 557,68	99,05 %	1,13 %	148 274,74	339,91

Tabulka 4.10: Vliv nastavení SYN fronty a znovuodesílání SYN+ACK

Pro všechna nastavení `tcp_synack_retries` vyšší něž 2 se situace opakovala. Jediná změna nastala, pokud byl SYN+ACK segment opětovně odesílán pouze jednou. Výsledky s tímto nastavením jsou k vidění v tabulce 4.10. Přestože procento vyřízených dotazů již nebylo nulové, jednalo se pouze o jednotky požadavků, které byly obslouženy. Jistá funkčnost webových stránek, byť velmi výrazně omezená, se tedy navrátila. Nelze však konstatovat, že by kombinace těchto nastavení byla za dané situace účinným řešením.

Je také třeba vzít v úvahu, že podobné nastavení není vhodné pro nasazení v reálném provozu i bez probíhajícího útoku. To se týká především `tcp_synack_retries`. Pouze jedno opakované odeslání SYN+ACK může vést k nedostupnosti služby pro uživatele, kteří disponují nestabilním připojením nebo pokud dochází ke ztrátám segmentů v důsledku chyb přenosu.

V posledním kroku zbývalo otestovat, jak útok ovlivní povolení SYN cookies na straně severu. Toto měření bylo prováděno s následujícím nastavením:

- `net.ipv4.tcp_syncookies` = 1
- `net.ipv4.tcp_max_syn_backlog` = 512
- `net.ipv4.tcp_synack_retries` = 5

Výsledné hodnoty jsou zobrazeny v tabulce 4.11. Je patrné, že během útoku vzrostlo zatížení procesoru serveru zhruba o 9%. Současně se podstatně zvýšila i doba odezvy webových stránek a to více než 5krát. Přestože byly veškeré požadavky úspěšně zpracovány, došlo oproti normálnímu stavu ke znatelnému nárůstu odchylky doby odezvy. To indikuje, že server byl pod vyšší zátěží a některé požadavky odbavoval déle než ostatní. Příčinou může být celkové vytížení síťového rozhraní a také to, že kalkulace hodnot SYN cookie používaná v odesílaných SYN+ACK segmentech zabírá část výkonu procesoru.

Stav	Doba odezvy [ms]		Chybovost	CPU	Pkt_{in}/s	Pkt_{out}/s
	Průměr	Odchylka				
Bez útoku	39,40	35,56	0 %	22,33 %	284,84	894,36
Při útoku	207,25	360,01	0 %	31.48 %	148 472,65	132 592,49

Tabulka 4.11: Výslekdy měření s aktivními SYN cookies

Přesto lze z dostupných dat konstatovat, že tato metoda je ze všech zmíněných tou nejefektivnější. Následky útoku sice nejsou zcela eliminovány, ale dostupnost služeb zůstala, na rozdíl od předchozích zkoumaných přístupů, zachována.

4.2.3 Detekce

K detekci záplavového SYN útoku můžeme opět využít program Snort. Nabízejí se dva možné způsoby, jakými lze detekční pravidlo napsat. Oba jsou založeny na sledování počtu přijatých SYN segmentů za určitý čas. Jeden kontroluje počet takových segmentů podle zdrojové adresy a druhý podle cílové (v našem případě adresy chráněného serveru).

První varianta není příliš vhodná, protože nemusí detekovat útok používající podvržení IP. Své uplatnění může ale nalézt v kombinaci s reputačními

databázemi nebo s tzv. černými listinami (blacklist), které evidují adresy spojované s útoky nebo podezřelými aktivitami. Snort obsahuje tzv. Reputation Preprocessor, který umožňuje aplikovat zvláštní pravidla pouze na adresy obsažené na takovém seznamu.

Preferovaná varianta je v tomto případě ta, která kontroluje celkový počet přicházejících SYN segmentů směrem k serveru. Její slabinou je, že je nejdříve nutné odhadnout, jaký počet nově navazovaných spojení je považován za běžný. Pokud je tento krok podceněn nebo se například neočekávaně zvýší počet návštěvníků webu, bude systém generovat falešná hlášení o útoku.

```
alert tcp any any -> $HOME_NET 80 (
        sid: 1000002;
        classtype: attempted-dos;
        flags: S;
        msg: "Possible SYN flood detected";
        flow: stateless;
        detection_filter:
                track by_dst,
                count 10000,
                seconds 10;
)
```

Ukázka 4.5: Pravidlo pro detekci SYN flood útoku

Pravidlo, viditelné v ukázce 4.5, spolehlivě rozpoznalo probíhající útok, ale vzhledem k velkému množství segmentů odesílaných na server, bylo množství hlášení vysoké. Snort umožňuje takovým situací předcházet pomocí parametru pravidla `event_filter`, který limituje maximální počet hlášení za časový interval. Nicméně pro účely demonstrace detekce je pravidlo dostačující. Varování ve formátu JSON vygenerované během útoku je zachyceno v ukázce 4.6.

```
{
   "seconds":1650930916,
   "action":"allow",
   "class":"Attempted Denial of Service",
   "dir":"C2S",
   "dst_addr":"192.168.0.10",
   "dst_port":80,
   "iface":"enp7s0",
   "msg":"Possible SYN flood detected",
   "pkt_len":40,
   "priority":2,
   "proto":"TCP",
   "rule":"1:1000001:0",
   "sid":1000001,
   "src_addr":"243.79.67.23",
   "src_port":63398,
   "tcp_ack":286423820,
   "tcp_flags":"******S*",
   "tcp_len":20,
   "tcp_seq":895414781,
   "tcp_win":64,
   "timestamp":"04/26-01:55:16.286856"
}
```


Ukázka 4.6: Upozornění programu Snort na možný útok

Kapitola 5

Demonstrační prostředí

Tato kapitola se věnuje návrhu architektury demonstračního prostředí. Pro jednoduchost a přenositelnost řešení byl k vytvoření prostředí použit virtualizační nástroj VirtualBox[15]. Nastavení a správa jednotlivých virtuálních strojů je řízena pomocí programu Vagrant[16]. Všechny používají obraz s operačním systémem Ubuntu 18.04 64-bit. Verze operačního systému a použitého softwaru jsou identické s těmi použitými během analýzy útoků v předchozí kapitole.

Příloha práce zahrnuje samostatné složky, které obsahují potřebné soubory pro zprovoznění jednotlivých demonstračních prostředí. Každé prostředí je konfigurováno pomocí souboru `Vagrantfile` a lze ho zprovoznit otevřením složky s Vagrantfile v terminálu a zadáním příkazu „`vagrant up`“. Připojit se na virtuální stroje je možné vykonáním příkazu „`vagrant ssh JMENO_STROJE`“. Výchozí uživatelské jméno a heslo jsou `vagrant` a `vagrant`.

V podsložce `bootstrap-scripts` se nacházejí podpůrné shell skripty k nastavení jednotlivých strojů. Podsložka `config` obsahuje doplňující soubory, jako jsou konfigurační soubory serverů nebo pravidla pro program Snort. Jejich podrobný popis jednotlivých prostředí a návod k jejich zprovoznění se nachází v kapitole A.

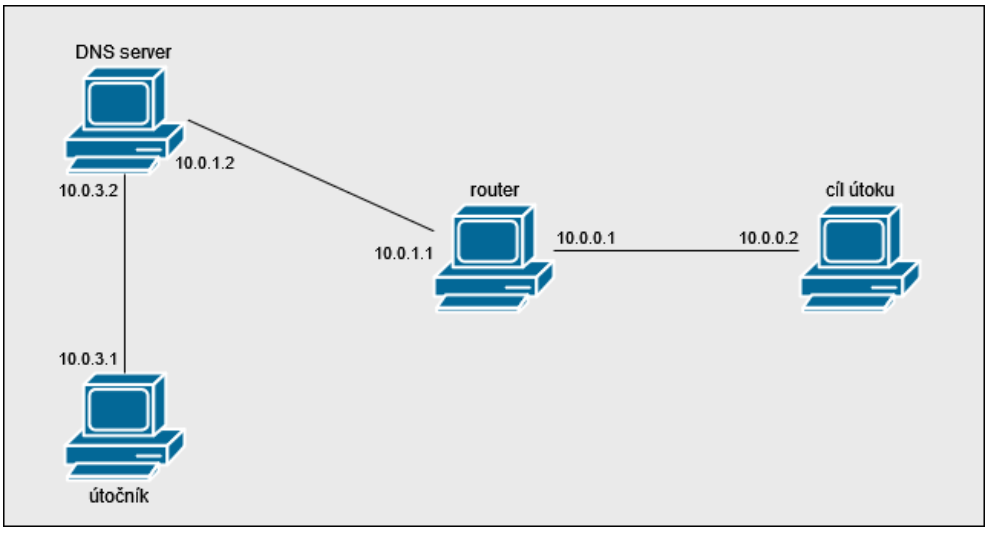
Testované verze program VirtualBox a Vagrant byly 6.1.34, resp. 2.2.19. Jako hostitelský operační systém sloužil Microsoft Windows 10 Home 21H2 v sestavení 19044.1645 běžící na notebooku s procesorem Intel i7-6700HQ a 16GB RAM paměti.

[15] https://www.virtualbox.org/
[16] https://www.vagrantup.com/

5.1 DNS Amplifikace

Schéma demonstračního prostředí pro DNS amplifikační útok je zachycené na obrázku 5.1 a skládá se celkem ze 4 virtuálních počítačů:

- DNS server
- směrovač
- cíl útoku
- útočník



Obrázek 5.1: Diagram testovací prostředí pro amplifikační útok

DNS server

V kapitole věnující se analýze tohoto útoku plnil DNS server roli rekurzivního resolveru. Proto byl vyžadována přístup k internetu alespoň v úvodní fázi, než se výsledky dotazů uložily do vyrovnávací paměti. Bylo také potřeba znát doménu, díky které bylo možné dosáhnout největším amplifikace. Pro demonstrační prostředí byl zvolen jiný přístup, který spočíval ve změně role DNS serveru. Konkrétně zde plní roli autoritativního serveru fiktivní domény s názvem `lab.com`. Zónový soubor s údaji o této doméně byl vytvořen tak, aby byly vraceny odpovědi stejné velikosti a struktury, jako v případě replikace útoku popsané v kapitole 4.1.3. Konfigurační soubory BIND9 se nacházejí v `/etc/bind/`. Na doméně je aktivní DNSSEC a celková velikost odpovědi na dotaz ANY je 4 044 bajtů. Cesta k souboru s údaji o doméně je `/etc/bind/zones/lab.com.zone`. Podepsaný zónový soubor společně s použitými klíči leží ve stejné složce.

Router/NIDS

Směrovač je v demonstrační prostředí pro tento útok zahrnut proto, zároveň plní roli síťového IDS. Jeho umístění na samostatný počítač na rozhraní interní a veřejné sítě bylo zvoleno z toho důvodu, že se nejvíce přibližuje reálnému prostředí.

K instalaci detekčního systému Snort verze 3.1.1 je potřeba relativně velké množství závislostí a program samotný je kompilován ze zdrojových kódů. Během testování na virtuálních strojích byl celý instalační proces velmi pomalý. Aby se předešlo zdlouhavému zprovozňování celého prostředí, byl vytvořen obraz virtuálního stroje, ve kterém je NIDS již předinstalován a nakonfigurován. Lze jej nalézt ve veřejném repozitáři na adrese `https://app.vagrantup.com/suchaja7/boxes/snort` a nástroj Vagrant by ho měl automaticky stáhnout při inicializaci prostředí. Obraz je současně umístěn i na přiloženém DVD.

Snort je nastaven tak, aby se spouštěl automaticky po startu počítače jako služba na pozadí. Konfigurační soubor `snort.lua` se nachází ve složce `/usr/local/etc/snort`. Proměnné `HOME_NET` a `EXTERNAL_NET`, které určují adresu chráněných a externích sítí, jsou nastaveny na 10.0.0.0/24, resp. `[[!$HOME_NET]]`.

IDS je v základu vybaven sadou přibližně 500 základních pravidel, která v demonstračním prostředí deaktivována. Povolit je lze změnou proměnné `enable_builtin_rules` v konfiguračním souboru na `true`. Pravidla, která jsou používána k detekci útoků a byla popsána v předchozích kapitolách, se nacházejí v souboru `/usr/local/etc/snort/rules/local.rules` a jsou aktivní ve výchozím nastavení.

Snort je nastaven ke generování upozornění ve formátu JSON. Výstup se nachází v souboru `/var/log/snort/alert_json`.

Útočník

Program pro provedení útoku se v adresáři `/home/vagrant/`. K jeho běhu jsou nutná administrátorská práva, protože vyžaduje privilegovaný přístup k síťovému rozhraní. Samotný útok lze spustit pomocí příkazu v ukázce 5.1. Parametry programu a jejich význam je následující:

- d – adresa DNS serveru
- s – nepovinný parametr, kterým lze nastavit zdrojovou adresu v odesílaných paketech
- q – DNS dotaz
- t – typ DNS dotazu v dekadické podobě (výchozí hodnota je 255, tedy dotaz ANY)

- l – pakety budou odesílány ve smyčce maximální rychlostí
- h – zobrazí nápovědu

```
sudo /home/vagrant/dns_flood -q lab.com -d 10.0.3.2 \
-s 10.0.0.2 -l
```

Ukázka 5.1: Příkaz pro spuštění amplifikačního DNS útoku v demonstračním prostředí

5.1.1 Demonstrace útoku

Útok byl ve vytvořeném prostředí otestován stejným způsobem, jako v předchozí kapitole. Pomocí nastroje sar byly sbírány udáje o vytížení CPU a síťových rozhraní na všech zúčastněných počítačích. Jediným rozdílem je, že detekční systém běžel na samostatném stroji, který zároveň plnil roli směrovače. Na DNS serveru nebyly aplikovány žádné omezení na počet datazů.

Naměřené hodnoty jsou k videní v tabulce 5.1. Celková zátěž procesorů výrazně vzrostla. Zaroveň došlo k poklesu počtu paketů, které byl stroj plnící úlohu útočníka schopný odesílat. Tyto výsledky jsou v souladu s očekáváním a jejich přícina je přisuzována běhu ve virtualizačním prostředí.

Počítač	CPU	Pkt_{in}/s	Pkt_{out}/s	kB_{in}	kB_{out}
DNS server	87,12 %	21 211,03	4 015,13	1 615,68	5 392,63
Oběť	33,60 %	4 338,73	1,07	5 827,26	0,58
Útočník	50,25 %	0,03	20 994,84	0	1 599,22
Směrovač	72,51 %	4 103,09	4 303,09	5 450,98	5 964,98

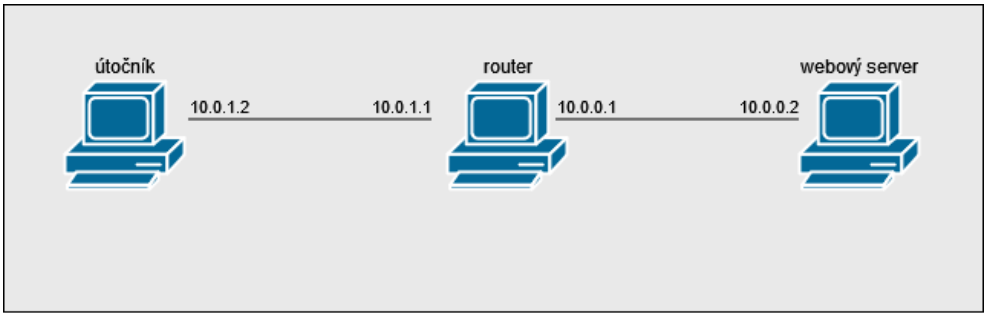
Tabulka 5.1: Výsledky měření zatížení jednotlivých strojů během amplifikačního útoku v demonstračním prostředí

5.2 SYN flood

K demonstraci záplavového SYN útoku bylo vytvořeno prostředí s celkem 3 počítači. Diagram znázorňující propojení jednotlivých virtuálních počítaču společně s jejich IP adresamy je na obrázku 5.2 a skládá se celkem ze 4 virtuálních počítačů:

- DNS server
- směrovač
- cíl útoku
- útočník

Virtuální stroj plnící roli směrovače a detekčního systému je identickým s tím v minulém scénáři. Stejné zůstává jak konfigurace operačního systému, tak nastavení programu Snort.



Obrázek 5.2: Diagram demonstračního prostředí pro TCP SYN záplavový útok

Webový server

Jako cíl útoku slouží počítač s nainstalovaným webovým serverem Apache. Redakční systémem Wordpress a databáze MySQL jsou přednastaveny pomocí shell skriptů vykonaných při spouštění prostředí. Port 8080 na hostitelském počítači je přesměrován na port 80 serveru. Tím je zajištěn přístup k webovým stránkám na virtuálním stroji a může být provedeno měření jejich výkonu pomocí nástroje JMeter.

V redakčním systému je vytvořen výchozí administrátorský účet s přístupovými údaji `admin:password`. Wordpress pro připojení do databáze používá uživatelský účet `wordpress` s heslem `password`.

5.2.1 Demonstrace útoku

Při demonstraci záplavového SYN útoku byl server ponechán ve výchozím nastavení. Hodnoty konfiguračních proměnných byly:

- `net.ipv4.tcp_syncookies` = 1
- `net.ipv4.tcp_max_syn_backlog` = 128
- `net.ipv4.tcp_synack_retries` = 5

Na počítači provádějící útok byl nainstalován program hping3, použitý během analýzy. V demonstračním prostředí byl útok proveden pomocí příkazu v ukázce 5.2.

Naměřené hodnoty zatížení CPU, doby odezvy a počtu přijatých a odeslaných paketů je možné vidět v tabulce 5.2. Oproti situaci kdy server běžel na fyzickém počítači došlo k výraznému nárůstu využití procesoru. To je stejně, jako v minulém případě, důsledek použití virtuálního prostředí. Výsledkem je mnohem větší doba odezvy i její odchylka.

Podobnému problému by se dalo předejít několika způsoby. Jako první se nabízí možnost zvýšit počet přidělených jader pro virtuální stroj s daným serverem. Dále pak je možné upravit testovaní scénář v programu JMeter, který sloužil k měření výkonu webových stránek. Ten během experimentu generoval 20 požadavků za sekundu po dobu 50 vteřin, což v tomto případě zjevně překračuje možnosti serveru.

```
sudo hping3 -S -p 80 --flood --rand-source 10.0.0.2
```

Ukázka 5.2: Příkaz pro spuštění záplavového SYN útoku v demonstračním prostředí

Stav	Doba odezvy [ms] Průměr	Odchylka	Chybovost	CPU	Pkt_{in}/s	Pkt_{out}/s
Bez útoku	17 192,38	6 524,41	0 %	98,97 %	306,00	220,94
Při útoku	26 930,10	18 508,35	0 %	100 %	24 095,94	23 648,13

Tabulka 5.2: Měření výkonu serveru během útoku v demonstračním prostředí

Závěr

Cílem práce bylo zanalyzovat síťové útoky, jejich základní principy a představit možné způsoby obrany a detekce. Na základě této rešerše pak mělo být navrženo prostředí, které by umožnilo vybrané útoky demonstrovat.

První kapitola se věnuje popisu nejznámějším útokům a jejich klasifikaci. Jsou v ní stručně představeny i protokoly rodiny TCP/IP společně s jejich slabinami, kterých útočníci využívají.

Detailní rozbor vybraných útoků je popsán v kapitole 4. Konkrétně se jedná o reflexivní amplifikační DNS útok a záplavový TCP SYN útok. V rámci analýzy první zmíněného útoku došlo k vytvoření přehledu nastavení administrativních DNS serverů nejnavštěvovanějších domén a jejich možného potenciálu pro zneužití k amplifikačnímu útoku. Současně vznikl program v jazyce C, kterým lze daný útok vykonat. Vedlejším výstupem jsou Python skripty, které slouží ke shromažďování dat o DNS serverech a výsledcích jednotlivých dotazů. Skripty mohou být v budoucnu použity k měření amplifikačního faktoru dotazů u jednotlivých domén nebo, po jednoduchých úpravách, ke zjištění dalších charakteristik DNS provozu.

Během analýzy záplavového útoku pomocí SYN segmentů bylo otestováno, jak může konfigurace systému probíhající útok ovlivnit nebo eliminovat. Výsledky provedených měření prokazují, že z nastaveních podrobených testování je nejefektivnější povolení SYN cookies. U ostatních konfiguračních proměnných nebyla pozorována významnější schopnost zmírnit dopady útoku.

Současně byly představeny možné způsoby obrany, které lze aplikovat na straně cíle útoku, a došlo i na ověření jejich efektivity. V závěrečné fázi analýzy byla vytvořena pravidla pro síťový detekční systém Snort, které upozorní administrátory na probíhající útok. Systém byl nasazen na strojích, které se staly cílem útoku, a bylo ověřeno, že pravidla probíhající útoky spolehlivě rozpoznají.

Poznatky získané během analýzy byly využity v poslední kapitole. Jejím výstupem je demonstrační prostředí, které je realizované za pomoci virtualizačního nástroje VirtualBox a programu Vagrant. Obrazy jednotlivých strojů

jsou při spuštění automaticky nakonfigurovány. Taktéž obsahují veškeré potřebné nástroje a skripty k vykonání útoků a ověření základních způsobů jejich detekce.

Uživatelům je tím dána možnost útoky demonstrovat bez ohledu na použitou platformu hostitelského systému. Součástí přílohy práce je obraz virtuálního počítače s přednastaveným programem Snort. Celé prostředí, stejně tak jako jeho jednotlivé součásti, lze například využít během výuky předmětů zabývajících se síťovou bezpečností. Všechny cíle práce tak byly naplněny.

Literatura

[1] Kepios: Digital 2022: Global Overview Report. leden 2022, [online], Navštíveno 27.3.2022. Dostupné z: https://datareportal.com/reports/digital-2022-global-overview-report

[2] Gutnikov, A.; Kupreev, O.; Shmelev, Y.: DDoS attacks in Q4 2021. únor 2022, [online], Navštíveno 10.4.2022. Dostupné z: https://securelist.com/ddos-attacks-in-q4-2021/105784/

[3] Menscher, D.: Exponential growth in DDoS attack volumes. *Google Cloud Blog*, říjen 2020, [online], Navštíveno 12.4.2022. Dostupné z: https://cloud.google.com/blog/products/identity-security/identifying-and-protecting-against-the-largest-ddos-attacks

[4] Amazon: Threat Landscape Report – Q1 2020. *AWS Security Blog*, květen 2020, [online], Navštíveno 12.4.2022. Dostupné z: https://aws-shield-tlr.s3.amazonaws.com/2020-Q1_AWS_Shield_TLR.pdf

[5] Hassan, R.; Mohamed Sid Ahmed, A. S. A.; Othman, N. E.; aj.: Enhanced encapsulated security payload a new mechanism to secure internet protocol version 6 over internet protocol version 4. *Journal of Computer Science*, ročník 10, 03 2014: s. 1344–1354, doi:10.3844/jcssp.2014.1344.1354.

[6] Commons, W.: Layout of a TCP packet. 2004, [online], Navštíveno 12.3.2022. Dostupné z: https://commons.wikimedia.org/wiki/File:TCP_header.png

[7] Kurose, J. F.; Ross, K. W.: Computer Networking: A Top-Down Approach Edition. *Addision Wesley*, 2007.

[8] Divakaran, D. M.; Murthy, H.; Gonsalves, T.: Detection of Syn Flooding Attacks using Linear Prediction Analysis. 10 2006, s. 1 – 6, doi:10.1109/ICON.2006.302563.

[9] Raghavan, S.; Dawson, E.: *An investigation into the detection and mitigation of denial of service (dos) attacks: critical information infrastructure protection*. Springer, 2011.

[10] Shields, C.: What do we mean by Network Denial of Service. In *Proceedings of the 2002 IEEE Workshop on Information Assurance and Security*, ročník 4, 2002.

[11] Mujtiba, S.; Begh, G. R.: Impact of DDoS attack (UDP Flooding) on queuing models. 09 2013, ISBN 978-1-4799-1572-9, s. 210–216, doi: 10.1109/ICCCT.2013.6749629.

[12] Eddy, W.: TCP SYN Flooding Attacks and Common Mitigations. Technická zpráva, RFC Editor, August 2007, [online], Navštíveno 26.3.2022. Dostupné z: `https://datatracker.ietf.org/doc/html/rfc4987`

[13] daemon9; route; infinity: Project Neptune. *Phrack Magazine*, ročník 7, č. 48, červenec 1996: str. 13 – 17, [online], Navštíveno 1.3.2022. Dostupné z: `http://phrack.org/issues/48/13.html`

[14] Postel, J.: Transmission Control Protocol. STD 7, RFC Editor, September 1981, `http://www.rfc-editor.org/rfc/rfc793.txt`. Dostupné z: `http://www.rfc-editor.org/rfc/rfc793.txt`

[15] Bani-Hani, R.; Al-Ali, Z.: SYN flooding attacks and countermeasures: a survey. In *Proceedings of ICICS*, 2013.

[16] Fielding, R. T.; Gettys, J.; Mogul, J. C.; aj.: Hypertext Transfer Protocol – HTTP/1.1. RFC 2616, RFC Editor, June 1999. Dostupné z: `http://www.rfc-editor.org/rfc/rfc2616.txt`

[17] Coleman, G.: Anonymous in context: The politics and power behind the mask. *Centre for International Governance Innovation*, září 2013, [online], Navštíveno 1.3.2022. Dostupné z: `https://www.cigionline.org/sites/default/files/no3_8.pdf`

[18] Postel, J.: Internet Control Message Protocol. Technická zpráva, RFC Editor, September 1981, [online], Navštíveno 26.3.2022. Dostupné z: `http://www.rfc-editor.org/rfc/rfc792.txt`

[19] Braden, R.: Requirements for Internet Hosts - Communication Layers. Technická zpráva, RFC Editor, October 1989, [online], Navštíveno 10.4.2022. Dostupné z: `http://www.rfc-editor.org/rfc/rfc1122.txt`

[20] Linkody: What is a root domain? *Linkody's blog*, May 2020, [online], Navštíveno 1.3.2022. Dostupné z: `https://blog.linkody.com/what-is-a-root-domain`

[21] Aitchison, R.: *Pro DNS and BIND 10*. Apress, první vydání, 2011, ISBN 1430230487; 9781430230489.

[22] Herzberg, A.; Shulman, H.: Negotiating DNSSEC Algorithms over Legacy Proxies. 10 2014, ISBN 978-3-319-12279-3, doi:10.1007/978-3-319-12280-9_8.

[23] Anagnostopoulos, M.; Kambourakis, G.; Kopanos, P.; aj.: DNS amplification attack revisited. *Computers and Security*, ročník 39, 2013: s. 475–485, ISSN 0167-4048, doi:https://doi.org/10.1016/j.cose.2013.10.001.

[24] UDP-Based Amplification Attacks. Technická zpráva, Cybersecurity and Infrastructure Security Agency, leden 2014, [online], Navštíveno 1.3.2022. Dostupné z: `https://www.cisa.gov/uscert/ncas/alerts/TA14-017A`

[25] Brodsky, Z.: The Psychology Behind DDoS: Motivations and Methods. [online], Navštíveno 1.3.2022. Dostupné z: `https://www.perimeter81.com/blog/network/the-psychology-behind-ddos-attacks`

[26] Streda, A.; Kaloc, J.: DDoS hacktivism: A highly risky exercise. březen 2022, [online], Navštíveno 15.3.2022. Dostupné z: `https://blog.avast.com/ddos-hacktivism-avast`

[27] Fabian, M. A. R. J. Z.; Terzis, M. A.: A multifaceted approach to understanding the botnet phenomenon. In *Proceedings of the Internet Measurement Conference*, Citeseer, 2006.

[28] Vormayr, G.; Zseby, T.; Fabini, J.: Botnet communication patterns. *IEEE Communications Surveys & Tutorials*, ročník 19, č. 4, 2017: s. 2768–2796.

[29] Rahimipour, M.; Jamali, S.: A Survey on Botnets and Web-based Botnet Characteristics. *International Journal of Science, Engineering and Computer Technology*, ročník 4, č. 11, 2014: str. 282.

[30] Wack, J.; Cutler, K.; Pole, J.: Guidelines on firewalls and firewall policy. Technická zpráva, NIST, 2002, [online], Navštíveno 5.4.2022. Dostupné z: `https://csrc.nist.gov/publications/detail/sp/800-41/rev-1/final`

[31] Holland, T.: Understanding IPS and IDS: Using IPS and IDS together for Defense in Depth. 2004, [online], Navštíveno 12.3.2022. Dostupné z: `https://www.sans.org/white-papers/1381/`

[32] Scarfone, K.; Mell, P.; aj.: Guide to intrusion detection and prevention systems (idps). *NIST special publication*, ročník 800, č. 2007, 2007: str. 94, [online], Navštíveno 5.3.2022. Dostupné z: `https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-94.pdf`

[33] Beale J., P. J., Foster J.C.: *Snort 2.0 Intrusion Detection*. První vydání, 2003, ISBN 1928994741,1931836744,1931836876.

[34] Rijswijk-Deij, R.; Sperotto, A.; Pras, A.: DNSSEC and its potential for DDoS attacks: A comprehensive measurement study. *Proceedings of the ACM SIGCOMM Internet Measurement Conference, IMC*, 11 2014: s. 449–460, doi:10.1145/2663716.2663731.

[35] Abley, J.; Gudmundsson, O.; Majkowski, M.; aj.: Providing Minimal-Sized Responses to DNS Queries That Have QTYPE=ANY. RFC 8482, RFC Editor, January 2019, [online], Navštíveno 1.3.2022. Dostupné z: `https://datatracker.ietf.org/doc/html/rfc8482`

[36] Ehrenkranz, T.; Li, J.: On the State of IP Spoofing Defense. ročník 9, č. 2, may 2009, ISSN 1533-5399, doi:10.1145/1516539.1516541. Dostupné z: `https://doi.org/10.1145/1516539.1516541`

Příloha A

Návod na použití demonstračního prostředí

Pro zprovoznění je nutné nainstalovat programy VirtualBox a Vagrant, které jsou k dispozici na webových stránkách jejich vývojářů[17,18]. Stránky obsahují i podrobný návod jak jednotlivé programy instalovat na různé operační systémy.

Soubory potřebné pro spuštění prostředí se nacházejí ve složce `env` v příloze práce. Každé prostředí má pak svojí podsložku. Jejich názvy jsou `dns-amp` a `syn-flood`. Každá z nich obsahuje podadresář `vagrant`, ve které se nachází soubor `Vagrantfile` zajišťující základní konfiguraci jednotlivých virtuálních počítačů. Shell skripty s příkazy, které slouží k dodatečnému nastavení po inicializaci virtuálního stroje, leží v adresáři `bootstrap-scripts`. Skripty jsou spouštěny automaticky, bez zásahu uživatele.

Pro spuštění prostředí je potřeba se v terminálu nebo příkazové řádce přepnout do podadresáře `vagrant` ve složce odpovídající scénáři, který chce uživatel spustit. Posléze je nutné vykonat příkaz „`vagrant up`“. Příkazy lze doplnit volitelným argumentem obsahujícím jméno virtuálního počítače. Tím dojde ke spuštění pouze toho konkrétního stroje.

Běh celého prostředí lze ukončit pomocí příkazu „`vagrant halt`“. Pro odstranění všech virtuálních strojů společně s obsahem jejich disku je třeba provést příkaz „`vagrant destroy`“. Po jeho provedení je před opětovným spuštěním demonstračního prostředí nutné znovu zadat příkaz „`vagrant up`“. Všechny zmíněné příkazy jsou společně s příklady užití zobrazeny v ukázce A.1.

Měření zátěže CPU a síťových rozhraní je možné využít nainstalovaného nástroje `sar`. Informace týkající se procesoru jsou zobrazeny pomocí příkazu „`sar -u ALL INTERVAL TRVANI`“. Vytížení síťových rozhraní, počet přijíma-

[17] https://www.virtualbox.org/wiki/Downloads
[18] https://www.vagrantup.com/downloads

ných a odesílaných paketů společně s dalšími údaji lze zobrazit příkazem „`sar -n DEV INTERVAL TRVANI`“. Parametry programu `INTERVAL` a `TRVANI` určují, v jakých intervalech budou naměřené hodnoty zobrazovány a jak dlouhá bude celková doba měření.

```
# Spusteni celeho prostredi
vagrant up

# Spusteni vybraneho stroje
vagrant up dns-server

# Zastaveni prostredi
vagrant halt

# Zastaveni jednoho stroje
vagrant halt dns-server

# Vymazani dat vsech pocitacu v prostredi
vagrant destroy

# Vymazani dat jednoho pocitace
vagrant destroy dns-server
```

Ukázka A.1: Přehled příkazů pro práci s demonstračním prostředím

Manuální přidání Vagrant obrazu

Soubor Vagrantfile obsahuje proměnnou `dns.vm.box`, která určuje, jaký obraz bude daný virtuální stroj používat. Většina počítačů v demonstračním prostředí používá obraz „`ubuntu/bionic64`“.

Jedinou výjimkou jsou počítače plnící roli směrovačů. Pro ně by vytvořen nový obraz, který obsahuje předinstalovaný detekční systém Snort verze 3.1.1. společně s detekčními pravidly. Obraz se nachází ve veřejně dostupném repozitáři na adresa `https://app.vagrantup.com/suchaja7/boxes/snort` a program Vagrant by ho měl automaticky stáhnout.

Pokud by nastaly problémy s automatickým načtením, lze Vagrant instruovat k tomu, aby obraz načetl z lokálního souboru. Ten je umístěn na přiloženém DVD. Konkrétně jde o soubor `suchaja7-snort.box` ve složce `env`. Uživatel musí spustit terminál ve zmíněné složce a vykonat příkaz na ukázce A.2. Tím dojde k registraci obrazu, a poté lze prostředí spustit standardním způsobem.

```
vagrant box add suchaja7/snort suchaja7-snort.box
```

Ukázka A.2: Příkaz pro manuální registraci Vagrant boxu

A.1 DNS amplifikace

Po spuštění prostředí je možné se na jednotlivé stroje připojit pomocí příkazu „`vagrant ssh NAZEV_STROJE`“. Názvy jednotlivých počítačů a jejich role v prostředí jsou vypsány v tabulce A.1.

Název	Role
dns-server	DNS server
dns-router	směrovač a detekční systém
dns-target	cíl útoku
dns-attacker	útočník

Tabulka A.1: Názvy a role jednotlivých počítačů v demonstračním prostředí pro amplifikační DNS útok

Útok lze provést po připojení k počítači s názvem `dns-attacker` a spuštění programu `dns_flood` podle ukázky A.3. Program se nachází v domovském adresáři uživatele vagrant společně se složkou obsahující jeho zdrojové kódy v jazyce C. Zdrojový soubor lze v případě nutnosti zkompilovat pomocí příkazu „`gcc dns_flood.c -lnet`“.

```
sudo /home/vagrant/dns_flood -q lab.com -d 10.0.3.2 \
-s 10.0.0.2 -l
```

Ukázka A.3: Příkaz pro spuštění útoku

K zobrazení varování generovaných detekčním systémem Snort je nutné se připojit ke stroji `dns-router`. Varování jsou zapisována do souboru `/var/log/snort/alert_json`.

Snort běží jako služba na pozadí a není potřeba jej nijak aktivovat. Soubory vztahují se ke konfiguraci detekčního systému jsou:

- `/usr/local/etc/snort/snort.lua` – konfigurační soubor Snort
- `/usr/local/etc/rules/local.rules` – soubor s pravidly
- `/usr/local/etc/lists/default.blacklist` – soubor s blokovanými IP adresami
- `/var/log/snort/alert_json` – výstup programu

- `/lib/systemd/system/snort3.service` – systemD skript pro spouštění služby na pozadí

V případě potřeby je možné detekční systém dočasně nebo trvale deaktivovat příkazem „`sudo systemctl stop snort3.service`“, resp. „`sudo systemctl disable snort3.service`“. Pokud dojde k úpravě souboru s pravidly nebo blokovanými adresami, je nutné službu restartovat přikazem „`sudo systemctl restart snort3.service`“.

DNS server slouží jako administrativní pro doménu `lab.com`, která používá DNSSEC. Podepsaný zónový soubor společně s použitými klíči se nachází ve složce `/etc/bind/zones` na stroji `dns-server`. O vygenerování klíčů a podepsaní domény se stará shell skript `dns-setup.sh`.

A.2 SYN flood

Postup pro spuštění demonstračního prostředí s tímto útokem je identický jako v předchozím případě. V tomto scénáři jsou zahrnuty celkem 3 virtuální stroje, jejichž názvy a role jsou uvedeny v tabulce A.2.

Název	Role
tcp-target	Webový server
tcp-router	směrovač a detekční systém
tcp-attacker	útočník

Tabulka A.2: Názvy a role jednotlivých počítačů v demonstračním prostředí pro záplavovým TCP SYN útok

Cílem útoku je počítač s webovým serverem Apache. Na něm běží redakční systémem Wordpress používající databázi MySQL. Při inicializaci prostředí je databáze naplněna daty ze souboru `db_init.sql` ve složce `config`. Redakční systém se do databáze připojuje pomocí uživatelského účtu `wordpress` s heslem `password`. K tomuto účtu se jde připojit pouze lokálně, tzn. z localhost.

Program Vagrant mapuje port 8080 na hostitelském počítači na port 80 webového serveru. Tím je ke stránkám umožněn přístup skrze adresu `http://localhost:8080`. Pro testování výkonu webu je možné použít nástroj JMeter uvedený v předchozí části práce. Ve složce `tcp_flood/vagrant/config` se nachází soubor `Test Plan.xml`, který obsahuje přednastavený testovací scénář. Po jeho naimportování do programu JMeter, je možné test spustit. Výsledky mohou být zobrazeny v GUI a případně exportovány ve formátu CSV.

Ke spuštění útoku je nutné se připojit na počítač `tcp-attacker` a spustit příkaz v ukázce A.4.

```
sudo hping3 -S -p 80 --flood --rand-source 10.0.0.2
```

Ukázka A.4: Příkaz pro spuštění záplavového SYN útoku v demonstračním prostředí

Příloha B

Seznam použitých zkratek

ARP – Address Resolution Protocol

BGP – Border Gateway Protocol

C&C – Command-and-Control

ccTLD – Country Code Top-Level Domain

DDoS – Distributed Denial-of-Service

DNS – Domain Name System

DoS – Denial-of-Service

EDSN – Extension Mechanisms for DNS

GRE – Generic Routing Encapsulation

gTLD – Generic Top-Level Domain

HTML – Hypertext Markup Language

HTTP – Hypertext Transfer Protocol

ICANN – Internet Corporation for Assigned Names and Numbers

ICMP – Internet Control Message Protocol

IP – Internet Protocol

IRC – Internet Relay Chat

ISP – Internet Service Provider

MPLS – Multiprotocol Label Switching

P2P – Peer-to-Peer

SMB – Server Message Block

TCB – Transmission Control Block

TCP – Transmission Control Protocol

TLD – Top-Level Domain

TTL – Time to Live

UDP – User Datagram Protocol

URI – Uniform Resoure Identifier

WWW – World Wide Web

Příloha C

Obsah přiloženého DVD

- `thesis` ..... text a zdrojové soubory práce
  - `src` ..... zdrojové kódy práce ve formátu LaTeX
  - `thesis.pdf` ..... text práce ve formátu PDF
- `env` ..... adresář se zdrojovými kódy prostředí
  - `suchaja7-snort.box` ..... obraz stroje s NIDS Snort
  - `dns_amplification`
    - `data-collection` ..... skripty pro sběr dat o DNS serverech
    - `vagrant`
      - `Vagrantfile` ..... konfigurační soubor prostředí
      - `bootstrap-scripts` ..... shell skripty s nastavením
      - `config` ..... doplňující konfigurační soubory
      - `src` ..... zdrojové kódy pro demonstraci útoku
  - `syn_flood`
    - `vagrant`
      - `Vagrantfile` ..... konfigurační soubor prostředí
      - `bootstrap-scripts` ..... shell skripty s nastavením
      - `config` ..... doplňující konfigurační soubory