VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ
DEPARTMENT OF COMPUTER SYSTEMS

AKTIVNÍ REAKCE NA VYBRANÉ TYPY SÍŤOVÝCH ÚTOKŮ

ACTIVE RESPONSE TO SELECTED NETWORK ATTACKS

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE **JAROMÍR WYSOGLAD**
AUTHOR

VEDOUCÍ PRÁCE **Ing. MARTIN ŽÁDNÍK, Ph.D.**
SUPERVISOR

BRNO 2023

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Zadání diplomové práce

146257

Ústav:	Ústav počítačových systémů (UPSY)
Student:	**Wysoglad Jaromír, Bc.**
Program:	Informační technologie a umělá inteligence
Specializace:	Kybernetická bezpečnost
Název:	**Aktivní reakce na vybrané typy síťových útoků**
Kategorie:	Počítačové sítě
Akademický rok:	2022/23

Zadání:

1. Nastudujte oblast útoků na počítačové systémy, které jsou vedeny po síti. Pro členění využijte vhodnou taxonomii útoků. Zaměřte se na konkrétní případy útoků, kdy proti útokům byla vedena aktivní obrana, tj. nikoliv pouze pasivní zablokování příchozího síťového provozu.
2. Seznamte se s bezpečnostními nástroji používanými v síti CESNET.
3. Vyberte aktivní reakce, které by bylo možné automatizovat, a navrhněte jejich implementaci do prostředí sítě CESNET.
4. Vybrané reakce na útoky implementujte.
5. Po dohodě s vedoucím implementaci ověřte buď v laboratorním či reálném síťovém prostředí.
6. Zhodnoťte dosažené výsledky a diskutujte možnosti pokračování projektu.

Literatura:

- Dle pokynů vedoucího.

Při obhajobě semestrální části projektu je požadováno:

- Splnění bodů 1 až 3 zadání.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce:	**Žádník Martin, Ing., Ph.D.**
Vedoucí ústavu:	Sekanina Lukáš, prof. Ing., Ph.D.
Datum zadání:	1.11.2022
Termín pro odevzdání:	17.5.2023
Datum schválení:	31.10.2022

Abstrakt

Cílem této práce je navrhnout a implementovat aktivní reakci na zvolený síťový útok. V práci je vyjmenováno několik různých reakcí na různé útoky vedené po síti. Tyto reakce jsou následně porovnány pomocí několika různých kritérií a je z nich vybrána nejvhodnější reakce pro následnou implementaci. Tato reakce je nejdříve detailně popsána a je detailně navržen návrh její implementace. V pozdějších fázích této práce je zvolená reakce implementována a otestována. Zvolená reakce dokáže reagovat na dva druhy útoků. Prvním z nich je skenování portů, kdy implementovaný projekt dokáže na sken libovolně odpovídat místo původně zamýšlené oběti. Druhým druhem útoku je situace, kdy se útočník pokouší přihlásit na SSH server oběti. Implementovaný projekt dokáže tyto pokusy přesměrovat na honeypot, který zaznamenává použité přihlašovací údaje. Dále je možnost útočníka nechat úspěšně přihlásit. V této situaci honeypot útočníka přesměruje do docker kontejneru, takže si útočník bude myslet, že je na skutečném SSH serveru. Honeypot následně zaznamenává veškerý útočníkův pohyb v kontejneru. V rámci práce byla také vytvořena jednoduchá aplikace umožňující přehrát takto zaznamenaný útok tak, jak jej viděl a provedl útočník.

Abstract

The goal of this thesis is to design and implement an active response to a chosen network attack. In the thesis are mentioned a few different possible responses to network attacks. These reactions are then compared using a few different criteria and the most appropriate response is then chosen for implementation. This response is then described in detail and its implementation is proposed. In the later fazes of the thesis the reaction is implemented and tested. The chosen reaction can react to two types of attacks. The first type is a port scan. The implemented project can answer a port scan instead of the original victim. The second type is a situation, when an attacker is trying to log into an SSH server of the victim. The project can reroute these login attempts to a honeypot, which can record the used login credentials. After this, it's possible to let the attacker to successfuly login. In this sitation the honeypot will reroute the attacker to a docker container, so the attacker will think, that this is a real ssh server. The honeypot will then record attacker's every move on the container. As a part of the thesis, there is also a simple application, which allows the defender to replay the recorded attack. The defender will se exactly the same output as the attacker saw during the attack.

Klíčová slova

počítačová bezpečnost, aktivní reakce, obrana, počítačová síť, honeypot, skenování portů

Keywords

cyber security, active response, defense, computer network, honeypot, port scan

Citace

WYSOGLAD, Jaromír. *Aktivní reakce na vybrané typy síťových útoků.* Brno, 2023. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Martin Žádník, Ph.D.

Aktivní reakce na vybrané typy síťových útoků

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením pana Ing. Martina Žádníka Ph.D. Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
Jaromír Wysoglad
16. května 2023

Poděkování

Děkuji Ing. Martinu Žádníkovi Ph.D. za vedení mé diplomové práce a mým rodičům za konzultaci jazykové stránky práce.

Obsah

Kapitola 1

Úvod

Internet a počítačové sítě obecně jsou v dnešní době nedílnou součástí života většiny lidí na světě. Spousta lidí používá internet denně, je zdrojem informací, zábavy, ale také zdrojem nebezpečí. Všichni se již od malička učíme, že v reálném světě může na člověka číhat spousta nebezpečí. V temných uličkách, ale také na přímém světle se mohou skrývat zlí lidé, snažící se ostatní lidi okrást, podvést, nebo jim jinak ublížit. Podobně však funguje také svět internetu, kde se na každém kroku může skrývat hacker, snažící se lidi podvést a různými způsoby z nich, nebo z nezabezpečených zařízení vymámit informace, které lze následně zneužít. Takovou informací může být téměř cokoliv. Může se jednat i o něco tak neškodného, jako je emailová adresa, kterou často sami zveřejňujeme a která se pak může stát terčem nevyžádaných e-mailů obsahujících další potencionálně nebezpečný obsah. Mezi další zneužitelné informace patří samozřejmě také přístupové údaje k jakémukoliv účtu, adresa bydliště, telefonní číslo, atd. Potencionální zneužití těchto údajů si jistě každý dokáže představit.

Často tato informace nemusí být hackerovi poskytnuta člověkem. Stačí pouze, aby hacker objevil zranitelnost v systému, na kterém je informace uložena, nebo přímo v systému k jehož přístupu jsou tyto tajné údaje potřeba. Hacker může zranitelnosti snadno využít a získat přístup k takto zranitelnému systému, včetně na něm uložených informací. Následně může takový systém často bez vědomí jeho majitele dále monitorovat a sbírat nově zadané informace, případně jej využít pro další útoky.

Proto je potřeba se proti útokům bránit. Je spousta způsobů obrany proti útokům, které by šlo označit za pasivní, tedy že obránce takové obranné řešení nastaví, zapne a obrana probíhá stále, bez zásahu obránce a při útoku obránce často ani neví, že útok probíhá. Mezi takovou obranu může patřit například firewall, který je přítomen nonstop a nonstop chrání proti pokusům útočníků o průnik do zařízení, aniž by to uživatel přímo věděl. Další takovou obranou může být antivirus, který může potencionálně nebezpečnou aplikaci před jejím spuštěním oskenovat a varovat uživaletele.

V této práci nás však zajímá aktivní obrana. Tedy situace, kdy jako obránce odhalím probíhající útok. Jaké jsou mé možnosti? Možností, která se nabízí jako první je například zablokování síťového provozu přicházejícího od útočníka. Tato obrana však často nemusí být zcela účinná. Pro průměrného útočníka není žádný problém útok vést odjinud a i kdyby to problém byl, internet je velký a útočník si najde jinou oběť, která jej odhalit nemusí. Možnosti aktivní obrany, kterými se zabývá tato práce si berou za úkol útočníka zpomalit, zmást, nebo jednoduše naštvat. Jednou z účinných implementací takové obrany je projekt LaBrea 2.3.3, který je schopen značně zpomalit probíhající útoky tak, že na útočníkovy dotazy odpovídá co nejpomaleji.

Kapitola 2

Současný stav

V této kapitole budou uvedeny vybrané metodologie, nástroje a techniky z oblasti sítí a bezpečnosti, které souvisí s touto prací. Jako první bude uveden rozbor jednotlivých fází útoku (kapitola 2.1). Poté se čtenář seznámí se současným stavem a používanými nástroji jak z pohledu obránce (kapitola 2.3), tak z pohledu útočníka (kapitola 2.4) a nakonec budou představeny bezpečnostní nástroje využívané v síti CESNET v kapitole 2.6.

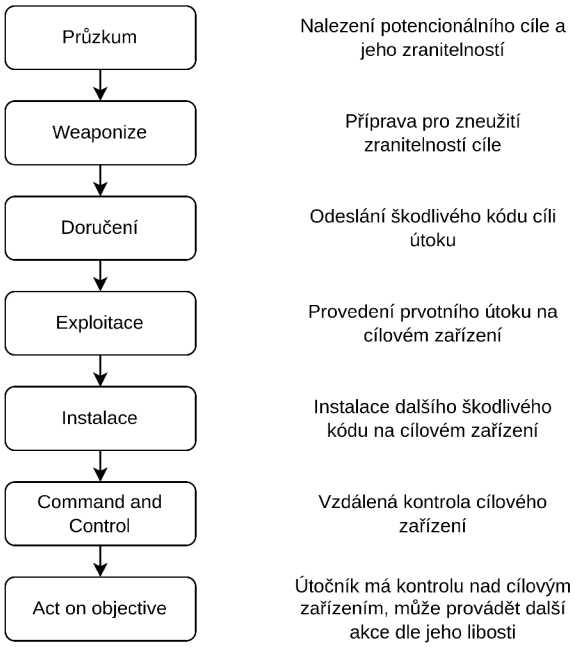
2.1 Fáze útoku

Počítačové útoky jsou staré skoro stejně jako počítače samy. V průběhu let si lidé všimli jistých podobností mezi jednotlivými útoky, většinu útoků lze rozdělit na několik stejných fází. Toto rozdělení podle fází nazýváme Cyber Killchain a lze jej poté použít pro popis jak daleko se daný útok nachází. Další možností, jak na fáze útoku nahlížet je MITRE ATT&CK, který popisuje jednotlivé fáze útoku včetně návrhů na jejich detekci a možností obrany.

2.1.1 Killchain

Cyber Killchain [20] je model pro incident response týmy, digitální forensní vyšetřovatele a malware analysty, sloužící pro popis fází útoku a umožňující analýzu útoku podle jeho jednotlivých kroků. Ve zbytku sekce budou popsány jednotlivé fáze a bude krátce vysvětlen jejich význam. Grafické znázornění tohoto modelu lze vidět na obrázku 2.1

1. **Reconnaissance - Průzkum**: Při průzkumu útočník sbírá informace o potenciálním cíli. Průzkum v kyber prostoru většinou zahrnuje automatické procházení internetu, tedy webových stránek, blogů, konferencí, mailing listů a také skenování sítě pomocí nástrojů jako je nmap 2.4.2 ve snaze získat co nejvíce informací o potenciálním cíli. Sesbírané informace jsou využity při pozdějších fázích útoku například pro vytvoření a doručení payloadu. [20] Průzkum lze rozdělit do dvou kategorií.
   (a) **Pasivní průzkum** Při tomto typu průzkumu útočník sbírá informace pasivně a nelze jej nijak odhalit.
   (b) **Aktivní průzkum** Při tomto typu průzkumu útočník sbírá informace aktivně, dokáže zjistit mnohem více informací, avšak používá nástroje, jako například nmap, u kterých riskuje odhalení.



Obrázek 2.1: **Killchain** Na obrázku si lze prohlédnout jednotlivé fáze killchainu s krátkým popiskem každé z nich. Obrázek převzat z [20]

2. **Weaponize - Vyzbrojení**: V této fázi útoku má útočník již vybraný cíl útoku a má o něm dostatek informací. Útočník tedy podle objevených zranitelností oběti navrhne a sestrojí něco, typicky malware, co využije daných zranitelností a provede útočníkem požadované akce. Jednoduchým příkladem takového malware může být skript, který po otevření uživatelem na cílovém systému otevře shell čekající na útočníkovy příkazy.

3. **Delivery - Doručení**: Po vytvoření malware jej útočník musí nějak doručit na cílový systém. Příkladem doručení malware je jeho stažení a spuštění samotným uživatelem. V některých případech může útočník zneužít některé nezabezpečené síťové služby a zařízení.

4. **Exploitation - Využití**: V momentě, kdy se útočníkovi podaří svůj malware úspěšně dostat k oběti a spustit, přichází na řadu fáze exploitace. V této fázi se provádí nějaký útočníkem připravený škodlivý kód, který využívá objevené zranitelnosti. Zranitelnosti mohou být chyby v software, nebo třeba špatná konfigurace zařízení uživatelem. Objeveným bezpečnostním chybám v software jsou přidělovány identifikační čísla a lze je najít v databázi CVE 2.2.

5. **Installation - Instalace**: Při instalaci se využívá zranitelností z předchozí fáze pro instalaci dalšího malware. Útočník v tomto kroku může například stáhnout malware z internetu a nastavit jej tak, aby se automaticky spouštěl při každém spuštění počítače.

6. **Command and Control - Vzdálené ovládání**: Jedná se o důležitou část vzdáleně prováděných útoků. Systém pro vzdálené ovládání může útočník použít pro ovládání napadeného zařízení. Díky tomu je útočník schopen z napadeného zařízení například získávat citlivá data, nebo napadat další zařízení.

7. **Act on Objective - Konečný cíl**: V poslední fázi útočník získal kontrolu nad zařízením a záleží pouze na útočníkovi a cíli jeho útoku, jak bude dále pokračovat. Oběť může přijít o citlivá data, stát se součástí botnetu, nebo se útočník může pokusit o poškození dat i hardware.

2.1.2 MITRE ATT&CK

MITRE ATT&CK je globálně přístupná báze znalostí o taktikách a technikách útočníků založená na pozorováních z reálného světa. Lze ji použít jako základ pro vývoj modelů hrozeb a metodologií. [5]

Na webových stránkách MITRE[1] lze tuto bázi znalostí nalézt jako tabulku technik setříděnou podle čtrnácti kategorií podobných Killchainu z předchozí sekce 2.1.

Každá technika obsahuje krátký popis, příklady jejího použití, jak použití této techniky detekovat a jak se proti ní bránit.

2.2 Zranitelnosti

Zranitelnosti jsou chyby v informačním systému, které může útočník využít k útoku. Známé zranitelnosti jsou shromažďovány například v databázi CVE (Common Vulnerabilities and Exposures).

CVE je program, který si dává za úkol identifikovat, definovat a katalogizovat veřejně známé kyberbezpečnostní zranitelnosti. V CVE katalogu je vždy jeden záznam pro každou zranitelnost. Zranitelnosti jsou objevovány a zveřejňovány partnerskými společnostmi po celém světě. Profesionálové z oboru informačních technologií a kyberbezpečnosti používají CVE záznamy, aby měli jistotu, že se baví o stejném problému a také pro koordinaci a prioritizaci při odstraňování těchto zranitelností. [4]

Tato databáze je však užitečná i pro útočníky. V databázi lze vyhledávat pomocí klíčových slov. U každé zranitelnosti je uveden její krátký popis, který často obsahuje i verze software, které zranitelnost obsahují. Za popisem následují také reference, které mohou útočníkovi poskytnout dostatek informací pro zneužití dané zranitelnosti. Útočník je tedy schopen provést sken zařízení oběti a tím zjistit některé informace o software, nacházejícím se na zařízení, včetně jeho verzí. Následně si útočník může v CVE databázi pomocí klíčových slov vyhledat všechny známé zranitelnosti pro daný software a s pomocí popisků a referencí tyto zranitelnosti může následně využít.

2.3 Nástroje používané obráncem

V této sekci budou popsány některé nástroje a techniky, které může obránce využít pro detekci a obranu proti síťovému útoku. Některé nástroje specifické pro síť CESNETu jsou popsány v sekci 2.6.

2.3.1 Flowspec

Flowspec [7] je mechanismus, který umožňuje na základě až dvanácti kritérií specifikovat síťový tok a následně s ním provádět určité akce. Pro svou funkci flowspec využívá protokol BGP [21].

Kritéria, která je možné specifikovat:

[1] https://attack.mitre.org/#

- Prefix cílové adresy
- Prefix zdrojové adresy
- IP protokol
- Zdrojový nebo cílový port
- Cílový port
- Zdrojový port
- ICMP typ
- ICMP kód
- TCP příznaky (TCP flagy)
- Délka packetu
- DSCP
- Zda se jedná o fragment, případně o který fragment v pořadí

Specifikace toku je tedy n-tice, která se skládá z několika kritérií uvedených výše. Pokud se na směrovači objeví packet, který odpovídá všem specifikovaným kritériím, tak je provedena specifikovaná akce. Výčet všech akcí, které je možno provádět lze nalézt níže.

Akce, které je možno provádět (různé implementace mohou poskytovat další akce navíc):

- Omezení šířky pásma pro daný tok
- Vzorkování a logování toku
- Přesměrování
- Modifikace DSCP bitů

V síti CESNET se používají nástroje ExaFS[2] a ExaBGP[3]. ExaBGP je nástroj pro implementaci SDN sítí nad protokolem BGP a umožňuje také použití Flowspec. ExaFS je nástroj vyvíjený CESNETem a rozšiřuje ExaBGP o autorizaci uživatelů, validaci BGP příkazů, webové rozhraní, REST API a další.

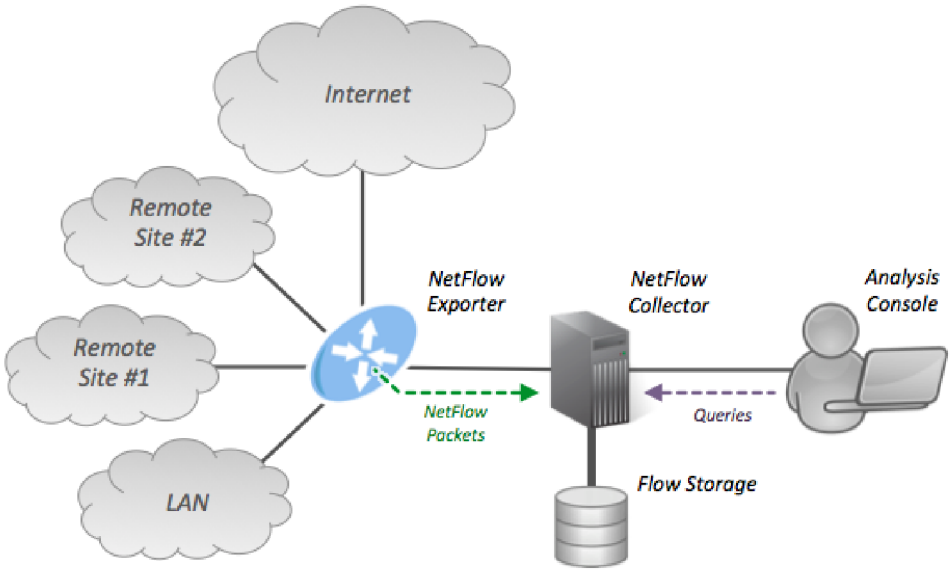
2.3.2 Netflow

Netflow [17] je technologie, která umožňuje administrátorům sbírat data o tocích v jejich síti. Síťová zařízení sbírají data o síťových tocích a exportují je do kolektorů, jak si lze prohlédnout na obrázku 2.2. Tato sesbíraná data lze poté použít k různým účelům. Sesbíraná data o tocích obsahují různé informace, jako např.: IP adresy, velikost a počet paketů, typ služby, čísla portů, ...

V současnosti jsou nejpoužívanější 3 různé verze netflow a to verze 5, která se však jeví jako limitující a přechází se na verzi 9. Zároveň se také usilovně pracuje na definici protokolu na přenos záznamů o tocích IPFIX, který vychází z netflow a měl by jej nahradit.

V síti CESNET se pro monitorování sítě využívá prototypová implementace protokolu IPFIX. Toky jsou automaticky analyzovány a v případě detekování útoku jsou generována upozornění.

[2] https://github.com/CESNET/exafs
[3] https://github.com/Exa-Networks/exabgp



Obrázek 2.2: **Architektura netflow** Na obrázku lze vidět architekturu netflow. Síťové zařízení, v tomto případě směrovač posílá data o tocích netflow kolektoru, který tato data ukládá. Uživatel se poté může na tato data dotazovat a provádět analýzy. [5]

2.3.3 Honeypoty

Zajímavou metodou obrany, která se blíží aktivní obraně, jsou tzv. honeypoty [2]. Většinou se jedná o zařízení na síti, která existují pouze za jediným účelem a to nalákat útočníka, aby na ně zaútočil.

Honeypoty mohou při obraně plnit více účelů. Jedním z nich může být útočníka co nejvíce zdržet, například mu podstrčit spoustu lživých dat, která však vypadají uvěřitelně. Dalším využitím může být sledování chování útočníka. Útočník, který nemá tušení, že se nachází na honeypotu a že je sledován, může při útoku využívat některé doposud neznámé postupy a bezpečnostní chyby. Díky tomu může obránce tyto chyby včas objevit a opravit. Honeypot může obránci také pomoci odhalit doposud skrytý útok tím, že honeypotem bude zařízení, se kterým by za normálních okolností neměla probíhat žádná interakce. V momentě, kdy se útočník pokusí na honeypot zaútočit, tak se obránce dozví, že se něco děje a může začít pracovat na obraně.

Důležitým příkladem honeypotu pro tuto práci je projekt LaBrea [15]. Jedná se o honeypot, který poslouchá síťový provoz a pokud zachytí několik ARP požadavků po sobě, na které nepřijde žádná odpověď, tak LaBrea usoudí, že dochází ke skenování sítě a jelikož na dotazy nepřichází odpovědi, lze předpokládat, že daná IP adresa není přidělena žádnému zařízení. LaBrea následně odpoví na tento ARP požadavek takovým způsobem, že je síťový provoz pro tuto IP adresu směrován k ní. Pokud se poté útočník pokouší LaBreu skenovat, tak LaBrea dělá vše proto, aby odpověď na skenování trvala pokud možno co nejdéle. LaBrea si tedy může zabrat velké rozsahy sítě a značně tak zpomalit útočníkovo skenování sítě.

2.3.4 Suricata

Suricata je open source nástroj, který slouží pro analýzu komunikace na síti a detekci hrozeb. Díky velké základně vývojářů zůstává jednou z nejlepších technologií pro dotekci a reakci na hrozby. [14]

Suricata může zaznamenávat HTTP požadavky, TLS certifikáty, může extrahovat soubory z toků dat a ukládat je na disk. Zvládá také zachytit a uložit celou komunikaci do souboru ve formátu pcap.

Nástroj může být spuštěn ve dvou různých módech. V IDS (intrusion detection system) módu sleduje provoz na síti a v případě detekce hrozby je schopna o hrozbě generovat oznámení. V IPS módu (intrusion prevention system) dokáže sama reagovat na hrozby. Typickou reakcí na hrozbu je ukončení spojení.

2.4 Nástroje používané útočníkem

V předchozí sekci jsem popsal dostupné nástroje na straně obránce. Pro účinnou obranu proti útoku je však nutné znát také možnosti útočníka, protože jak napsal jeden slavný čínský vojevůdce Sun-c': „Pokud znáte nepřítele a znáte sami sebe, nemusíte se bát výsledku sta bitev." [16] Proto v této sekci popíšu některé často používané nástroje útočníků, s tím že se zaměřím na nástroje, jejichž použití lze detekovat na síti.

2.4.1 Kali linux

Než začnu s popisem jednotlivých nástrojů, nelze se nezmínit o některých linuxových distribucích, které poskytují jednoduchý a snadný přístup ke všem dále zmíněným nástrojům. Tou nejznámější je pravděpodobně Kali Linux. Jedná se o open-source linuxovou distribuci založenou na Debianu, která se zaměřuje na poskytnutí nástrojů pro různé kyberbezpečnostní úkoly, jako je například penetrační testování, reverzní inženýrství nebo výzkum v oblasti kyberbezpečnosti [6]. Tato distribuce a její nástroje je však snadno zneužitelná různými útočníky.

Kali obsahuje různé známé nástroje pro různé úkoly, mnohé z nich budou dále popsány v následujících sekcích, patří mezi ně například Metasploit, Burp Suite, Hydra, nmap, John the Ripper, atd. Kromě Kali existují i další podobné distribuce jako BlackArch, Backbox nebo Parrot Security OS.

2.4.2 Nmap

V prvotní fázi útoku je většinou potřeba provést důkladný sken sítě oběti. Cílem skenu je získání co nejvíce informací o síti a o zařízeních k ní připojených. Rozlišujeme dva základní typy skenu. Vertikální sken, kdy si útočník vybere jedno zařízení a postupně skenuje všechny jeho porty s cílem zjistit na kterých portech se nachází která služba. V případě nalezení některé služby lze použitím skenovacího nástroje zjistit o službě další informace, jako například její verzi a následně si útočník může dohledat, zda pro tuto konkrétní službu a verzi není známá nějaká zranitelnost 2.2. Dalším typem skenu je horizontální sken. V tomto případě se útočník zaměřuje na konkrétní službu a skenuje stejný port na všech zařízeních na síti, protože má pravděpodobně už dopředu vyhlídnutou zranitelnost, kterou by chtěl použít. Kromě odhalení služeb nacházejících se na zařízení, skenovací nástroje dokážou také uhádnout použitý operační systém oběti včetně jeho verze, nebo typ zařízení.

Nejčastěji používaným nástrojem pro skenování je nmap [3]. Jedná se o opensource nástroj pro skenování sítí. Nmap postupně posílá packety na specifikované IP adresy a porty tak, aby zjistil která zařízení jsou připojena k síti, které služby jsou na daných zařízeních, včetně jejich jména a verze, který operační systém a jakou verzi zařízení používá, jaké typy filtrů a firewallů jsou na síti použity a spoustu dalšího.

Dalším podobným nástrojem je například masscan [9]. Je velice podobný nmapu, ale používá se pro sken rozsáhlých sítí. Podle jeho vývojářů je schopen oskenovat celý internet za méně než 5 minut.

2.4.3 Hydra

Pokud útočník na zařízení oběti objeví službu, kterou chce využít k útoku, má několik možností. Pokud objevená služba obsahuje nějakou zranitelnost 2.2, tak se útočník může rozhodnout danou zranitelnost zneužít. Příkladem takové zranitelnosti by mohlo být třeba buffer overflow, nebo sql injection při zadávání přihlašovacích údajů, díky kterému lze přihlašování například zcela přeskočit. Nejlepší obranou v takovém případě je udržovat takové služby aktuální.

Další možností útoku je útok silou. Cílem tohoto útoku je většinou pokus o uhádnutí přihlašovacích údajů pro nalezené služby, avšak to nemusí být jediný cíl. Dalším cílem podobného útoku může být třeba snaha zmapovat strukturu webového serveru, atd. Pro útok lze zvolit jednu z následujících strategií, nebo jejich kombinaci. A to buď použít předpřipravený seznam údajů a tak provést slovníkový útok, nebo postupně, systematicky generovat všechny možné kombinace znaků.

Jedním z používaných nástrojů je Hydra [18]. Hydra je nástroj, který lze použít pro hádání přihlašovacích údajů. Jako zdroj údajů lze použít předpřipravený soubor, nebo údaje postupně generovat. Nástroj podporuje desítky síťových protokolů. Při spuštění si uživatel definuje protokol, který pro přihlášení použít, formát v jakém údaje odesílat, zdroj přihlašovacích údajů a kladnou, nebo zápornou odpověď serveru při pokusu o přihlášení.

2.5 Techniky a taktiky útočníka

V této sekci budou popsány všechny techniky a taktiky zmíněné v MITRE ATT&CK 2.1.2, jejichž použití lze detekovat na síti a zároveň u nich existuje šance pro provedení aktivní reakce. Jedná se tedy o síťové útoky, na něž může obránce reagovat i jinak než pouhým zablokováním síťového provozu.

- **Aktivní skenování**[6]: Při aktivním skenování se útočník snaží získat informace o síti a o zařízeních na síti. Pokusem o aktivní reakci by mohlo být odesílání nepravdivých odpovědí na pokusy o sken, díky čemuž lze útočníka značně zdržet, nebo skrýt některé zranitelnosti.

- **Využití existujícího protokolu pro skrytou komunikaci s napadeným zařízením**[7][8]: Jedná se o techniku, kdy se útočník snaží ovládat již dříve napadené zařízení tak, že mu předává příkazy pomocí legitimního aplikačního (http, ssh, ...), nebo neaplikačního (ICMP, UDP) protokolu, aby komunikaci skryl. Pokud se obránci

[6] https://attack.mitre.org/techniques/T1595/
[7] https://attack.mitre.org/techniques/T1071/
[8] https://attack.mitre.org/techniques/T1095/

podaří komunikaci odhalit, tak ji může sledovat. To může obránci umožnit předvídat nadcházející útoky, nenápadně útočníkovi podstrkávat nepravdivá data a tak využít probíhajícího útoku proti útočníkovi.

- **Hádání přihlašovacích údajů hrubou silou**[9]: Při této technice se útočník snaží uhádnout nějaké přihlašovací údaje. Hádání může probíhat offline, například v případě předchozího získání hashů hesel, nebo online, například opakovanými pokusy o přihlášení k SSH serveru. Obránce může probíhající útok sledovat a zaznamenávat slovník přihlašovacích údajů, které útočník zkouší. Další možností by bylo využití honeypotu 2.3.3, který útočníka nakonec nechá úspěšně se přihlásit. Následně může útočník sledovat útočníkovo další počínání.

- **Získávání dat z konfiguračních repozitářů**[10]: Útočník se pokouší získat informace o zařízeních na síti z různých repozitářů obsahujících jejich konfiguraci. Pokud obránce útok odhalí, může útočníkovi místo reálných informací poskytnout falešné informace a tím útočníka zmást a zpomalit.

- **Získávání dat ze síťových disků**[11]: U této techniky se útočník snaží získat data ze sdílených síťových disků. Obránce může využít honeypotů 2.3.3, kdy honeypot bude sdílet disk s falešnými daty. Jelikož na takový disk při normálním provozu nikdo nepřistupuje, tak nejenom že útočník získá falešná, neužitečná data, ale také se touto akcí může prozradit. Tento postup lze tedy využít také pro detekci probíhajícího útoku.

- **Využití aplikace přístupné z internetu**[12]: Při provádění této techniky se útočník snaží nalézt aplikaci, která je volně přístupná z internetu. Může se jednat například o webové aplikace, databáze, nebo třeba SSH. Útočník se v této aplikaci snaží nalézt chybu, kterou by mohl využít pro následný přístup do sítě. Obránce může využít honeypotu 2.3.3, který bude vypadat, že obsahuje aplikaci se známou chybou.

- **Využití služeb pro vzdálený přístup**[13]: Po úspěšném útoku na zařízení v síti se útočník snaží získat přístup k dalším zařízením na síti tak, že se pokouší využít chyb ve službách pro vzdálený přístup. Mezi tyto služby může patřit například: SMB, RDP, atd. Podobně jako u „Získávání dat ze síťových disků“ může na síti být přítomen honeypot, který zdánlivě trpí nějakou zranitelností. Díky tomu lze útočníka nalákat k útoku na honeypot, což obránci usnadní detekci útoku a umožní mu poskytnout útočníkovi falešná data.

- **Přesouvání nástrojů útočníkem**[14]: V případě, že se útočníkovi podaří získat kontrolu nad zařízením v síti, je velice pravděpodobné, že se pokusí na dané zařízení přesunout různé nástroje, které využije k následujícím útokům. Pokud by se obránci podařilo tento přesun detekovat, mohl by nástroje nahradit nějakou upravenou verzí, která buď není nebezpečná, nebo například umožní útočníka sledovat.

[9] https://attack.mitre.org/techniques/T1110/
[10] https://attack.mitre.org/techniques/T1602/
[11] https://attack.mitre.org/techniques/T1039/
[12] https://attack.mitre.org/techniques/T1190/
[13] https://attack.mitre.org/techniques/T1210/
[14] https://attack.mitre.org/techniques/T1105/

- **Využití software pro vzdálený přístup**[15]: Útočník se u této techniky pokouší využít legitimního software, jako Team Viewer, AnyDesk, ... pro přístup k zařízení oběti. Obránce může vytvořit honeypoty čekající na příchozí spojení od útočníka. Tyto honeypoty můžou umožnit sledování útočníka a podstrčení falešných dat útočníkovi.

- **Využití webové služby pro ovládání napadeného zařízení**[16]: U této techniky útoku se útočník pokouší využít legitimní webové služby (twitter, facebook, google doc) pro zasílání příkazů napadeným zařízením. Tyto služby může také využít pro extrakci dat z napadených zařízení. Pokud se tento typ komunikace podaří obránci detekovat, může útočníka sledovat, dále může útočníkovi podstrčit falešná data, nebo může touto formou napadeným zařízením také zaslat příkazy, které obránci mohou nějakým způsobem pomoct. V případě nějakého většího útoku lze také doufat, že se k obraně připojí také provozovatelé těchto webových služeb.

2.6 CESNET

Jak lze zjistit z [13] CESNET je sdružení vysokých škol a Akademie věd České republiky, které provozuje a rozvíjí národní e-infrastrukturu pro vědu, výzkum a vzdělávání zahrnující počítačovou síť, výpočetní gridy, datová úložiště, prostředí pro spolupráci a nabízející širokou škálu služeb. Mezi hlavní činnosti CESNETu patří například:

1. provádění nezávislé aktivity výzkumu a vývoje v oblasti informačních a komunikačních technologií a poskytování výzkumné služby v této oblasti

2. podpora vzdělání v oblasti informačních a komunikačních technologií

3. uvádět výsledky výzkumu a vývoje do praxe

V rámci počítačové sítě CESNETu jsou k dispozici i některé nástroje související s bezpečností:

- **Nerd**: Nerd je reputační databáze, jedná se tedy o systém, který sbírá data o různých kybernetických hrozbách z mnoha různých zdrojů a sestavuje databázi známých nebezpečných síťových entit. Poskytuje detailní informace o každé entitě, včetně kdy a kde byla tato entita nahlášena jako nebezpečná, různá pomocná data, jako název a lokaci entity a číselné vyjádření reputace dané IP adresy. Většina dat je také veřejně přístupná z webového rozhraní.[17] [19]

- **Warden**: Warden je systém pro efektivní sdílení informací o detekovaných hrozbách. Umožňuje bezpečnostním týmům sdílet a následně využívat informace o anomáliích na síti detekovaných různými nástroji, mezi které patří např: IDS, honeypoty 2.3.3, síťové sondy, logy, ... [11]

- **Mentat**: Mentat je distribuovaný, modulární SIEM systém vytvořený pro monitorování sítí všech velikostí. Jeho architektura umožňuje příjem, uložení, analýzu, zpracování a následnou odpověď na velké množství bezpečnostních incidentů pocházejících z různých zdrojů. [12]

[15] https://attack.mitre.org/techniques/T1219/
[16] https://attack.mitre.org/techniques/T1102/
[17] https://nerd.cesnet.cz/nerd/ips/

- **Nemea**: Nemea je modulární detekční systém pro analýzu síťového provozu. V praxi je to množina nezávisle běžících modulů, které nepřetržitě zpracovávají příchozí data. Nemea je vhodná pro analýzu dat z toků a to buď v reálném čase, nebo z dříve zachyceného síťového provozu. Momentálně nemea obsahuje moduly pro detekci různé podezřelé komunikace, vypočítávání různých statistik o komunikaci, filtrování a agregování zpráv a následné upozorňování na detekovanou podezřelou komunikaci. [1]

Kapitola 3

Návrh řešení

Tato kapitola je zaměřena na výčet a analýzu různých možností aktivní reakce na síťové útoky. Pokud se obránci podaří odhalit probíhající útok, má možnosti popsané v následující sekci.

Nakonec je z popsaných typů obrany vybrána ta nejvhodnější a je popsán detailní návrh na její implementace.

3.1 Výčet možných reakcí

V této sekci následuje výčet několika možných reakcí na detekovaný síťový útok. Tato sekce se u každé reakce soustředí hlavně na její podrobný popis. Jejich dopad z různých hledisek a jejich porovnání bude diskutován v následujících sekcích 3.2, 3.3.

1. **Zablokování komunikace**: Nejjednodušší reakcí na útok, kterou lze provést téměř vždy je prosté zablokování síťového provozu pocházejícího od útočníka.

2. **Falešná zařízení na síti**: Dalším způsobem obrany by mohlo být útočníka nějak zdržet, zmást a zaměstnat. Například pokud se útočník snaží provést horizontální sken sítě. Lze využít nějakého honeypotu 2.3.3 na síti, který bude odpovídat na pokus o sken volných adres v určitém adresním prostoru. Tím lze útočníka zmást a zdržet, protože je šance, že se bude pokoušet dále skenovat, případně útočit na tento honeypot. Díky tomu obránce získá více času na odhalení útoku a na případnou další reakci.

3. **Falešné otevřené porty na falešných zařízeních**: Podobným způsobem reakce, avšak tentokrát na vertikální sken by bylo na skenovaném zařízení nastavit firewall a použít nějaký software tak, aby zařízení klamně odpovídalo i na portech, na kterých ve skutečnosti není spuštěna žádná služba. Lze tak například vytvořit honeypot, který bude vypadat, jako že na něm běží starší verze nějaké služby, která obsahuje nějakou bezpečnostní chybu 2.2. Útočník se pokusí honeypot oskenovat, zjistí potencionální bezpečnostní díru a pokusí se na ni zaútočit. Jelikož se však jedná o podstrčený software obránce, který žádnou bezpečnostní díru neobsahuje, dává to tedy obránci možnost snadno odhalit probíhající pokus o útok.

4. **Falešné otevřené porty na zařízení oběti**: Postup z předchozího odstavce by však šel upravit také pro použití na skutečných síťových zařízeních. Tedy na potencionálních obětech útoku. Uvažujme opět situaci, kdy se útočník snaží provést vertikální

sken zařízení s cílem odhalit běžící služby a podrobnosti o nich, jako například jejich verze. Tyto informace poté útočník může využít pro zjištění zranitelností a pro jejich následné zneužití k útoku. Obránce by mohl použít podobný software a konfiguraci firewallu jako v předchozím odstavci tak, aby zařízení například kladně odpovídalo na všech portech. Skutečně běžící služby se tedy snadno schovají mezi obrovským množstvím falešných pozitiv a útočník útok pravděpodobně velice brzy vzdá.

5. **Využití vlastností TCP ke zpomalení komunikace**: Zůstaňme ještě chvíli u použití honeypotů ke zdržení a zmatení útočníka. Způsob obrany z předchozích odstavců by šel ještě dále rozšířit pro ještě větší zpomalení útočníka. Představme si, že na honeypotu odpovídá na sken, ať už vertikální, nebo horizontální, námi vytvořený software. Nyní nás nezajímá zda software odpovídá kladně, nebo záporně. Software může využít vlastností TCP IP tak, aby každá jeho odpověď trvala pokud možno co nejdéle a tím se může pokusit útočníka co nejvíce zdržet.

6. **Protiútok**: Často se říká, že nejlepší obranou je útok, takže jednou z možností je také protiútok.

7. **Využití chyby útočícího nástroje**: Zůstaňme ještě u protiútoku. Jistou formou protiútoku, která však na rozdíl od předchozí možnosti nemusí být nelegální by bylo odpovídat na útočníkovu komunikaci takovým způsobem který by využil nějaké chyby v nástroji, který útočník používá. Například pokud útočník používá nmap 2.4.2 pro skenování portů oběti a obránci se podaří objevit chybu v nmapu, mohlo by zařízení obránce na sken odpovídat takovým způsobem, aby této chyby využil. Využitím této chyby by obránce mohl nástroj například ukončit. Díky tomu by zařízení obránce bylo imunní proti skenování. Podobného principu by šlo využít u jakéhokoliv nástroje, který by útočník mohl použít.

8. **Nahrazení zařízení oběti zařízením obránce**: Jednou z posledních možností, která zde bude uvedena je možnost přesměrování toku dat útočníka, který provádí útok na nějaké zařízení, u kterého nevadí, že je na něj útočeno, například honeypot 2.3.3. Obránce poté může toto zařízení využít k tomu, aby útočníkovi vracel pouze takovné odpovědi, jaké chce obránce.

9. **Naslouchání útoku a vydávání se za oběť**: Poslední možností aktivní reakce je variantou obrany z předchozího odstavce. Obránce by mohl místo přesměrování síťového toku a následného vydávání se za oběť, toku dat pouze naslouchat. Data by tedy stále proudila od útočníka k oběti a zpět, avšak s obráncem někde uprostřed. Pokud by však obránce zjistil, že se jedná o útok, mohl by se pokusit útočníkovi odpovídat zároveň s obětí tak, aby útok nějak překazil. Například pokud by útočník prováděl vertikální sken zařízení oběti, oběť by pravdivě odpovídala na stav každého jejího portu. Obránce by však zároveň mohl na útočníkovo skenování také odpovídat, avšak způsobem, aby se všechny porty oběti zdály jako zavřené. Při správném načasování tak bude útočník naslouchat obránci a odpovědi oběti bude ignorovat, čímž se obránci podaří skrýt otevřené porty oběti.

Tabulka 3.1: Tabulka ukazující výhody a nevýhody různých možností obrany proti útokům

Reakce	**Technická náročnost**	**Etika**	**Legálnost**	**Účinek**
1 Zablokování komunikace	Snadná a rychlá na provedení	Bez problémů	Legální	Minimální
2 Falešná zařízení na síti	Je potřeba vytvořit honeypot, který se bude chovat tak, jako by měl více IP adres. Náročnost je vyšší než u předchozí reakce, ale stále je relativně malá	Bez problémů	Legální	Útočníka lze zdržet, bohužel při tomto způsobu nedochází přímo k obraně skutečných síťových zařízení.
3 Falešné otevřené porty na falešných zařízeních	Rozšíření předchozí rekace, na všechny porty. Nárůst v náročnosti spočívá ve složitosti komunikující aplikace.	Bez problémů	Legální	Rozšíření předchozí reakce a proto může být účinnější. Nejedná se o obranu skutečných zařízení.
4 Falešné otevřené porty na zařízení oběti	Předchozí varianta, která je přímo na skutečných zařízeních. Jednodušší než předchozí 2 varianty.	Bez problémů	Legální	Útočníka lze zdržet a také od útoku odradit. Nejedná se o obranu skutečně běžících služeb.
5 Využití vlastností TCP ke zpomalení komunikace	Rozšíření předchozích dvou reakcí, dochází k dalšímu nárůstu v náročnosti.	Bez problémů	Legální	Rozšíření předchozích řešení, vyšší účinek, nefunguje pro skutečné služby.
6 Protiútok	Náročnost se liší v závislosti na zabezpečení původního útočníka a zkušenostech obránce, který bude protiútok provádět.	Neetické	Nelegální	Účinek závisí na schopnostech obránce, může být nejvyšší z diskutovaných variant
7 Využití chyby útočícího nástroje	Extrémně náročné. K nalezení použitelné chyby by bylo potřeba velké množství času a štěstí.	Závisí na typu nalezené chyby a způsobu jejího využití.	Závisí na typu chyby a způsobu využití.	Záleží na nalezené chybě. Účinek může být vysoký. Po opravě chyby zcela bez účinku.
8 Nahrazení zařízení oběti zařízením obránce	Náročné správně směrovat komunikaci skrz celou síť. Zavisí na složitosti komunikující aplikace.	Problémové, pokud se nejedná o skutečný útok.	Problémové, pokud se nejedná o skutečný útok	Nejvyšší po protiútoku a využití chyby útočícího nástroje. Dokáže ochránit skutečně běžící služby.
9 Naslouchání útoku a vydávání se za oběť	Náročnější varianta předchozí reakce. Náročnější směrování i složitější komunikující aplikace	Lepší než předchozí varianta. Stále však naslouchámе cizí komunikaci.	Lepší než předchozí varianta.	Potencionálně stejný jako předchozí varianta.

3.2 Analýza reakcí

Minulá sekce nastínila různé druhy reakcí na různé útoky. Tato sekce naváže na tento výčet, jednotlivé reakce dále rozvede a analyzuje jejich výhody a nevýhody z různých hledisek, jako je jejich provedení, etika, legálnost a zařadí je do fáze útoku podle killchain a MITRE ATT&CK 2.1. Stručné shrnutí této sekce lze nalézt v tabulce 3.1.

1. **Zablokování komunikace**: Jednoznačnou výhodou této reakce je její jednoduchost. Po detekování útoku lze například jedním flowspec 2.3.1 příkazem zcela zablokovat příchozí komunikaci od útočníka. Nevýhodou této reakce však může být to, že při pouhém zablokování útočníkovy komunikace totiž útočníkovi nic nebrání v tom, aby útok prostě vedl odjinud, nebo aby si vybral jiný cíl útoku, který jej nemusí vůbec odhalit. Tato reakce by byla legální, nabízí se však otázka, zda je zcela etické útok prostě přerušit a tím vlastně poslat útočníka útočit na jiné oběti, které nemusí být schopny útok detekovat. Tuto obranu lze provést okamžitě po odhalení jakéhokoliv útoku bez ohledu na to, v jaké fázi se útok nachází.

2. **Falešná zařízení na síti**: Tato reakce je už o něco obtížnější na provedení, než předchozí reakce. Je zapotřebí vytvořit honeypot 2.3.3, který se na síti chová tak, jako by měl více IP adres. Možným řešením by bylo naslouchat arp komunikaci. Pokud se honeypotu podaří zachytit několik arp požadavků, na které nenásleduje odpověď, může honeypot na požadavek sám odpovědět. Díky tomu oklame směrovač a jsou mu odesílány packety pro danou IP adresu. Následně lze například snadno upravit iptables tak, aby honeypot skutečně přijímal a odesílal packety s danou IP adresou. Výhodou této techniky je její jednoduchost, všeho co je napsáno výše lze docílit krátkým skriptem. Nevýhodou však je, že se jedná pouze o zpomalení a zmatení útočníka. Skutečná zařízení budou také stále viditelná a přímo u nich nedochází k žádné obraně. Z pohledu etiky a legálnosti žádný problém nenastává. Tato reakce je účinná pouze ve fázi průzkumu.

3. **Falešné otevřené porty na falešných zařízeních**: Tato reakce je rozšířením předchozí reakce. Zatímco předchozí reakce cílila pouze na fázi průzkumu, kdy útočníkovi při skenování podstrkovala falečná zařízení. Tato reakce se zabývá i následným útokem. Narozdíl od předchozí reakce je tedy zapotřebí navíc vytvořit i aplikaci, která bude komunikovat s útočníkem. Navíc je potřeba nasměrovat komunikaci z více portů na tuto aplikaci, což lze opět snadno provést například pomocí iptables. Náročnost provedení této reakce spočívá zejména ve složitosti aplikace, která provádí komunikaci. Nejjednodušší variantou by mohla být například aplikace, která všechna příchozí spojení zamítne, nebo všechna přijme. V té nejsložitější variantě by aplikace pro větší věrohodnost mohla plně implementovat některé síťové služby. Například by tato aplikace mohla být schopna správně odpovídat na pokus o přihlášení pomocí ssh a na pozadí si zapisovat zadané přihlašovací údaje. Bohužel však podobně jako u předchozího útoku se nejedná o přímou obranu skutečných zařízení na síti, ta jsou pro útočníka stále snadno viditelná a lze na ně útočit. Tato reakce je účinná hlavně ve fázi průzkumu. Narozdíl od předchozí reakce umožňuje na falešných zařízeních také otevřít libovolné porty. V závislosti na aplikaci, která provádí komunikaci může být tato reakce účinná také v dalších fázích útoku. Tato aplikace může útočníkovi například poskytnout shell a může umožnit obránci sledovat útok přes všechny jeho fáze.

4. **Falešné otevřené porty na zařízení oběti**: Reakce je podobná předchozí reakci, avšak místo na honeypotu probíhá na zařízeních potencionálních obětí. Provedení i jeho složitost je stejné jako u předchozí reakce, liší se pouze v tom, že se obrana provádí na skutečných zařízeních. Podobně jako v předchozích variantách se jedná pouze o zpomalení a zmatení útočníka. Bohužel, stejně jako u předchozích reakcí, nedochází k jakékoliv obraně skutečně běžících služeb. Pokud se útočníkovi podaří tento pokus o obranu prohlédnout a mezi všemi falešnými službami objeví skutečné běžící služby, tak mu nic nebrání v provedení útoku podobně, jako kdyby tento typ obrany vůbec nebyl použit. Dalším problémem je, že pro nastavení této obrany je potřeba přístup k zařízením potencionálních obětí. Stejně jako předchozí obrany je tato obrana účinná hlavně ve fázi průzkumu.

5. **Využití vlastností TCP ke zpomalení komunikace**: Jedná se o rozšíření předchozích dvou reakcí. V tomto případě aplikace, která naslouchá komunikaci a odpovídá na ni při komunikaci použije v TCP hlavičce takové údaje, aby například přesvědčila zařízení útočníka o tom, že je síť zahlcená a je potřeba výrazně zpomalit komunikaci. Podobně funguje například projekt LaBrea 2.3.3. Implementace této reakce tedy bude o něco složitější než předchozí reakce. Tato metoda obrany bude ještě lepší ve zpomalení útočníka než předchozí metody. Jelikož se jedná o rozšíření předchozích metod, tak také působí hlavně ve fázi průzkumu.

6. **Protiútok**: Náročnost a proveditelnost protiútoku se velice mění se zkušenostmi obránce, který bude protiútok vykonávat a úrovní zabezpečení útočníka. Jasnou výhodou protiútoku je, že v nejlepším případě je jeho účinnost absolutní. Zkušený obránce může být schopen smazat veškerá ukradená data a také poškodit útočníkovo zařízení, tak že nějakou dobu nebude schopen dalšího útoku. Velkou nevýhodou je legální a etická stránka. Protiútok by ve většině zemí světa byl nelegální. Při pohledu na etickou stánku víme, že útoky jsou často vedeny přes jiná napadená zařízení, tedy taková zařízení, která útočník napadl někdy dříve a teď je pouze využívá. Protiútok by tedy v takovém případě směřoval proti majiteli tohoto napadeného zařízení a ne proti útočníkovi samotnému. Chytřejší útočník by dále mohl zneužívat našeho protiútoku k tomu, abychom za něj napadali jeho cíle. Navíc jako obránce nejsme soudce, abychom útočníka mohli nějak trestat. Tuto možnost obrany lze provést v jakékoliv fázi útoku.

7. **Využití chyby útočícího nástroje**: Velkou nevýhodou provedení tohoto druhu reakce je extrémně náročné hledání chyb v nástrojích útočníků, které často vyžaduje také velkou dávku štěstí. Další nevýhodou je doba, po kterou lze danou chybu využít. Po opravení námi objevené chyby se tato obrana stává zcela neúčinnou. Výhodou je, že pokud se podaří nalézt ta správná chyba v nějakém nástroji, tak je účinnost obrany velice vysoká, podobně jako u protiútoku. V závislosti na typu chyby a jejím zneužití může tento typ reakce nést různé legální a etické problémy. Například využití chyby v nástroji ke smazání útočníkova disku by jistě nebylo v pořádku. Tento typ obrany lze použít proti jakémukoliv nástroji, v jakékoliv fázi útoku.

8. **Nahrazení zařízení oběti zařízením obránce**: Tento způsob reakce je zatím po protiútoku a využití chyby útočícího nástroje asi nejtěžší na provedení. Pro provedení této obrany je potřeba správně přesměrovat tok dat, který teče od útočníka k oběti, směrem k obránci a následně odeslat data od obránce k útočníkovi tak, aby si útočník

myslel, že obránce je oběť. K přesměrování by měl jít použít například flowspec 2.3.1. Při použití flowspecu by mělo být možné docílit správného přesměrování komunikace pomocí již existujících BGP routerů. Výhodou tohoto přístupu je, že pokud síť využívá kompatibilní hardware, tak není potřeba žádných hardwarových změn na síti. Další výhodou je, že by výsledek této práce, který by se staral o vydávání flowspec příkazů mohl být umístěn kdekoliv v síti. Ovšem nevýhodou je velký risk nekompatibility hardware s flowspecem. Další nevýhodou je, že pomocí flowspecu zasahujeme do směrování packetů na síti a případná chyba může způsobit větší problémy, než úspěšný útok. Další možností pro provedení přesměrování tedy je použití speciálního zařízení na síti, které bude pro většinu komunikace průchozí. Komunikace, která není součástí útoku bude beze změny propuštěna dále do sítě. Avšak u komunikace, která je součástí útoku mohou být změněny zdrojové a cílové adresy tak, aby došlo ke správnému přesměrování na zařízení obránce. K vytvoření takového zařízení lze použít například Suricatu 2.3.4, kterou lze spustit v módu, kdy naslouchá na dvou síťových rozhraních a kopíruje veškerou komunikaci z jednoho na druhé. Jednoznačnou výhodou nahrazení zařízení oběti zařízením obránce je, že po odhalení probíhajícího útoku, by tato metoda obrany jako zatím jediná, kromě protiútoku a využití chyby útočícího nástroje, měla být schopna útočníkovi zabránit v úspěšném oskenování běžících služeb na cílovém zařízení. Dále by stejná metoda šla použít i pro pozdější fáze útoku, kdy se útočník snaží zaútočit na konkrétní síťovou službu. V tomto případě se na zařízení obránce může nacházet honeypot, který komunikuje s útočníkem místo původně zamýšlené oběti. Pro komunikaci s útočníkem lze využít podobné aplikace jako v metodách, které využívají „falešné otevřené porty“. Z etického a legálního hlediska může nastat problém, protože se nyní obránce vydává za oběť. Tento problém je zejména závažný, pokud by nešlo o skutečný útok, ale o legitimní komunikaci. Podobně jako metody využívající „falešné otevřené porty“ je tato obrana účinná hlavně ve fázi průzkumu avšak i zde by šlo aplikaci, která komunikuje s útočníkem naprogramovat tak, aby útočníkovi simulovala nějakou skutečnou síťovou službu a umožnila obránci sledovat útok i v dalších fázích.

9. **Naslouchání útoku a vydávání se za oběť**: Jedná se o rozšíření předchozí reakce. Narozdíl od předchozí reakce je útočníkova komunikace od obránce přesměrována dále k oběti a na útok tak odpovídá jak obránce, tak i oběť. Přesměrování komunikace k oběti přidává na složitosti této reakce. Na složitosti také přidává fakt, že odpovědi je potřeba správně načasovat tak, aby útočník považoval za platné odpovědi obránce a odpovědi oběti ignoroval. Nevýhodou tohoto řešení zejména oproti předchozí reakci je, že přidává na složitosti, což přidává nebezpečí selhání této reakce, ale funkcionalita zůstává stále stejná. Tato reakce alespoň částečně řeší etický problém předchozí reakce. Stejně jako předchozí reakce je účinná hlavně ve fázi průzkumu, ale v závislosti na komunikující aplikaci ji lze využít i k reakci v dalších fázích útoku.

Z analýzy reakcí výše je patrné, že většina probíraných reakcí se zaměřuje hlavně na první fázi útoku, tedy na fázi průzkumu. K tomu jsou dva důvody. Hlavním důvodem je, že v této fázi útoku zatím nedošlo k žádným reálným škodám, takže reakce na útok už v této ranné fázi nejlépe umožňuje minimalizaci škod. Dalším nepatrným důvodem je, že probíhající skenování jsme schopni snadno detekovat například s pomocí nástroje Nemea 2.6.

3.3 Vyhodnocení

V předchozí sekci byly uvedeny různé možnosti reakcí na síťové útoky. V této sekci budou jednotlivé možnosti porovnány s cílem nalezení nejvhodnější možnosti obrany.

Začněme vyřazením některých reakcí, které jsou velmi málo účinné, nebo je v našich podmínkách nelze provést. První takovou reakcí je zablokování komunikace, protože není moc účinná, útočník může prostě vést útok odjinud a obránce ho nemusí být schopen znovu odhalit. Případně si útočník může nalézt jinou oběť a velice rychle pokračovat v útoku. Další reakcí kterou můžeme vyloučit je protiútok. Cílem této práce je nalézt možnou obranu proti útoku, ne provádění útoků. Navíc je tato reakce v řadě zemí nelegální a tedy zcela nepoužitelná. Poslední možností reakce, kterou můžeme okamžitě vyřadit je nalezení a následné využití chyby v nástroji používaném útočníkem. Obránce by potřeboval velké množství znalostí o daném nástroji, času a štěstí, aby byl schopen nějakou využititelnou chybu odhalit. Zároveň pokud není obránce schopen tyto chyby hledat pravidelně, tak je tato obrana v podstatě zcela neúčinná, protože lze předpokládat, že útočník může snadno chybný nástroj nahradit podobným nástrojem, který žádnou chybu neobsahuje. Dále lze předpokládat, že jakákoliv nalezená chyba bude rychle opravena, díky čemuž by taková reakce na útok byla využitelná pouze po velice krátkou dobu.

Nyní nám zbyly pouze možnosti, které jsou opravdu schopny pomoci při obraně a zároveň je možné je legálně provést. Dále by bylo vhodné, aby konečné řešení obrany bylo schopno přímo reagovat na probíhající útok na zařízení oběti. Vyřaďme tedy také možnosti „Falešná zařízení na síti“ a „Falešné otevřené porty na falešných zařízeních“, které na síti vytváří fiktivní zařízení, jsou schopny oběti mezi těmito zařízeními skrýt a útok oddálit, pokud však útočník dokáže odhalit která zařízení jsou skutečná, tak na ně může začít útočit a tyto reakce se stávají zcela neúčinnými.

Zbývají nám tedy poslední čtyři možnosti: „Falešné otevřené porty na zařízení oběti“, „Využití vlastností TCP ke zpomalení komunikace“, „Nahrazení zařízení oběti zařízením obránce“ a „Naslouchání útoku a vydávání se za oběť“. Zásadním rozdílem mezi těmito možnostmi je, že první dvě možnosti vyžadují přímý přístup k zařízení oběti. Je potřeba na všech zařízeních, u kterých se v případě útoku chceme pokusit o aktivní reakci, správně nainstalovat a nakonfigurovat software, který bude reakci provádět. Zároveň oba tyto přístupy trpí podobnou nevýhodou jako reakce z předchozího odstavce. Obě tyto reakce jsou schopny útočníka zmást a zdržet tak, že mezi skutečně běžícími službami vytvoří několik falešných služeb a následně mohou reagovat při pokusu o útok na tyto služby. Bohužel však nedokážou zabránit nalezení a následnému útoku na skutečně běžící služby.

Nejvhodnější volbou se tedy zdají poslední dvě možnosti. Oproti dříve zmíněným možnostem mají jisté výhody:

- Instalace a konfigurace probíhá pouze na jednom zařízení na síti, které je zcela pod kontrolou obránce.
- Jako jediné jsou schopny před útočníkem skrýt skutečně běžící služby a tak při včasném odhalení zcela zabránit útoku.

Zároveň jsou si tyto reakce velice podobné, jediným podstatným rozdílem je, zda komunikace mezi obětí a útočníkem skutečně dorazí až k oběti a zároveň zda skutečné odpovědi oběti dorazí k útočníkovi. Tento rozdíl téměř nemění funkcionalitu reakce, avšak zásadně mění složitost jejího provedení.

Tato práce se tedy bude dále zabývat hlavně reakcí „Nahrazení zařízení oběti zařízením obránce“, která se jeví jako nejvhodnější z uvažovaných možností aktivní reakce. Po

úspěšné implementaci této reakce práce ověří také proveditelnost reakce „Naslouchání útoku a vydávání se za oběť“.

3.4 Návrh

V této sekci bude popsán podrobný návrh vybraných řešení z předchozí sekce včetně použitých technologií a podrobnějšího vysvětlení navrhovaného řešení. V předchozí sekci byl kladen důraz na reakci na skenování. Dále však u zvolených reakcí pro implementaci bylo zmíněno, že by tyto reakce šly použít i v pozdějších fázích útoku. Návrh popsaný v této sekci se tedy bude zabývat jak reakcí na skenování, tak i reakcí na útok na konkrétní služby v pozdějších fázích útoku.

Představme si, že na síti probíhá útok, na který by měl výsledek této práce reagovat. Typicky by se takový útok dal rozdělit do následujících tří fází. Tato práce by měla být schopna reagovat jak na situace, kdy se útok skládá ze všech těchto částí, tak i na situace, kdy se útok skládá pouze z některých z nich. Například proto, že se dřívější části útoku nepodařilo detekovat.

1. Útočník provádí skenování za účelem odhalení běžících služeb na zařízeních.
   (a) Provádí vertikální sken za účelem odhalení běžících služeb na konkrétním zařízení.
   (b) Provádí horizontální sken za účelem nalezení zařízení, na kterém běží konkrétní služba.
2. Útočník odhalil běžící službu, kterou použije pro svůj útok a nyní provádí hádání přístupových údajů k odhalené službě.
3. Útočníkovi se podařilo správně uhodnout přístupové údaje, úspěšně je použil pro přihlášení a nyní se pohybuje po napadeném systému.

Nyní předpokládejme, že na síti, kromě řešení této práce, běží také Nemea 2.6, nebo jiný detektor útoku, který neustále v reálném čase skenuje síťový provoz a předává informace o detekovaných útocích Wardenu. Nakonec předpokládejme, že se tomuto detektoru podaří útok skutečně detekovat.

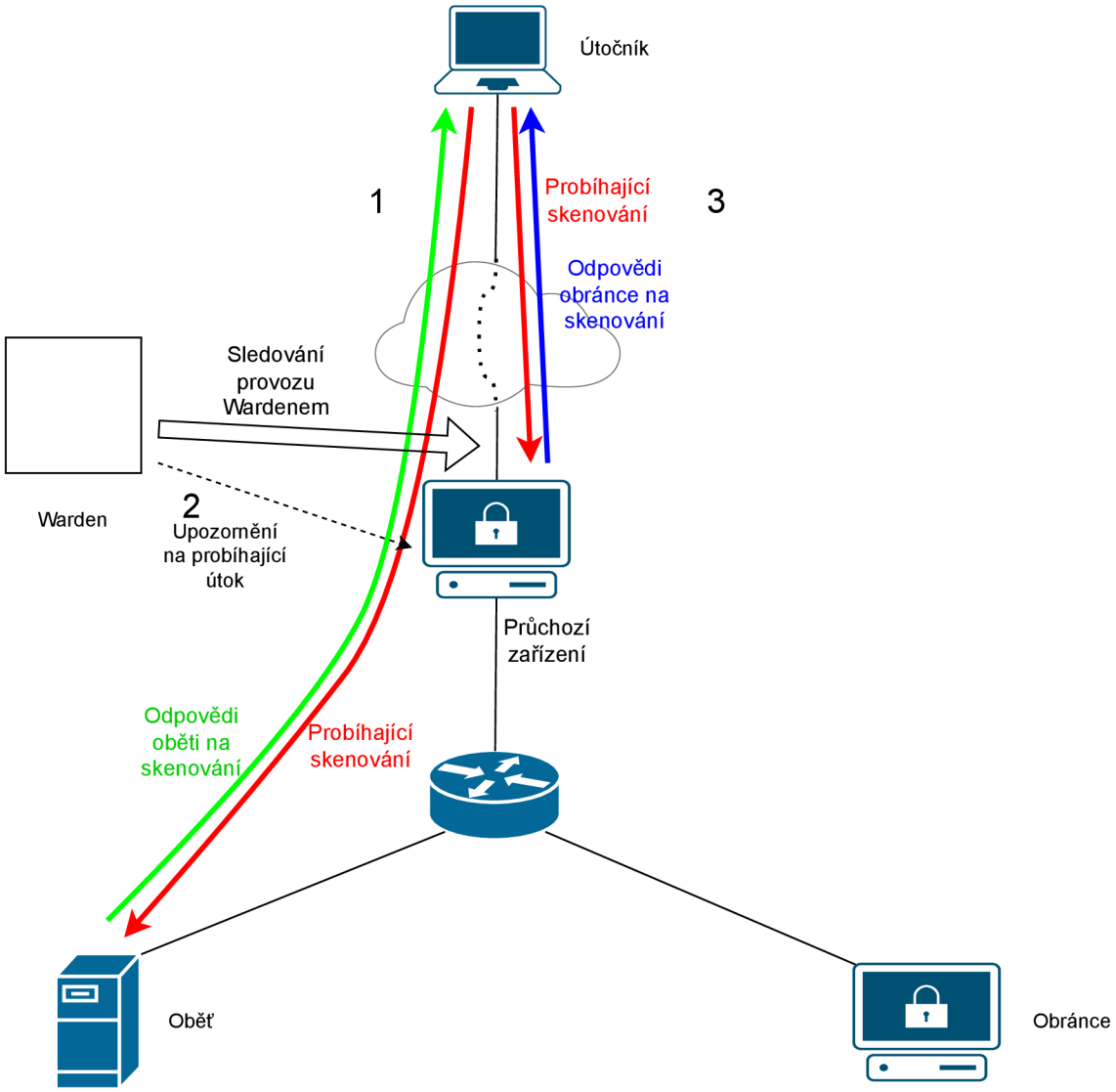
Po odhalení útoku tedy detektor na probíhající útok upozorní Warden a z Wardena následně veškeré potřebné informace získá skript, který bude výsledkem této práce a bude na ně reagovat.

V předchozí kapitole byla popsána žádaná reakce na výše popsaný útok. Touto reakcí podle jednotlivých částí ve zkratce je:

1. Umožnění falešných, předkonfigurovaných odpovědí na sken.
2. Sesbírání slovníku použitých přihlašovacích údajů.
3. Umožnění přihlášení do honeypotu a sledování útočníka.

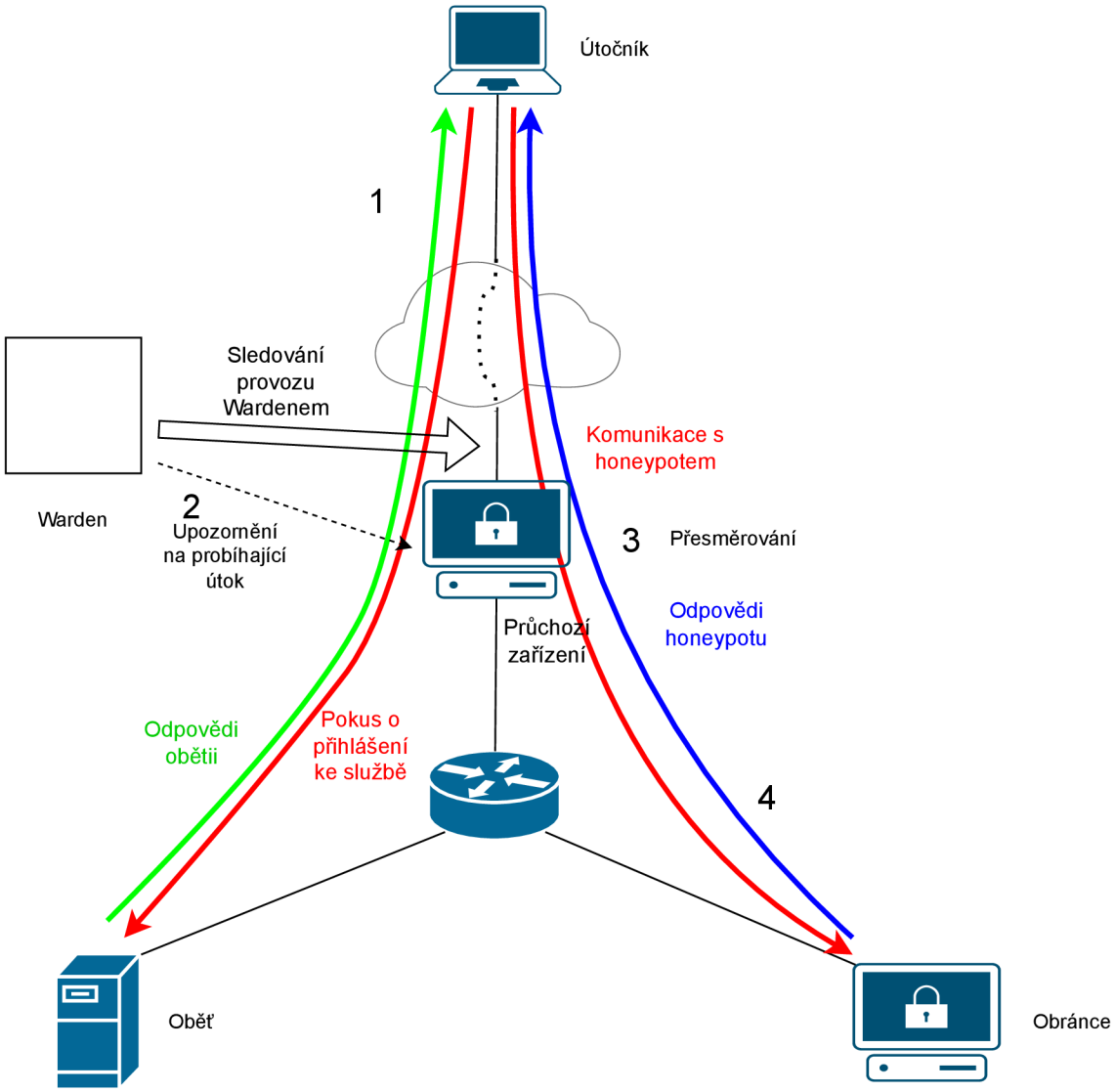
Schéma útoku a následné reakce si lze prohlédnout na obrázcích 3.1 a 3.2. Jednotlivé fáze reakce jsou v obrázcích očíslovány. Na prvním obrázku lze vidět reakce na skenování. V první fázi útok zatím nebyl odhalen a probíhá skenování oběti. Druhá fáze zobrazuje odhalení útoku a upozornění na něj. Třetí fáze již zobrazuje finální podobu zbytku útoku, kdy na pokusy o skenování místo oběti odpovídá řešení této práce. Na druhém obrázku lze vidět

reakce na pokus o komunikaci s nějakou službou, která je poté přesměrována na honeypot. Podobně jako u předchozího obrázku v první fázi až do odhalení útoku probíhá komunikace přímo mezi útočníkem a obětí. Ve druhé fázi je znázorněna detekce útoku a upozornění na něj. Následuje třetí fáze, kdy se provede přesměrování útoku. Ve čtvrté a finální fázi je zobrazen stav, kdy došlo k přesměrování a útočník nyní komunikuje s honeypotem místo s obětí.



Obrázek 3.1: **Schéma probíhající reakce na skenování**

Tento seznam reakcí vede na tři komponenty. První komponentou bude Odpovídač, který se bude starat o první část reakce na útok a bude odpovídat na útočníkovy skeny. Druhou částí bude nějaký honeypot. Po prozkoumání dostupných honeypotů bylo zjištěno, že většina honeypotů zvládá druhý bod reakce na útok, tedy sesbírání slovníku použitých přihlašovacích údajů. Avšak u třetího bodu reakce nemusí být z pohledu útočníka zcela přesvědčivé. Po důkladnějším výběru, nebo po jeho vylepšení, by však kvalitní honeypot měl splňovat poslední dva body reakce. Odpovídač a honeypot však nebudou sami o sobě



Obrázek 3.2: **Schéma probíhající reakce na útok na síťovou službu**

stačit. Honeypot by dokázal reagovat na útoky, které jsou směřovány přímo proti němu. Tato práce však chce reagovat také na útoky, které prochází přes zařízení, na kterém je výsledek této práce nainstalován, avšak skutečně zamýšlenou obětí je zařízení někde dále v síti. K tomuto účelu bude potřeba ještě třetí komponenta: Přesměrovávač. Přesměrovávač zachytí útočníkovu komunikaci a přesměruje ji na honeypot. Následně zachytí komunikaci od honepotu a přesměruje ji zpět k útočníkovi. Další výhodou Přesměrovávače je také fakt, že lze honeypot umístit někde jinde v síti a tím rozložit zátěž v případě, že bude probíhat více útoků najednou. Přesměrovávač dále také umožní využití více honeypotů s různými vlastnostmi na různých zařízeních na síti. Jistě ne poslední výhodou toho, že lze honeypot mít na libovolném zařízení na síti, je bezpečnost. Síťové zařízení, které je vyhrazeno pouze pro honeypot, půjde lépe zabezpečit než zařízení, na kterém běží i některé další služby, jako například Odpovídač a Přesměrovávač. V případě, že by bylo potřeba probíhající útok

přerušit a zařízení s honeypotem například vypnout, nedojde k vypnutí žádné další běžící služby.

3.5 Odpovídač

Funkce odpovídače je docela jednoduchá. Odpovídač bude naslouchat na nakonfigurovaném síťovém rozhraní a bude zkoumat příchozí komunikaci. Pokud objeví komunikaci, kterou Warden označil jako probíhající sken, tak na ni v závislosti na své konfiguraci odpoví místo původně zamýšlené oběti. Tyto odpovědi budou záležet pouze na konfiguraci Odpovídače, nehledě na skutečný stav skenovaného zařízení. Možnosti odpovědí budou tři. Odpovídač může příchozí spojení buď příjmout a tím útočníkovi dát dojem, že je daný port otevřený a že se tato služba na skenovaném zařízení skutečně nachází. Druhou možností je, že příchozí spojení odmítne, což bude pro útočníka vypadat, že je port zavřený a že se na skenovaném zařízení daná služba nenachází. Poslední možností je, že Odpovídač bude žádost o spojení prostě ignorovat. Pro útočníka bude tedy daný port vypadat jako filtrovaný a nebude tedy vědět, zda se na daném portu nachází nějaká služba, nebo ne. Toto nastavení také může prodloužit čas potřebný k provedení skenu, protože na rozdíl od předchozích dvou možností si skenovací nástroj nemůže být jistý, zda je port filtrovaný, a nebo zda nedošlo k chybě při přenosu. V takové situaci minimálně nmap provádí další pokus o spojení. Pokud tedy obránce nastaví všechny porty jako filtrované, tak nejenom, že se útočník nic nedozví, ale sken bude teoreticky probíhat až dvojnásobnou dobu.

3.6 Přesměrovávač

Podobně jako Odpovídač bude Přesměrovávač naslouchat na nakonfigurovaném síťovém rozhraní a bude zkoumat příchozí komunikaci. Pokud objeví komunikaci, kterou bude podle konfigurace potřeba přesměrovat, tak ji přesměruje. Celý tento proces bude muset fungovat také i v opačném směru. Přesměrovávač tedy bude muset vždy po přesměrování nějaké komunikace začít naslouchat také na zařízení, ze kterého danou komunikaci vyslal. Pokud v této zpětné komunikaci Přesměrovávač objeví zprávu, která pochází z cíle přesměrování, bude muset tuto zprávu opět přesměrovat k původnímu zdroji přesměrování.

Přesměrování by mělo jít provést tak, že pokud přesměrovávač objeví zprávu, kterou je potřeba přesměrovat, nahlédne do své konfigurace na údaje o cíli přesměrování. Následně by měl v této zachycené komunikaci přepsat cílovou IP adresu na IP adresu cíle z konfigurace. Dále by měl přepsat cílovou MAC adresu na MAC adresu příštího skoku. Zdrojové IP a MAC adresy záleží na směru přesměrovávání. Pokud se přesměrovává ve směru od útočníka k honeypotu, tak musí Přesměrovávač přepsat tyto adresy tak, aby souhlasily s adresami svého vlastního síťového rozhraní, odkud bude zprávu znovu odesílat. Díky tomu lze docílit toho, že honeypot bude odpovědi odesílat zpět k Přesměrovávači, který je poté přesměruje k útočníkovi. Pokud se jedná o přesměrovávání od honeypotu k útočníkovi, tak musí Přesměrovávač jako zdrojovou IP adresu nastavit IP adresu původní oběti, aby si útočník myslel, že odpověď pochází od zamýšlené oběti. Jako zdrojovou MAC adresu pak musí nastavit MAC adresu svého vlastního síťového rozhraní, odkud bude zprávu odesílat. Nakonec před odesláním každého přesměrovaného packetu musí Přesměrovávač přepočítat IP a TCP checksum. Shrnutí změn zobrazuje tabulka 3.2

Tabulka 3.2: Tabulka ukazující potřebné změny adres při přesměrovávání

Adresa	**Útočník -> Honeypot**	**Honeypot -> Útočník**
Zdrojová MAC	MAC Přesměrovávače	MAC Přesměrovávače
Cílová MAC	MAC příštího skoku	MAC příštího skoku
Zdrojová IP	IP Přesměrovávače	IP oběti
Cílová IP	IP honeypotu	IP útočníka

3.7 Honeypot

V této práci bude honeypot plnit druhé dvě části reakce na útok. Honeypot, případně více honeypotů, bude naslouchat příchozí komunikaci a bude simulovat odpovědi reálných služeb na daných portech. Pokud honeypot zaznamená pokus o přihlášení ke službě, tak zaznamená přístupové údaje tak, aby bylo později možné získat slovník použitých přístupových údajů. V závislosti na konfiguraci a na možnostech honeypotu po nějaké době umožní útočníkovi přihlášení a bude s útočníkem interagovat podobně, jako by to dělala skutečná služba, za kterou se honeypot vydává. Tato interakce by měla být pokud možno k nerozeznání od interakce se skutečnou službou. Dále by mělo být možno detailně zaznamenávat každý krok útočníka, aby bylo možné se z takového útoku poučit a příště na podobný útok lépe reagovat. Jako honeypot by mělo být použito nějaké již existující řešení. Toto řešení by mělo být stále aktivně vyvíjeno tak, aby byly opravovány případné nalezené chyby. Dále by toto řešení mělo podporovat co nejvěrnější simulaci co největšího počtu síťových služeb. Pokud nebude možné nalézt existující řešení, které by splňovalo veškeré předchozí podmínky, bude cílem použít nejlepší řešení a doupravit jej tak, aby splňovalo veškeré podmínky.

Kapitola 4

Implementace

Pro implementaci všech částí řešení jsem zvolil jazyk python z několika důvodů. Honeypot pouze rozšiřuji o novou funkcionalitu, u této komponenty tedy nebylo na výběr. U zbylých komponent jsem zvolil python z důvodu konzistence s honeypotem a také z důvodu jeho jednoduchosti, což umožňuje pozdější snadné rozšíření a úpravy mého finálního řešení. Veškerý vývoj a testování probíhalo na různých distribucích linuxu, což je také platforma, na které by tato práce měla být později spouštěna.

Komponenty Odpovídač a Přesměrovávač jsou implementovány společně jako jeden skript. Honeypot je implementován zvlášť a může dokonce bežet na jiném zařízení na síti než Odpovídač a Přesměrovávač

4.1 Významné použité technologie

V této sekci budou popsány některé významné technologie použité při implementaci. Při implementaci byly použity i některé další technologie a knihovny, avšak rozsah jejich použití nebyl příliš velký.

4.1.1 Python

Python [8] je jednoduchý, všestranný interpretovaný programovací jazyk. Poskytuje efektivní implementaci základních datových struktur a jednoduchý, ale efektivní přístup k objektově orientovanému programování. Je velice rozšířený, což spolu s jeho jednoduchostí znamená vysokou pravděpodobnost, že i neznalý vývojář syntaxi velice snadno porozumí a je schopen snadno a rychle porozumět neznámému kódu. Další výhodou pythonu je velké množství nabízených knihoven, které značně usnadňují a urychlují vývoj.

4.1.2 Scapy

Scapy [10] je python knihovna pro interaktivní manipulaci s packety. Scapy dokáže vytvořit, nebo dekódovat velké množství různých protokolů. Dokáže odesílat a přijímat data a mnoho dalšího. V práci byla tato knihovna použita převážně pro parsování zachycených packetů, změnu hodnot v packetu a odesílání packetů.

4.1.3 Docker

Docker, nebo podman jsou nástroje, které umožňují vytváření a následné spouštění kontejnerů. Díky tomu je tato práce schopna vytvořit zdání skutečného zařízení, na které útočník

útočí, i když se ve skutečnosti jedná pouze o kontejner oddělený od hostitelského zařízení. Vytváření vlastních obrazů kontejnerů je velice snadné pomocí tzv. Dockerfile. Uživatel této práce je tedy schopen snadno vytvořit vlastní obraz s vlastními daty, který poté poskytne útočníkovi k útoku. Při testování a experimentování s touto prací byly využity kontejnery pro vytvoření laboratorní sítě. Veškeré zdrojové soubory pro vytvoření všech použitých kontejnerů jsou součástí odevzdaného projektu.

4.2 Implementace jednotlivých komponent

V této sekci bude detailně popsána implementace jednotlivých komponent této práce.

4.2.1 Odpovídač

Jedná se o základní komponentu této práce. Jak již bylo dříve zmíněno, pro její vývoj byl použit jazyk python spolu s knihovnou scapy, která je použita pro snadnější parsování zachycené komunikace a následné sestavování nových packetů, které jsou poté odesílány zpět útočníkovi.

Po spuštění skriptu se nejprve načte konfigurace ze souboru ve formátu yaml. Tato konfigurace je rozdělena do následujících sekcí, které zásadně mění fungování celého projektu:

- **filter** Tato sekce umožňuje uživateli specifikovat filtr ve formátu bpf, kterým lze popsat komunikaci, na kterou má skript reagovat. Komunikace, která nebude souhlasit s filtrem bude skriptem ignorována.
- **reroute_targets** V této sekci může uživatel specifikovat cíle, kam následně přesměrovávat komunikaci a slouží hlavně pro Přesměrovávač
- **behavior** Tato sekce dovoluje nastavit chování skriptu pro jednotlivé porty. Každý port se může chovat jako otevřený, zavřený, nebo filtrovaný. Další možností je u portů specifikovat cíl ze sekce výše, na který má být komunikace na tomto portu přesměrována.
- **information_input** Zde lze specifikovat různé způsoby, kterými se má skript dozvědět o probíhajících útocích a o komunikaci, na kterou by měl reagovat. V současnosti tato sekce podporuje kompletní nastavení warden klienta.

Po načtení konfigurace skript spustí dvě vlákna, které běží až do jeho ukončení a vykonávají veškerou funkcionalitu Odpovídače.

Information poller

Prvním vláknem je information poller, funkce tohoto vlákna je velice jednoduchá. Na začátku podle konfigurace inicializuje warden klienta a následně se po celou dobu běhu vlákna opakovaně dotazuje wardena na nové bezpečnostní hrozby. Pokud warden vrátí novou hrozbu, která odpovídá konfiguraci, skript rozšíří filtr z konfigurace tak, aby zahrnoval i tuto novou hrozbu a uvědomí druhé vlákno.

Tabulka 4.1: Tabulka ukazující chování odpovídače v závislosti na konfiguraci daného portu a obdrženém packetu

Konfigurace / Druh packetu	**Otevřený**	**Zavřený**	**Filtrovaný**	**Přesměrovaný**
SYN	SYN + ACK	RST + ACK	nic	Packet je předán Přesměrovávači
ACK nebo ACK + FIN	RST	RST	nic	Packet je předán Přesměrovávači
Jiný packet	nic	nic	nic	Packet je předán Přesměrovávači

Listener

Listener naslouchá příchozí komunikaci na raw socketu opatřeném filtrem poskytnutým konfigurací a prvním vláknem. Pokud zachytí TCP packet odpovídající filtru, tak se řídí podle konfigurace v sekci behavior v závislosti na druhu obdrženého packetu podle tabulky 4.1.

4.2.2 Přesměrovávač

Tato komponenta je součástí stejného skriptu jako Odpovídač. Využívá tedy také jazyk python spolu s knihovnou scapy.

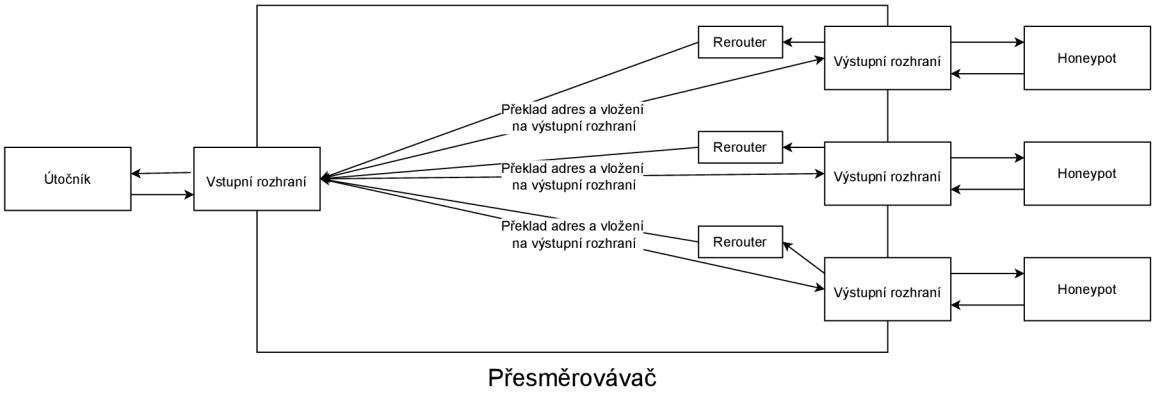
Konfigurace Přesměrovávače probíhá ve stejném konfiguračním souboru jako konfigurace Odpovídače, konkrétně v sekci reroute_targets. Tato sekce konfigurace dovoluje definovat libovolné množství cílů, kam směrovat zachycenou komunikaci. Definice každého cíle obsahuje cílovou IP adresu, MAC adresu příštího skoku a výstupní síťové rozhraní.

Přesměrovávač je volán Odpovídačem v okamžiku, kdy Odpovídač zachytí packet s cílovým TCP portem, pro který je v sekci konfigurace behavior nastaven některý z cílů pro přesměrování. U každého takového packetu Přesměrovávač přepíše cílové adresy údaji z konfiguračního souboru a zdrojové adresy tak, aby odpovídaly síťovému rozhraní, ze kterého se bude packet posílat dále. Takto upravený packet je následně odeslán z nakonfigurovaného síťového rozhraní.

Díky tomuto lze uskutečnit komunikaci směrem od útočníka k nakonfigurovanému cíli, avšak ne komunikaci od nakonfigurovaného cíle zpět k útočníkovi. Z tohoto důvodu Odpovídač spustí pro každé síťové rozhraní, ze kterého se odesílají přesměrované packety, instanci třídy Rerouter. Každá instance této třídy běží v samostatném vláknu a pomocí raw socketu naslouchá na síťovém rozhraní zpětné komunikaci od nakonfigurovaného cíle k útočníkovi. Pokud Rerouter zachytí packet, který je součástí zpětné komunikace k útočníkovi, provede podobnou změnu adres jako provedl Přesměrovávač v opačném směru komunikace tak, aby útočník nic nepoznal.

Posledním problémem při zpětném přesměrovávání je, že firewall na zařízení, na kterém Přesměrovávač běží, neočekává zpětnou komunikaci k útočníkovi. Proto rerouter upraví iptables tak, že na komunikaci která odpovídá zpětné komunikaci od nakonfigurovaného cíle k útočníkovi, reaguje akcí drop. Iptables tedy tuto komunikaci ignoruje, namísto jeho tradiční reakce, kdy by na neočekávanou komunikaci reagoval RST packety a tím narušoval komunikaci mezi útočníkem a přesměrovaným cílem.

Schématický nákres fungování přesměrovávače lze vidět na obrázku 4.1. Na obrázku je vyobrazeno, že hlavní vlákno naslouchá na vstupním rozhraní, provádí překlad adres a odeslání packetů na výstupní rozhraní. Každé výstupní rozhraní má poté svůj vlastní rerouter, který provádí přesměrování v opačném směru.



Obrázek 4.1: **Schéma Přesměrovávače**

4.2.3 Honeypot

Další použitou komponentou je honeypot, který dokáže simulovat různé síťové služby. Pro výběr honeypotu jsem použil seznam honeypotů awesome-honeypots[1]. Zaměřil jsem se na honeypoty, které:

- Dokážou simulovat více síťových služeb
- Jsou stále vyvíjené, konkrétně poslední změna kódu proběhla nejpozději v roce 2022.

Z těchto honeypotů jsem nakonec vybral honeypot s jednoduchým názvem: „Honeypots“[2]. Jedním z důvodů výběru bylo také to, že je součástí kolekce honeypotů T-Pot[3] a případné vylepšení tohoto honeypotu v rámci této práce se tedy stanou také součástí tohoto projektu.

V průběhu výběru konkrétního honeypotu jsem se také zaměřil na způsob implementace simulace služby ssh. Většina zkoumaných honeypotů dovolovala zaznamenávat přihlašovací údaje a tím zaznamenat slovník použitý útočníkem při slovníkovém útoku. Mnoho honeypotů také útočníkovi dovolovalo se do honeypotu pomocí ssh přihlásit a následně se snažilo reagovat na útočníkovy příkazy a tím mu dát zdání, že se přihlásil na skutečný systém. Mnou zkoumané honeypoty však implementovaly pouze několik málo příkazů. U Honeypots to byly konkrétně příkazy: ls, pwd, whoami, exit. Útočník byl tedy schopen velice snadno poznat, že něco není v pořádku.

Z tohoto důvodu jsem se rozhodl upravit Honeypots tak, aby útočníka po přihlášení přesměroval do docker kontejneru. Díky tomu útočník získá přístup k plnohodnotnému shellu, tudíž pro něj bude složitější poznat, že se jedná o honeypot. Zároveň si může obránce vytvořit vlastní image kontejneru, který může podle libosti naplnit falešnými daty, které takhle podstrčí útočníkovi. Použitý způsob přesměrování do docker kontejneru také dovoluje kompletní záznam komunikace mezi útočníkem a docker kontejnerem. Tento záznam komunikace je zapsán do souboru a lze jej následně přehrát pomocí přehrávače útoku.

[1] https://github.com/paralax/awesome-honeypots
[2] https://github.com/qeeqbox/honeypots
[3] https://github.com/telekom-security/tpotce

Přesměrování útoku do docker kontejneru funguje tak, že při útočníkově úspěšném přihlášení honeypot otevře TCP socket mezi útočníkem a honeypotem. Následně honeypot spustí bash v novém docker kontejneru a použije python knihovnu „docker“ k otevření socketu, který komunikuje s stdin, stdout a stderr bashe v kontejneru. Poté honeypot spustí vlákno, které na socketu naslouchá útočníkovým vstupům, vstupy zaznamená do souboru a následně je beze změny překopíruje do socketu komunikujícího s kontejnerem. Hlavní vlákno mezitím dělá stejnou práci, ale v opačném směru, tedy naslouchá výstupům kontejneru, ty zaznamenává do souboru a následně je kopíruje do socketu komunikujícího s útočníkem. Výsledkem je, že útočník komunikuje skrz honeypot s kontejnerem, aniž by si čehokoliv všiml.

Formát souboru je následující. Každý záznam v souboru začíná posloupností znaků: "I:", nebo "O:", které značí, zda se následující záznam týká útočníkova vstupu, nebo výstupu z kontejneru. Následuje celé číslo o délce čtyř bytů little endian, které značí délku záznamu. Nakonec se v souboru nachází samotný záznam vstupu nebo výstupu.

Podobný přístup, kdy je útočník přesměrován do docker kontejneru, na kterém pokračuje v útoku a je při tom sledován lze v budoucnosti rozšířit i na další služby, které po přihlášení útočníkovi poskytují shell, příkladem může být například telnet. Dalším možným rozšířením je namísto bashe v kontejneru spouštět přímo skutečný server pro danou službu. Útočník by se tedy například mohl snažit připojit na ftp server, honeypot by spustil docker kontejner, ve kterém by spustil ftp server a stejným způsobem by sledoval a zaznamenával veškeré útočníkovy vstupy a výstupy ze serveru. Tento záznam by pak pravděpodobně bylo možné přehrát pomocí stejného přehrávače, který je součástí současného řešení.

4.2.4 Přehrávač útoku

Poslední komponentou naimplementovanou v rámci této práce je jednoduchý přehrávač útoku. Tento přehrávač je rovněž implementován pomocí pythonu. Přehrávač načítá proběhlý útok ze souboru, který vznikl při záznamu útoku honeypotem, což je popsáno v předchozí sekci 4.2.3. Tato komponenta podporuje dva různé režimy spuštění.

- **input**: V tomto režimu přehrávač ze souboru jednoduše extrahuje a následně vypíše posloupnost útočníkových zadaných příkazů.
- **output**: V tomto režimu se přehrávač pokusí přehrát útok tak, jak jej viděl útočník. Lze tak vidět veškerý vstup a výstup v terminálu, stejně jako jej viděl útočník během útoku, včetně například pohybu kurzoru v terminálu pomocí šipek, mazání znaků pomocí backspace atd. Přehrávač umožňuje také zobrazení grafičtějších aplikací. Při testování šlo znak po znaku vidět, jak útočník edituje soubory například pomocí terminálové aplikace vim.

V budoucnu by mohl přehrávač být snadno rozšířen do podoby, kdy by byl schopen v reálném čase přehrávat právě probíhající útok. Dalším vylepšením by bylo například pro záznam útoku použít formát souboru, který je používán projektem asciinema[4], který slouží pro záznam a následné přehrávání operací v terminálu. Avšak po dokončení implementace stávajícího přehrávače jsem dospěl k názoru, že použití asciinema momentálně nepřináší žádné výhody navíc.

[4]https://asciinema.org/

Kapitola 5

Testování

Tato kapitola popisuje laboratorní prostředí, ve kterém probíhal vývoj a veškeré experimenty. V pozdějších sekcích se lze dočíst o samotných experimentech a jejich výsledcích. Experimenty proběhly celkem tři. Ve všech experimentech se jedná o spuštění projektu v laboratorní síti a následného provedení útoků. Rozdílem mezi experimenty je rozdílná konfigurace jednotlivých komponent a různé umístění některých komponent v laboratorní síti.

5.1 Popis prováděného útoku při experimentech

V této sekci popíšu útok a použité nástroje útočníka. Stejný typ útoku se bude provádět u všech experimentů.

1. **Skenování**: Bude proveden pokus o skenování portů oběti pomocí nástrojů nmap a massscan 2.4.2. Pro spuštění nástrojů byly využity následující příkazy 5.1:

```
# -Pn - preskoci overeni, zda je cil on-line a tim urychli sken
# -sS - typ skenu - SYN sken. Ostatni typy skenu take funguji podle
    ocekavani
# -r - skenovani portu postupne od nejnizsiho po nejvyssi. Nemeni nic
    na funkcionalite, pouze usnadnuje pozdejsi zjistovani na ktere
    porty odpovidala obet a na ktere uz odpovidal odpovidac
nmap -Pn -sS -r 10.1.1.2 -p1-500
masscan 10.1.1.2 -p1-500
```

Výpis 5.1: Příkazy pro skenování

2. **Hádání hesla**: Bude proveden pokus o uhádnutí hesla na ssh server oběti pomocí nástroje hydra 2.4.3. Pro spuštění nástroje byl použit následující příkaz 5.2.

```
hydra -l ssh -P /usr/share/wordlists/rockyou.txt ssh://10.1.1.2
```

Výpis 5.2: Příkaz pro hádání hesel

3. **Průnik na ssh server**: Budu předpokládat, že se útočníkovi podařilo uhádnout správné heslo. Toto heslo bude použito pro přístup k ssh serveru a extrakci dat.

5.2 Očekávaná reakce na jednotlivé fáze útoku

V této sekci následuje očekávaná reakce této práce na výše uvedený útok.

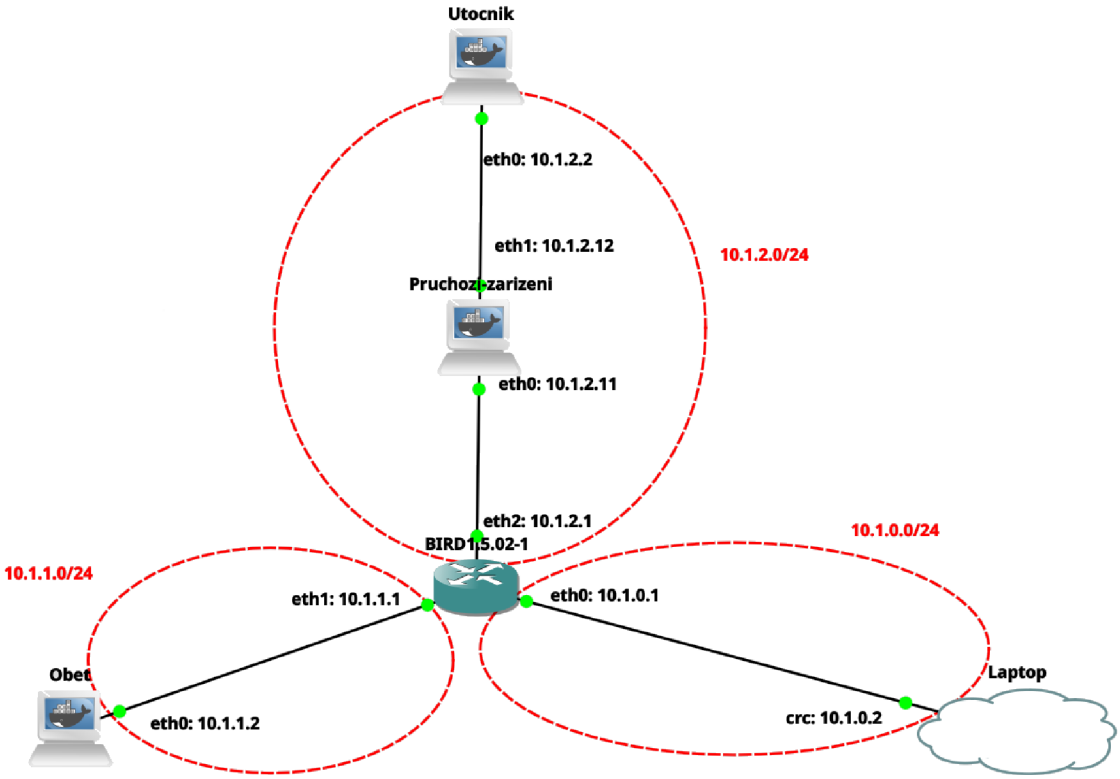
1. **Skenování**: V průběhu skenování bude nasimulována komunikace od wardenu, která bude obsahovat informaci o probíhajícím skenu. Očekává se, že odpovědi na skenování převezme odpovídač. Dále se očekává, že pokud toto převzetí proběhne v průběhu spuštěného skenu, tak si nmap, ani deepscan ničeho nevšimnou a budou pokračovat ve skenu dále. Nakonec se očekává, že výsledky skenu od okamžiku, kdy na sken začal reagovat odpovídač, budou souhlasit s konfigurací odpovídače.

2. **Hádání hesla**: V průběhu hádání hesla bude nasimulována komunikace od wardenu, která bude obsahovat informaci o probíhajícím pokusu o hádání hesla. Očekává se, že od té doby přesměrovávač úspěšně přesměruje všechny další pokusy o uhádnutí hesla na honeypot. Dále se očekává, že pokud toto přesměrování proběhne v průběhu spuštěného útoku, tak si hydra ničeho nevšimne a bude dále pokračovat v hádání hesla ssh serveru na honeypotu. Nakonec se očekává, že z honeypotu bude možné extrahovat seznam použitých hesel.

3. **Průnik na ssh server**: Útočník se bude připojovat pomocí správného uhádnutého hesla. Předpokládá se, že se připojení úspěšně podaří a zároveň, že se útočník ocitne uvnitř kontejneru. Očekává se, že se útočník bude moct nerušeně pohybovat po kontejneru. Nakonec se očekává, že provedený útok bude možno přehrát přesně tak, jak jej viděl útočník.

5.3 Popis laboratorního prostředí

Pro vytvoření laboratorního prostředí, které sloužilo pro vývoj a následné testování jsem využil nástroj GNS3. Tento nástroj umožňuje snadné vytvoření sítě, podobně jako například výukový nástroj Cisco Packet Tracer. Rozdíl oproti Cisco Packet Traceru je však možnost do sítě začlenit skutečná síťová zařízení. GNS3 tedy dovoluje vytvoření virtuální sítě, do níž lze začlenit skutečné fyzické rozhraní zařízení, na kterém běží. Dále dovoluje jako koncová zařízení použít také mimo jiné docker kontejnery. Díky těmto možnostem jsem byl schopen vytvořit jednoduchou síť, obsahující tři docker kontejnery. Na této síti je také připojeno síťové rozhraní mého notebooku a veškeré směrování v síti řídí směrovací daemon BIRD. Tuto síť si lze prohlédnout na obrázku 5.1. Horní část obrázku reprezentuje část sítě mimo CESNET. Součástí této části je kontejner patřící útočníkovi. Pod útočníkem se nachází kontejner se suricatou. Tento kontejner reprezentuje vstup do CESNET sítě. Ve většině experimentů na tomto kontejneru vedle suricaty poběží také Odpovídač a Přesměrovávač. Pod kontejnerem se suricatou lze vidět router. Tento router v laboratorní síti reprezentuje libovolnou posloupnost routerů a dalších síťových zařízení mezi vstupem do CESNET sítě a obětí útoku. Dále tento router rozděluje laboratorní síť na tři /24 podsítě. Pod routerem vlevo je kontejner oběti. Pod routerem vpravo je připojeno rozhraní hostitelského zařízení. V laboratorním prostředí se hostitelské zařízení chová stejně jako jakékoliv další koncové zařízení a při některých experimentech na něm bude spuštěn honeypot a bude sloužit jako cíl přesměrovaného útoku.



Obrázek 5.1: **Laboratorní síť**

Tabulka 5.1: Tabulka s informacemi o laboratorní síti

Role zařízení	**Provedení**	**IP**
Útočník	kontejner	10.1.2.2
Průchozí zařízení	kontejner	10.1.2.11, 10.1.2.12
Oběť	kontejner	10.1.1.2
Obránce	hostitelské zařízení	10.1.0.2

5.4 Konfigurace komponent

Tato sekce obsahuje nastavení jednotlivých komponent použitých při experimentech. Většina konfigurace komponent je pro všechny experimenty stejná. Případné odlišnosti budou uvedeny u jednotlivých experimentů

- **Honeypot**:

```
{
  "logs": "file,terminal",
  "logs_location": "/tmp/honeypots",
  "honeypots": {
    "ssh": {
      "port": 22,
      "username": "ssh",
      "password": "hhh",
      "log_file_name": "ssh.log",
      "max_bytes": 10000,
      "backup_count": 10,
      "docker_image": "docker.io/vyzigold/kali",
      "docker_socket_path": "unix://tmp/podman.sock",
      "options":["capture_commands", "docker"]
    }
  }
}
```

Výpis 5.3: Konfigurace honeypotu

- **Aktivní reakce**:

```
interface: "eth1"
ip: "10.1.2.11"
debug: False
reaction_duration: 60000
suricata_rules_filename: reaction.rules
filter:
  "
  "

reroute_targets:
  honeypot:
    ip: "127.0.0.1"
    mac: "00:00:00:00:00:00"
    interface: "lo"

behavior:
  filtered: ["12", "401-410"]
  opened: ["1", "4-10", "80", "100-400"]
  closed: ["3", "81", "411-500"]
  honeypot: ["20-30"]
  default: "closed"

input:
  warden:
    client_config:
      url: 'https://warden.example.com/warden3'
      keyfile: '/opt/warden3/etc/key.pem'
      certfile: '/opt/warden3/etc/cert.pem'
      cafile: '/opt/warden3/etc/tcs-ca-bundle.pem'
      timeout: 10
      errlog_level: "debug"
      filelog_level: "debug"
      idstore: "MyClient.id"
      name: "cz.example.warden.test"
    event_filter:
      categories: ["Recon.Scanning", "Attempt.Login"]
      no_categories: []
      tag: []
      no_tag: []
      group: []
      no_group: []
    polling_interval: 1
    match: ["source", "target"]
```

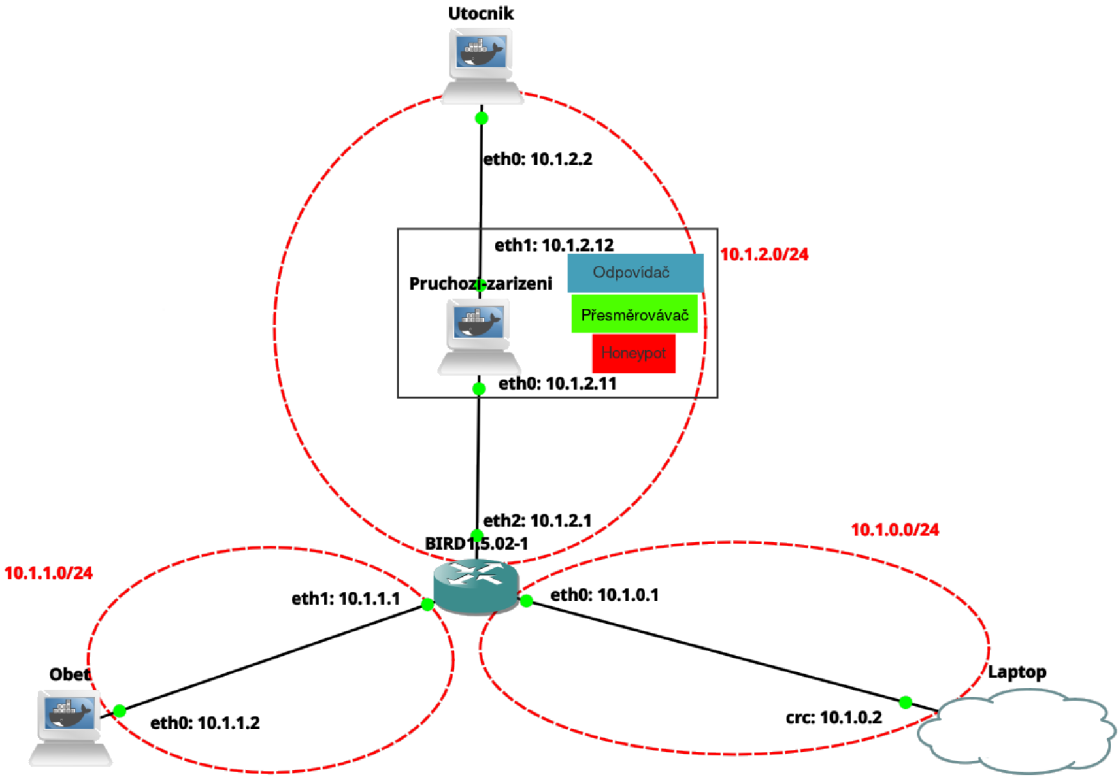
Výpis 5.4: Konfigurace aktivní reakce

5.5 Experimenty

V této sekci popíšu detaily jednotlivých experimentů a jejich výsledky, které dále porovnám s očekávanými výsledky ze začátku této kapitoly.

5.5.1 Všechny komponenty na stejném zařízení

V prvním experimentu otestuji nejjednodušší nasazení této práce. Veškeré komponenty budou nasazeny na průchozím zařízení se suricatou, budou tedy nejblíže útočníkovi. Díky tomu, že jsou všechny komponenty na stejném zařízení, není potřeba přesměrovávat útok skrz celou síť. Přesměrovávač tedy bude nastaven tak, aby útoky přesměroval na localhost, kde bude naslouchat honeypot. Další výhodou je, že jelikož je práce umístěna v cestě útoku, lze zablokovat komunikaci směřující k oběti. Umístění jednotlivých komponent si lze prohlédnout na obrázku 5.2. Konfigurace komponent při tomto experimentu byla shodná s konfigurací uvedenou výše.



Obrázek 5.2: **Laboratorní síť**

Výsledeky experimentu

1. **Skenování**: Reakce proběhla přesně podle očekávání popsaných výše. V situaci, kdy byla reakce započata v průběhu skenu, si nmap, ani deepscan ničeho nevšimli a pokračovali dále. Každý následný sken poté končil s výsledky shodnými s konfigurací aktivní reakce.

2. **Hádání hesla**: V této části reakce rovněž proběhla dle očekávání. V průběhu útoku byl útok přesměrován na honeypot. Hydra si ničeho nevšimla a dále pokračovala v útoku. Honeypot správně zapisoval přihlašovací údaje do terminálu a do nakonfigurovaného souboru. Pomocí jednořádkového bash skriptu bylo možné z tohoto souboru extrahovat seznam použitých hesel.

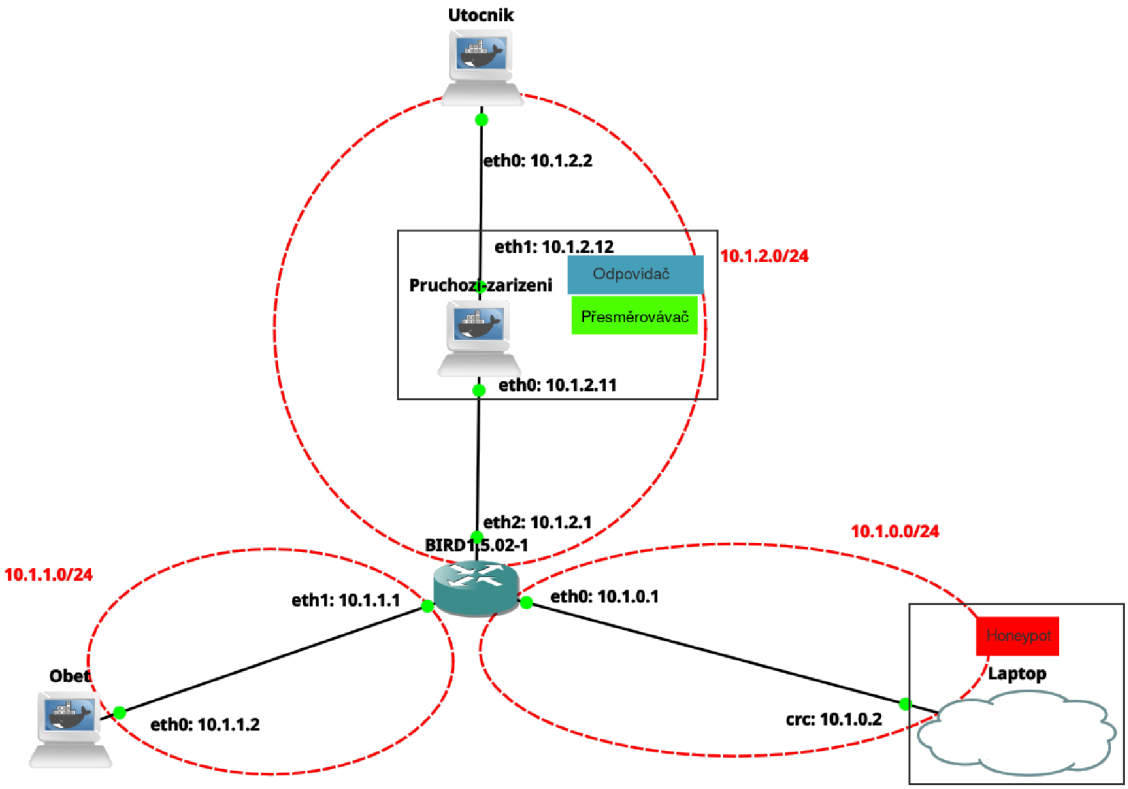
3. **Průnik na ssh server**: Tato fáze rovněž dopadla podle očekávání. Po zadání správných přístupových údajů bylo možné se přihlásit na ssh server, což byl ve skutečnosti docker kontejner, do kterého útočníka přesměroval honeypot. Veškerý pohyb útočníka v kontejneru byl zaznamenán a následně jej bylo možno úspěšně přehrát.

5.5.2 Honeypot na zařízení mimo původní cesty útoku

Ve druhém experimentu otestuji druhou možnost nasazení této práce a to, že honeypot bude oddělený od zbytku komponent na samostatném zařízení na síti. Díky tomu lze rozložit zátěž na více zařízení, protože různé druhy honeypotů se mohou nacházet na různých zařízeních v síti. Odpovídač a Přesměrovávač se tedy budou nacházet na zařízení se suricatou. Honeypot se bude nacházet na zařízení obránce, tedy na hostitelském zařízení. Přesměrovávač bude tudíž nakonfigurován tak, aby útoky přesměrovával na zařízení obránce, kde na ně bude honeypot odpovídat. Umístění jednotlivých komponent si lze prohlédnout na obrázku 5.3. Konfigurace aktivní reakce se v tomto experimentu z důvodu odlišného umístění honeypotu liší. Veškeré odlišnosti jsou uvedeny v 5.5

```
reroute_targets:
  honeypot:
    ip: "10.1.0.2" # laptop
    mac: "0c:21:3c:ad:00:02"
    interface: "eth0"
```

Výpis 5.5: Rozdíly v konfiguraci aktivní reakce



Obrázek 5.3: **Laboratorní síť**

Výsledeky experimentu

1. **Skenování**: U odpovídače nedošlo oproti předchozímu experimentu k žádné změně. Reakce tedy opět proběhla správně podle očekávání výše.

2. **Hádání hesla**: Přesměrovávač byl správně schopen přesměrovat útok na zařízení obránce, kde honeypot opět zaznamenal veškerá použitá hesla. Stejně jako u předchozího experimentu si Hydra ničeho nevšimla a pokračovala v útoku i po přesměrování. Reakce tedy dopadla správně podle očekávání popsaných výše.

3. **Průnik na ssh server**: Jelikož předchozí fáze experimentu potvrdila, že přesměrovávač funguje správně i pro tuto variantu nasazení, tak není překvapením, že i tato reakce proběhla úspěšně. Útočník se úspěšně přihlásil na honeypot, celý útok byl zaznamenán a úspěšně přehrán.

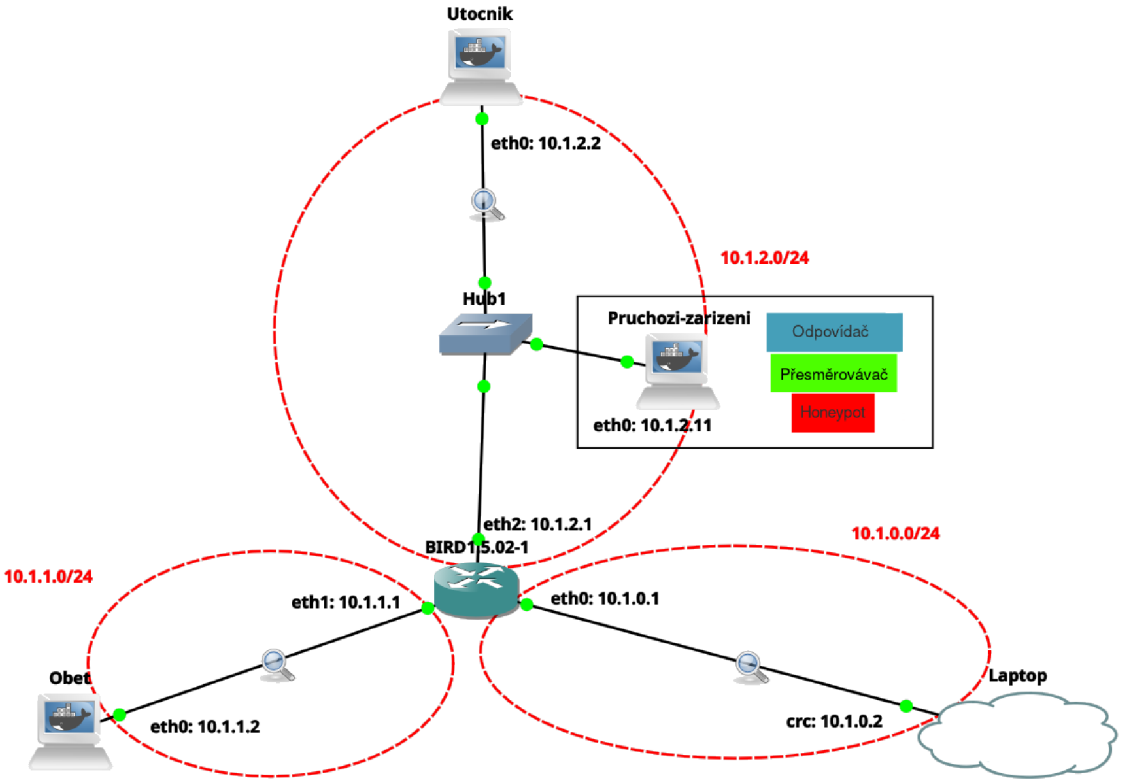
5.5.3 Všechny komponenty na zařízení mimo cesty útoku

Poslední experiment bude pokus o nasazení všech komponent této práce mimo přímou cestu útoku. Doposud se Odpovídač a Přesměrovávač vždy nacházeli přímo v cestě útoku, což jim dovolovalo útoku naslouchat a reagovat na něj. Nyní bude otestována situace, kdy veškeré komponenty sdílí stejné zařízení, které je mimo cestu útoku, avšak toto zařízení je schopno vidět veškerou komunikaci útočníka. V praxi lze stavu, že nějaké zařízení vidí tuto komunikaci, dosáhnout různě. V laboratorním prostředí však tento stav bude dosáhnut tak, že místo průchozího zařízení použijeme hub, který veškerou komunikaci zachycenou na některém jeho rozhraní odešle na všechna ostatní rozhraní. Díky tomu můžeme na třetí rozhraní hubu přidat zcela nové zařízení, které však uvidí veškerou komunikaci, která hubem prochází. Umístění jednotlivých komponent si lze prohlédnout na obrázku 5.4. Konfigurace komponent je shodná s konfigurací uvedenou výše.

Na rozdíl od předchozích experimentů, kdy bylo očekáváno, že všechny komponenty budou fungovat podle očekávání, v tomto experimentu není očekávána 100% úspěšnost. Vytvořené řešení nebylo navrženo pro tuto formu nasazení, jedná se tedy hlavně o zjištění, jak se budou jednotlivé komponenty v této situaci chovat, zda se podaří dosáhnout alespoň nějakých výsledků a také jakým způsobem tuto práci v budoucnu vylepšit tak, aby v tomto experimentu dosáhla lepších výsledků.

Výsledeky experimentu

1. **Skenování**: Při reakci na skenování byly dosaženy různé výsledky. Na pokus o sken totiž najednou odpovídaly obě zařízení, tedy oběť i zařízení, na kterém byla nasazena tato práce. Jak nmap, tak i masscan považovali za výsledek skenu tu odpověď, která přišla první. Úspěšnost aktivní reakce tedy závisela na rychlosti oběti a zařízení provádějícího aktivní reakci. Někdy dokonce jeden pokus o sken více portů obsahoval mix výsledků z obou odpovídajících zařízení. Byly tedy provedeny další pokusy s nmapem, jejichž výsledky zachycuje tabulka 5.2. Originální oběť byla s odpověďmi bohužel většinou rychlejší, avšak občas se podařilo získat mix odpovědí z oběti a z Odpovídače. Pokud by se podařilo Odpovídač zrychlit, pravděpodobně by bylo možno dosáhnout lepších výsledků. Další možností jak zlepšit úspěšnost by bylo přiblížit Odpovídač v síti blíže k útočníkovi. V případech, kdy nmap zachytil pouze odpovědi oběti bylo zjištěno, že oběť odpovídá asi o 0,65 ms rychleji než Odpovídač. Tento časový údaj



Obrázek 5.4: **Laboratorní síť**

Tabulka 5.2: Tabulka výsledky opakovaného skenování

	Zdroj zachycených odpovědí		
Celkový počet skenů	**Oběť**	**Odpovídač**	**Mix zdrojů**
10	8	0	2

zohledňuje také tzv. round-trip time, tedy čas, který packety potřebují pro cestu mezi útočníkem a cílovým zařízením.

2. **Útok na ssh**: V dalších fázích experimentu se prováděl útok na ssh server. Při tomto útoku dochází k navázání TCP spojení mezi útočníkem a obětí, případně mezi útočníkem a honeypotem. Toto spojení je navázáno vždy pouze mezi dvěma zařízeními a není jednoduché, do této komunikace vstupovat třetím zařízením. Tato fáze útoku, ať už při použití Hydry, nebo při použití klasického ssh klienta, tedy probíhala následovně:

   - Útočník se pokouší navázat spojení
   - Obě zařízení mu kladně odpoví
   - Útočník potvrdí navázání spojení s jedním zařízením
   - Druhé zařízení vidí, že je s potvrzením spojení něco v nepořádku a posílá RST packet a tím právě navázané spojení mezi útočníkem a prvním zařízením ukončí.

Výsledkem tedy bylo, že útočník nedokázal komunikovat ani s jedním z ssh serverů.

5.6 Zhodnocení výsledků a budoucí vývoj

Implementované řešení plně uspělo v prvních dvou experimentech, které testovaly původní návrh reakce, tak jak byla popsána v 3.1.8. Ve třetím experimentu, který testoval rozšíření této reakce, které bylo popsáno v 3.1.9 se bohužel jednalo pouze o částečný úspěch. Při reakci na skenování záleželo na rychlosti reakce v porovnání s rychlostí odpovědí oběti. Při reakci na útok na konkrétní síťovou službu současná komunikace z oběti i z honeypotu způsobila, že se útočník nemohl připojit ani k honeypotu, ani k oběti.

V budoucnu lze projekt vyvíjet v několika směrech:

- Zrychlit odpovídač tak, aby byl mnohem častěji rychlejší než oběť a tím zvýšit jeho úspěšnost v situacích podobných třetímu experimentu
- Přidat přesměrování do docker kontejneru i do dalších služeb honeypotu.
- Rozšířit odpovídač tak, aby reagoval i na sken udp portů.

Kapitola 6

Závěr

Cílem této diplomové práce bylo nastudování problematiky síťových útoků, zejména možností aktivní reakce na útoky a následná implementace vybraných reakcí tak, aby je bylo možno použít v síti CESNET.

Dle zadání tato práce měla zahrnovat nastudování oblasti útoků vedených po síti, taxonomii útoků a různé případy, kdy proti útoku byla vedena aktivní obrana. Na začátku práce lze tedy nalézt popis současného stavu síťové bezpečnosti jak z pohledu útočníka 2.4, tak z pohledu obránce 2.3 a to především se zaměřením na aktivní reakci na probíhající síťový útok. Dále tato práce popsala dvě různé možnosti členění útoku podle fáze, ve které se útok právě nachází 2.1, 2.1.2. Možný způsob aktivní reakce na druhy síťových útoků podle MITRE ATT&CK lze nalézt v sekci 2.5.

Dalším bodem zadání byla povinnost seznámit se s bezpečnostními nástroji používanými v síti CESNET, jejichž popis lze nalézt v sekci 2.6. Nástroj warden a částečně také některý detektor útoku, například Nemea jsou dále použity pro implementaci aktivní reakce v pozdějších kapitolách.

Dále bylo podle zadání potřeba zvážit, které aktivní reakce by bylo možné automatizovat a také bylo vyžadováno navrhnout jejich implementaci do sítě CESNET. Tomuto bodu zadání se věnuje kapitola 3 Návrh řešení. V této kapitole bylo navrženo několik různých způsobů aktivní reakce. V pozdějších sekcích byly tyto reakce detailně analyzovány a srovnány. Nakonec byla vybrána nejvhodnější reakce a jedno její rozšíření. V poslední sekci této kapitoly 3.4 byl uveden detailní návrh implementace této vybrané reakce.

Posledními body zadání bylo navrhované řešení implementovat, otestovat a komentovat výsledky a případné možnosti pokračování projektu. O implementaci pojednává kapitola 4 Implementace. S implementovaným řešením byly následně v laboratorním prostředí provedeny tři experimenty, které se lišily ve způsobu nasazení projektu. O experimentech se lze dočíst v kapitole 5 Testování. Nakonec byly navrženy některé možnosti budoucího pokračování projektu. Některé z těchto možností byly zmíněny už v kapitole o implementaci 4, shrnutí těchto možností však lze nalézt v sekci 5.6.

Kromě tohoto textu je součástí práce také projekt, o kterém se v tomto textu píše. Tento projekt úspěšně dokáže komunikovat s wardenem, díky čemuž je schopen se dozvědět o právě probíhajících útocích. Po tom, co se tento projekt dozví o probíhajícím útoku, je schopen na něj aktivně reagovat. Tato reakce spočívá v odpovídání na skenování místo původně zamýšlené oběti. Tyto odpovědi jsou plně konfigurovatelné. Dále tento projekt zvládá přesměrování útoku na konkrétní službu z oběti na honeypot. Nakonec byl vybrán jeden honeypot, který byl rozšířen tak, aby zachycený útok na ssh server přesměroval do docker kontejneru, kde se může útočník připojit aniž by ohrožoval zařízení, na kterém honeypot

běží. To obránci umožňuje, aby útočníkovi podstrčil libovolná falešná data. Veškerý pohyb útočníka v kontejneru je také nahráván a byl vytvořen jednoduchý přehrávač tohoto útoku, který dovoluje jak výpis všech útočníkem zadaných příkazů, tak i přehrání průběhu celého útoku tak, jak jej viděl útočník.

Literatura

[1] Cejka, T., Bartos, V., Svepes, M., Rosa, Z. a Kubatova, H. NEMEA: A Framework for Network Traffic Analysis. In: *12th International Conference on Network and Service Management (CNSM 2016)*. 2016. DOI: 10.1109/CNSM.2016.7818417. Dostupné z: http://dx.doi.org/10.1109/CNSM.2016.7818417.

[2] Fortinet. *What are honeypots (Computing)?* [online]. 2022 [cit. 2022-11-01]. Dostupné z: https://www.fortinet.com/resources/cyberglossary/what-is-honeypot.

[3] Gordon Lyon. *Nmap: Discover your network* [online]. 2022 [cit. 2022-10-23]. Dostupné z: https://nmap.org/.

[4] Mitre Corporation. *CVE* [online]. 2022 [cit. 2022-11-20]. Dostupné z: https://www.cve.org/About/Overview#AbouttheCVEProgram.

[5] Mitre Corporation. *MITRE ATT&CK* [online]. 2022 [cit. 2022-11-20]. Dostupné z: https://attack.mitre.org/#.

[6] OffSec Services Limited. *The most advanced Penetration Testing Distribution* [online]. 2023 [cit. 2023-1-10]. Dostupné z: https://www.kali.org/.

[7] Pedro Marques, R. R. B. G. J. M. D. M. *Dissemination of Flow Specification Rules* [Internet Requests for Comments]. RFC 5575. RFC Editor, August 2009. 1-22 s. Dostupné z: https://www.rfc-editor.org/rfc/rfc5575.

[8] Python Software Foundation. *The Python Tutorial* [online]. 2023. Aktualizováno 13. 4. 2023 [cit. 2023-04-16]. Dostupné z: https://docs.python.org/3/tutorial/index.html.

[9] Robert Graham. *MASSCAN: Mass IP port scanner* [online]. 2021 [cit. 2022-10-24]. Dostupné z: https://github.com/robertdavidgraham/masscan.

[10] Scapy community. *What is Scapy?* [online]. 2023. Aktualizováno 16. 4. 2023 [cit. 2023-04-16]. Dostupné z: https://scapy.net/.

[11] Sdružení CESNET, z. s. p. o.. *Warden* [online]. 2017 [cit. 2023-1-10]. Dostupné z: https://warden.cesnet.cz/.

[12] Sdružení CESNET, z. s. p. o.. *Mentat* [online]. 2018 [cit. 2023-1-10]. Dostupné z: https://mentat.cesnet.cz/.

[13] Sdružení CESNET, z. s. p. o.. *O nás* [online]. 2022 [cit. 2023-1-10]. Dostupné z: https://www.cesnet.cz/sdruzeni/.

[14] The Open Information Security Foundation. *Features* [online]. 2023 [cit. 2023-04-20]. Dostupné z: https://suricata.io/features/.

[15] Tom Liston. *Tom Liston talks about LaBrea* [online]. [cit. 2023-1-14]. Dostupné z: https://labrea.sourceforge.io/Intro-History.html.

[16] Tzu, S. *The art of war*. Lulu.com, 2020. ISBN 9781678000097. Dostupné z: https://books.google.cz/books?id=K6vWDwAAQBAJ.

[17] Vamsidhar Valluri, G. S. B. C. *Cisco Systems NetFlow Services Export Version 9* [Internet Requests for Comments]. RFC 3954. RFC Editor, October 2004. 1-33 s. Dostupné z: https://www.rfc-editor.org/rfc/rfc3954.

[18] van Houser. *H Y D R A* [online]. 2022 [cit. 2022-10-25]. Dostupné z: https://github.com/vanhauser-thc/thc-hydra.

[19] Václav Bartoš. *Network Entity Reputation Database (NERD)* [online]. 2022 [cit. 2023-1-10]. Dostupné z: https://nerd.cesnet.cz/.

[20] Yadav, T. a Rao, A. M. Technical Aspects of Cyber Kill Chain. In: Abawajy, J. H., Mukherjea, S., Thampi, S. M. a Ruiz Martínez, A., ed. *Security in Computing and Communications*. Cham: Springer International Publishing, 2015, s. 438–452. ISBN 978-3-319-22915-7.

[21] Yakov Rekhter, S. H. *A Border Gateway Protocol 4* [Internet Requests for Comments]. RFC 4271. RFC Editor, January 2006. 1-104 s. Dostupné z: https://www.rfc-editor.org/rfc/rfc4271.