VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

BAKALÁŘSKÁ PRÁCE

Brno, 2021 Luděk Huška

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

HRA KYBERNETICKÉ BEZPEČNOSTI NA PLATFORMĚ OPENSTACK
CYBERSECURITY GAME IN OPENSTACK PLATFORM

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR
Luděk Huška

VEDOUCÍ PRÁCE
SUPERVISOR
Ing. Tomáš Stodůlka

BRNO 2021

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní program **Informační bezpečnost**

Ústav telekomunikací

Student: Luděk Huška ***ID:*** 203704

Ročník: 3 ***Akademický rok:*** 2020/21

NÁZEV TÉMATU:

Hra kybernetické bezpečnosti na platformě OpenStack

POKYNY PRO VYPRACOVÁNÍ:

Prostudujte a zpracujte problematiku bezpečnostního testování. Na základě nabytých znalostí navrhněte komplexní scénář simulující menší podnikovou síť, která se bude skládat ze serverů/klientů Linux/Windows, směrovačů, několika sítí a firewallu. V rámci praktické části práce realizujte navržený scénář kybernetické hry prostřednictvím platformy OpenStack, kde student dostane k dispozici nový hardware s běžící platformou. Výsledkem bakalářské práce bude návrh a implementace bezpečnostní hry s vypracovaným návodem, který bude studenty provádět hrou, případně jim podá nápovědy ke splnění dané části hry. Celková doba hraní hry bude v rozmezí 75 až 90 minut. Na závěr práce analyzujte a přehledně zobrazte průběžnou náročnost hardwarových zdrojů (RAM, CPU, disk) hostujícího serveru při hraní kybernetické hry.

DOPORUČENÁ LITERATURA:

[1] GUPTA, B. B. (ed.). Computer and cyber security: principles, algorithm, applications, and perspectives. CRC Press, 2018.

[2] CARDWELL, K. Building virtual pentesting labs for advanced penetration testing. Packt Publishing Ltd, 2014.

Termín zadání: 1.2.2021 ***Termín odevzdání:*** 31.5.2021

Vedoucí práce: Ing. Tomáš Stodůlka

doc. Ing. Jan Hajný, Ph.D.
předseda rady studijního programu

ABSTRAKT

Bakalářská práce se zabývá tvorbou kybernetické hry na platformě OpenStack. Práce začíná vysvětlením potřebných teoretických znalostí jako: informační, kybernetická a počítačová bezpečnost, bezpečnostní hrozby a bezpečnostní testování. V praktické části je navržena a vytvořena kybernetická hra se zaměřením na exploitace a základní bezpečnostní rizika. V další části je analyzována průběžná náročnost hardwarových zdrojů (RAM, CPU, disk). Na závěr je vypracován návod pro laboratorní cvičení, který je ve dvou vyhotoveních a to pro žáky a cvičícího.

KLÍČOVÁ SLOVA

OpenStack, virtualizace, virtuální stroj, kybernetická hra, CTF, metasploit, NMAP

ABSTRACT

The bachelor thesis is about creation of the cybernetic game on the OpenStack platform. The work starts with an explanation of necessary knowledge like: Informational cybernetic and computer security, security threats, and testing of the overall security. In the practical part, there is created a cyber game that focus on exploitation and basic security vulnerabilities. The next part involves analysis ongoing complexity of hardware resources like RAM, CPU, disc, etc. In the end, there is a guideline for a lab exercise, which is made for students and professors or teachers both separated.

KEYWORDS

OpenStack, virtualization, virtual machine„ cyber game, CTF, metasploit, NMAP

HUŠKA, Luděk. *Hra kybernetické bezpečnosti na platformě OpenStack*. Brno, 2021, 66 s. Bakalářská práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Vedoucí práce: Ing. Tomáš Stodůlka

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „Hra kybernetické bezpečnosti na platformě OpenStack“ jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora

PODĚKOVÁNÍ

Velmi rád bych poděkoval vedoucímu bakalářské práce panu Ing.Tomáši Stodůlkovi, za odborné vedení, vstřícnost, trpělivost a podnětné návrhy k práci. Taktéž bych chtěl poděkovat nejbližší rodině za podporu, během mého studia.

Obsah

Úvod **9**

1 Teoretická část práce **10**
1.1 Informační, kybernetická a počítačová bezpečnost 10
1.2 Kybernetické hrozby v České republice 11
1.3 Kybernetické hrozby ve světě a OWASP 13
1.4 Bezpečnostní testy 17
1.4.1 Bezpečnostní analýza 17
1.4.2 Sken zranitelností 18
1.4.3 Penetrační testování 18
1.4.4 Bezpečnostní audit 19
1.4.5 Etické hackování 19
1.4.6 Řízení rizik 20
1.5 Bezpečnostní hry 20
1.6 Možnosti využití OpenStacku 22

2 Návrh kybernetické hry **25**
2.1 Informace o projektu v OpenStacku 25
2.2 Návrh topologie kybernetické hry 25
2.3 Návrh jednotlivých částí kybernetické hry 26
2.3.1 Fedora WEB 27
2.3.2 Ubuntu VPN 27
2.3.3 Windows SMB 28
2.3.4 Windows phpMyAdmin 28
2.3.5 Admin-PC 28

3 Tvorba kybernetické hry **30**
3.1 Tvorba v prostředí Openstack 30
3.1.1 Vytváření sítí 30
3.1.2 Tvorba instance 32
3.2 Tvorba serverů a její problémy 34
3.2.1 Fedora WEB 34
3.2.2 Ubuntu VPN 36
3.2.3 Windows SMB 38
3.2.4 Windows PHPMYADMIN 40
3.2.5 Admin PC 42

4 Měření HW náročnosti kybernetické hry **43**
4.1 Testování kybernetické hry studentem 43
4.2 Měření HW při hraní kybernetické hry 44
4.3 Měření hardwarové náročnosti serverů 46
4.4 Shrnutí měření 49

Závěr **50**

Literatura **51**

Seznam symbolů, veličin a zkratek **54**

A Hra kybernetické bezpečnosti na platformě OpenStack - verze pro studenty **56**
A.1 Teoretický úvod 56
A.1.1 Bezpečnostní zranitelnosti a exploitace 56
A.1.2 NMAP 58
A.1.3 Metasploit 59
A.2 Postup řešení 62
A.2.1 Potřebné nástroje. 62
A.2.2 Nastavení VPN serveru 62
A.2.3 Metodika řešení 62
A.2.4 Postup úlohou 63
A.3 Seznam zkratek 65

Literatura **66**

Seznam obrázků

1.1 Vztah mezi informační, kybernetickou a počítačovou bezpečností . . . 11
1.2 Graf počtu kybernetických případů v ČR (data převzaty z [7]) 12
1.3 Typy hrozeb u bezpečnostní analýzy rizik 17
1.4 Značení týmů v bezpečnostních hrách 21
2.1 Topologie sítě bezpečnostní hry 26
3.1 Formulář pro vytvoření nové sítě 30
3.2 Formulář pro vytvoření nové sítě 31
3.3 Import obrazu disku do OpenStack systému 32
3.4 Vytvoření nové instance v OpenStack systému 33
3.5 Nový vzhled webové stránky 35
3.6 Instalace OpenVPN 37
3.7 Správce Windows 2012 serveru 39
3.8 Nové sdílení 39
3.9 Administrace prostředí phpMyAdmin 41
4.1 Využití RAM při hraní kybernetické hry 44
4.2 Využití CPU při hraní kybernetické hry 44
4.3 Využití CPU v klidovém stavu serveru 45
4.4 Využití disku při hraní kybernetické hry 45
4.5 Síťový provoz při hraní kybernetické hry 46
4.6 Vytíženost RAM při uspávání instancí 47
4.7 Využití CPU při uspávání instancí 47
4.8 Využití disku při uspávání instancí 48
4.9 Vytížení sítě při uspávání instancí 48
A.1 Moduly metasploitu 60
A.2 Vyhledání exploitu pomocí search 60
A.3 Aktivování exploitu pomocí příkazu use 61
A.4 Zobrazení parametrů exploitu 61
A.5 Nastavení parametrů exploitu 61
A.6 Nastavení síťového rozhraní virtuálního stroje 62
A.7 Topologie sítě 63

Úvod

Informační bezpečnost klade na firmy a instituce v posledních letech čím dál větší požadavky. Kybernetické útoky, krádeže osobních údajů, účtů, financí a mnoho dalšího jsou problémy se kterými se potýkají téměř na denní bázi. To může vést k obecné ztrátě důvěry vůči firmě a dokonce i k finančním ztrátám. V určitých případech může být firma taktéž za tyto selhání souzena. Jediná účinná praktika jak těmto rizikům zabránit je, za pomocí bezpečnostních postupů, které vyžadují znalé osoby v dané oblasti. Těchto lidí je však v dnešní době velice málo a zaškolování nových je časově i finančně náročné.

Cílem této bakalářské práce je vysvětlit jak důležitou roli hraje v dnešní době bezpečnost a že jeden z nejlepších způsobů jak se v této problematice vzdělávat je pomocí bezpečnostních her. Jejím úkolem je taktéž uvést do dané problematiky a poskytnout vzhledem k ní širší rozhled, tak aby bylo jednodušší se v ní zorientovat.

Bakalářská práce obsahuje teoretický úvod, kde jsou vysvětleny základy bezpečnosti. Jaké hrozby se vyskytují v České Republice a ve světě. Taktéž uvádí jak se proti těmto hrozbám bránit. Tato kapitola dále obsahuje vysvětlení důležitosti kybernetických her a úvod do OpenStack platformy.

Na tuto část navazuje kapitola, která se zabývá teoretickým návrhem kybernetických her. V té je představen projekt v OpenStacku, ve kterém bude kybernetická hra vytvořena. Taktéž obsahuje návrh topologie a jednotlivých částí kybernetické hry.

Třetí kapitola má za úkol popsat tvorbu jednotlivých částí kybernetické hry. Tvorbu v OpenStack systému a následně konfiguraci na jednotlivých serverech.

Čtvrtá kapitola je výstupem kybernetické hry. Nachází se zde testování hry studentem. Měření hardwarových zdrojů při jejím hraní a taktéž měření náročnosti serverů.

Jako poslední část bakalářské práce je vypracován návod pro řešení kybernetické hry. Tento návod je poskytnut ve dvou verzích a to pro studenta a cvičícího.

1 Teoretická část práce

Tato kapitola bakalářské práce se zabývá základními pojmy a znalostmi, které jsou nutné k porozumění tématiky této práce. Především vysvětluje rozdíly mezi počítačovou, kybernetickou a informační bezpečností. Dále pak, proč se danou tématikou zabývat. Zmiňuje jaké hrozby v této oblasti reálně hrozí a podrobněji vysvětluje tématiku bezpečnosti webových aplikací. V poslední části pak udává jak se těmto hrozbám bránit.

1.1 Informační, kybernetická a počítačová bezpečnost

K pochopení informační bezpečnosti je nutné si zadefinovat její koncepty. Lze je rozdělit do tří skupin [5]:

- **Důvěrnost** je zachována tehdy když mají k informacím přístup jen osoby určené. Je důležité data chránit proti přístupu osobou neoprávněnou. Tento bod si lze rodělit na dva koncepty.
  - **Důvěrnost údajů**, zajišťuje ať soukromé nebo důvěrné informace, nejsou dostupné neoprávněným osobám.
  - **Privátnost údajů**, udává jaké informace jsou shromažďovány a komu jsou dostupny.
- **Integrita** představuje uchování dat, aby nebyla nesprávně modifikována. Modifikace může být úmyslná, nebo vzniknout nějakým nedopatřením. Tento bod lze taktéž rozdělit na dva koncepty.
  - **Integrita údajů** zajišťuje, že s informacemi je nakládáno jen stanoveným způsobem.
  - **Integrita systému** zajišťuje, aby systém vykonával svou funkci nenarušeným způsobem.
- **Dostupnost** zaručuje ať jsou informace přístupny těm, kteří mají příslušné oprávnění.

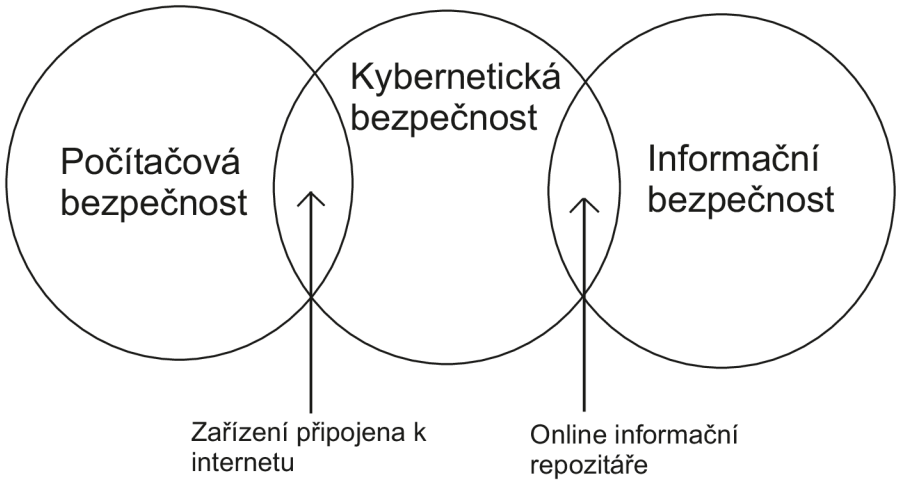
Rozdíly mezi kybernetickou a informační bezpečností

Kybernetickou bezpečnost lze definovat jako soubor technologií, procesů a postupů určených k ochraně sítí, zařízení, programů a dat před útoky, poškozením, malwarem nebo neoprávněným přístupem [4]. Z toho plyne, že informační bezpečnost je oblast, která se věnuje zabezpečení všech informací. Je to soubor pravidel a cílů, které je nutno dodržet. Zatímco kybernetická bezpečnost je z části pod odvětvím informační bezpečnosti, jelikož má na starost ochranu digitálních informací. Je však

doplněna o další problematiky jako například bezpečnost sítí, aplikací, infrastruktur a mobilní bezpečnosti [5, 6]. Vzájemný vztah lze vyjádřit graficky dle obrázku 1.1.

Rozdíly mezi kybernetickou a počítačovou bezpečností

V tomto případě musí být vycházeno z předpokladu, že kybernetická bezpečnost se bude vztahovat k zařízením připojená k internetu. Zatímco počítačová bezpečnost může existovat od něj separátně. Tato bezpečnost musí zahrnovat zařízení jako tablety, chytré telefony, čtečky knih, chytré hodinky a mnoho dalších zařízení. Vzájemný vztah lze vyjádřit graficky dle obrázku 1.1.



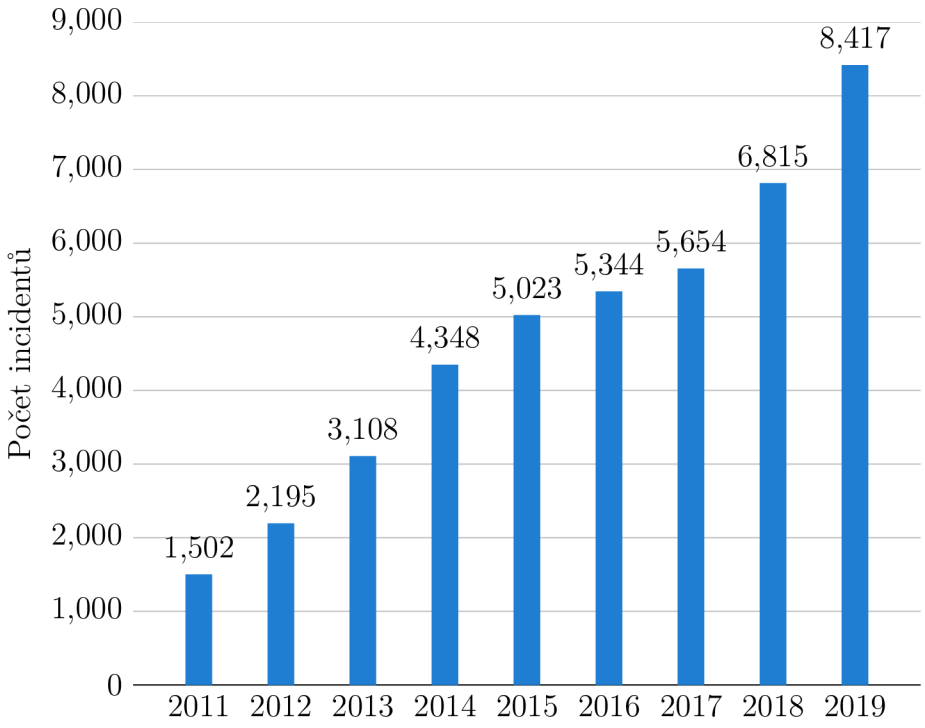
Obr. 1.1: Vztah mezi informační, kybernetickou a počítačovou bezpečností

1.2 Kybernetické hrozby v České republice

Po nabytí vědomostí ohledně typů bezpečnostní, je možné tyto znalosti aplikovat na reálné případy, kdy došlo k jejich překonání. Pro získání většího povědomí o těchto hrozbách a jejích typech, bude prozkoumaná výroční zprávu národního úřadu pro kybernetickou a informační bezpečnost České republiky. S hlavním zaměřením na počet vyšetřovaných kyberkriminálních případů v průběhu posledních několika let a jejich typům.

Z přiloženého grafu 1.2 vyplývá, že počet incidentů s léty dramaticky přibývá. Zatímco v roce 2011 bylo zaznamenáno 1502 případů, v roce 2019 se tento počet navýšil až na 8417 (tj. nárůst zhruba o 5,6násobek) S počtem útoků roste taktéž

sofistikovanost a jejich cílenost. Tyto útoky jsou blíže vysvětleny v následujících podkapitolách [7].



Obr. 1.2: Graf počtu kybernetických případů v ČR (data převzaty z [7])

Phisingové útoky

Phisingové útoky jsou podvodné či klamavé jednání, kdy se útočník snaží vylákat z oběti osobní informace. Uživatelské jméno, heslo, číslo kreditní karty a podobně. Toho nejčastěji dosáhne tak, že v uživateli vzbudí důvěru, sníží jeho ostražitost, či ho donutí akceptovat scénář, který si předem připravil.

Phisingový útok může vypadat, například jako odkaz na podvrženou webovou stránku. Ta může být vzhledově k nerozeznání od té reálné, na kterou jsme zvyklí. Jelikož tato stránka je pod kontrolou útočníka, tak při vyplnění a odeslání údajů, tyto informace získá.

Nejrozšířenější formou phisingu jsou však podvodné e-maily. Tento e-mail může mít různé formáty, může vypadat například jako požadavek podepsaný známou institucí, jenž má za cíl vylákat z uživatele osobní informace. Další možnosti tohoto e-mailu můžou vypadat jako: výhry v soutěži, dědictví po cizí osobě a podobně [7, 8].

Útoky DoS a DDoS

DoS (Denial of Service) má za cíl odepřít dostupnost vybrané služby. Toho docílí tím, že službu zahltí TCP (Transmission Control Protocol) nebo UDP (User Datagram Protocol) pakety. Jako příklad tohoto útoku můžeme zmínit ping smrti, kde útočník zašle podvržený ping dotaz o velikosti větší než 65 535 bajtů. Jelikož tato velikost je nad rámec maximální propustnosti, může dojít k pádu cílového systému. Za **DDoS** (Distributed Denial of Service) útok se považuje, když vícero uzlů útočí na systém pomocí DoS. Cíl je pak napaden z mnoha stran. Taktéž DDoS útoků existuje mnoho typů, například Reset Flood, Syn Flood, PingSweep a další [9].

Spam

Spam je nevyžádaná elektronická zpráva, která může obsahovat škodlivý kód, trojské koně a další. Nejčastější však bývá ve formě reklamního, či jiného nevyžádaného sdělení [7, 8].

Škodlivý kód

Neboli malware, je jakýkoliv software[1], který byl vytvořen se záměrem uškodit. Šíření tohoto kódu, bývá nejčastěji pomocí pochybných stránek, e-mailů, ale lze jej objevit i na USB klíčenkách, nenápadně rozmístěných kolem objektu, který je středem pozornosti útočníka. Malware dělíme do mnoha kategorií, jako ty základní můžeme uvést, adware, spyware, viry, červy a trojské koně [7, 8].

1.3 Kybernetické hrozby ve světě a OWASP

Tato pasáž si klade za cíl vysvětlit další kybernetické hrozby, které jsou považovány ve světě za nejvíc časté. Vycházeno bude, z nejčastějších deseti webových zranitelností podle neziskové organizace OWASP (Open Web Application Security Project). Následující kapitoly popisují nejaktuálnější zranitelnosti v roce 2020 na základě [10].

Injekce

Je nejčastější typ útoku na webovou aplikaci. Využívá vložení škodlivého kódu, přesněji databázového dotazu, který je pak zaslán interpretu[2]. Interpret může tento dotaz přeložit nesprávně a může tak dojít k úniku, či ztrátě dat, popřípadě ke spuštění nežádoucího kódu.

[1] Pojem, který zahrnuje nejen počítačový program, ale i databáze, multimédia a podobně.

[2] Program, který překládá a spouští zdrojový kód řádek po řádku.

V praxi pak tato zranitelnost může vypadat následovně. Předpokládejme standardní přihlašovací formulář požadující od uživatele jméno a heslo. Tento formulář je naprogramovaný následovně. V popředí aplikace se nachází tento HTML[3] (HyperText Markup Language) kód.

```
<form action='index.php' method="post">
<input type="text" name="jmeno" required="required"/>
<input type="password" name="heslo"/>
<input type="submit" value="Submit"/>
</form>
```

Formulář má za úkol zaslat uživatelem zadané jméno a heslo do pozadí aplikace, PHP[4] (HyperText Markup Language) souboru, jenž obsahuje tento databázový dotaz.

```
SELECT * FROM uzivatele WHERE jmeno = $_POST['jmeno']
AND heslo = md5($_POST['heslo']);
```

Zde si můžeme povšimnout, kde nastává problém. Zadané jméno a heslo se totiž vkládá přímo do dotazu. Jelikož neproběhne kontrola, tak dotaz může útočník jakkoliv kompromitovat. V konečném důsledku takový kompromitovaný dotaz vypadá například následovně.

```
SELECT * FROM uzivatele WHERE jmeno = 'administrator'
AND heslo = '' or 1=1--'
```

V tomto případě zadal uživatel údaje jako jméno: `administrator` a heslo: `' or 1=1--`. Podmínka `or 1=1` bude vždy splněna a pokud takový dotaz nebude ošetřen, lze se přihlásit na jakéhokoliv uživatele, v této situaci jako administrátor. Ostatní znaky slouží k zachovaní správné syntaxi SQL, kde dvě pomlčky označují komentář.

Nefunkční autentizace

Nefunkční autentizace dává útočníkovy nové možnosti útoku jako slovníkové útoky, generování náhodných přihlašovacích údajů, či využití výchozích jmen a hesel. To může vyústit v přebrání uživatele nebo v horším případě celého systému. Správně implementovaná autentizace by měla splňovat minimálně jeden z těchto bodů:

- vícefaktorová autentizace,
- kontrola komplexity hesla,
- omezený počet na přihlášení.

Nezabezpečení citlivých dat

Nezabezpečení dat představuje v dnešní době nejen bezpečnostní problém, ale taktéž právní provinění. Tento fakt upravuje směrnice GDPR, která vzhledem k povaze

[3]Značkovací jazyk, používaný pro tvorbu webových stránek.

[4]Skriptovací jazyk pro tvorbu pozadí aplikace, její logiky.

dat stanovuje, jakému zabezpečení musí podléhat. Za prvky tohoto zabezpečení se považuje pseudonymizace, šifrování a podobně. Při nedostatečném zabezpečení, může útočník tato data změnit nebo zneužít k dalším útokům.

XML External Entities

Tento typ útoku patří mezi ty složitější, jelikož útočník musí vlastnit, či si vytvořit, vlastní webový server, na který si vloží upravený XML soubor. Útok pak spočívá v tom, že útočník nahradí relativní cestu k výchozímu XML souboru cestou absolutní, na jeho vytvořený server, kde má onen upravený soubor. Aplikace si přebere obsah od uživatele a zpracuje jej interním parserem[5].

Tato zranitelnost může být využita k přístupu k chráněným souborům, spuštění škodlivého kódu, odepření služeb, nebo spuštění na serveru libovolného příkazu operačního systému [11].

Nefunkční kontrola přístupu

Nefunkční kontrola přístupu má za následek, že i běžný uživatel může získat přístup, kde by mít neměl. Potenciální útočník může získat z běžného účtu přístup na účty jiných uživatelů, možnost zobrazovat jejich osobní složky a soubory. Taktéž může tyto data upravit, či využít k dalšímu útoku.

Chybná konfigurace

Zastaralé verze programů, doplňků, skriptů a podobně, mohou obsahovat známé zranitelnosti. Je nutné se dívat co služba, kterou poskytujeme, potřebuje a vyhnout se tak případným problémům.

Cross-site scripting (XSS)

Jedná se o jednu z nejvíc podceňovaných zranitelností a taktéž o tu nejvíc rozšířenou. XSS útok bývá vedený proti koncovým uživatelům. Útočník se zde snaží spustit kód JavaScriptu ve webovém prohlížeči své oběti. Skript musí být načten ze stejné domény jako data, ke kterým má získat přístup. Útočník ,usí tento script na doménu nahrát. Útok XSS lze rozdělit do následujících kategorií[12]:

- **Perzistentní XSS** – je jeden z nejjednodušších a nejsnazších k pochopení. Útočník v tomto případě nahraje script na server. Tento script může být například ve formě příspěvku na fóru, osobní zprávy, nebo kdekoliv, kde mají uživatelé možnost nahrát text. Tím útočník docílí toho, že při každém načtení stránky se kód spustí.

[5]Část programu, která má na starosti syntaktickou analýzu textu.

- **Reflektovaný XSS** – je nejčastějším typem těchto útoků. Oproti perzistentnímu typu, není uložen trvale na serveru. Při útoku se infikovaný skript pošle GET požadavkem na server a ten vložený kód zpracuje. Toho útočník docílí tak, že kód vloží do vyhledávání či pozmění URL a podobně.
- **DOM base XSS** – je podobný jako reflektovaný. Rozdíl je v tom že požadavek se neodesílá na server, ale vše funguje na straně klienta v prohlížeči.
- **Self XSS** – je v tomto případě script, který vkládá do konzole sám uživatel. Je spouštěn na straně prohlížeče. Útočník může za pomocí manipulace uživatele k tomuto přimět.
- **Blind XSS** – je případ, kdy útočník naslepo vkládá skripty například do databází, logů a čeká až bude tato informace otevřena druhou stranou a jestli se skript spustí.

Následkem XSS je spuštění škodlivého kódu. Tím může dojít k změně v kódu stránky, či databázi, ukradení relace, dat, cookies a dalších možných následků [12].

Nezabezpečená deserializace

K pochopení tohoto útoku je nutné si vyjasnit pojem serializace a deserializace. Serializace v informatice znamená konvertování objektu na posloupnost bytů, které pak lze uložit do databáze, souboru, či přenést do sítě. Naopak deserializace převádí proud bytů zpět na kopii objektu.

Tento útok vzniká, když uživatel může volně manipulovat se vstupy a upravovat tak serializovaná data. Když upravené data projdou nezabezpečeným deserializátorem, může to vést k řadě útoků jako například spuštění škodlivého kódu, injekcím, eskalaci práv a podobně.

Použití komponent se známými zranitelnostmi

Tento bod souvisí se zranitelností výše a to chybnou konfigurací. Komponenty, frameworky a doplňky třetích stran, mohou obsahovat zranitelnosti. Obzvláště pak ty neaktualizované, či špatně nastavené. Je důležité se jim proto nejlépe vyhnout, nebo používat jen ty nezbytně nutné, či ověřené komunitou.

Nedostatečné logování a monitorování

Tento poslední bod má za úkol spíše předcházet bezpečnostním incidentům. Poukazuje na důležitost monitorování sítě, vytváření logů, aktivit a notifikací na vzniklé události.

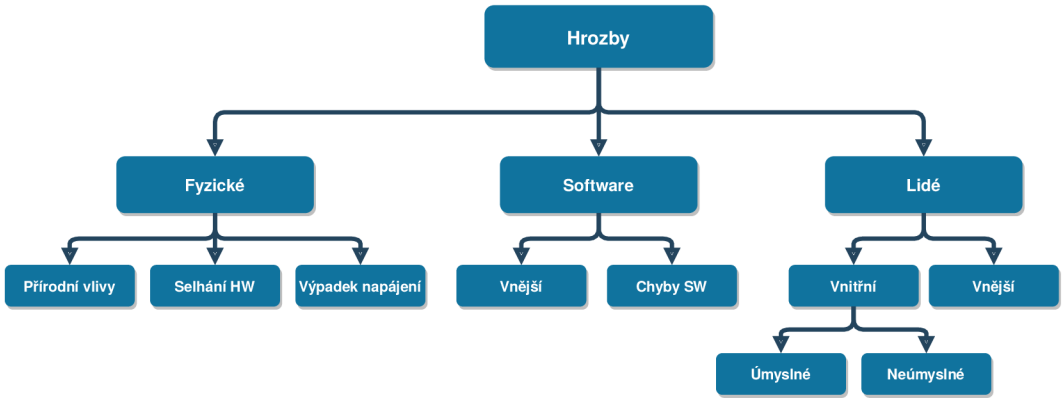
1.4 Bezpečnostní testy

Nejjednodušší způsob odhalení bezpečnostních mezer, které byly popsány v předešlých kapitolách je bezpečnostní testování. O tomto testování a jeho typech bude řečeno více v této kapitole. Budou vysvětleny základní typy bezpečnostního testování, metodiky a postupy.

Bezpečnostní testy jsou softwarové testy penetračního charakteru, které reálně simulují pokus o kompromitaci IT systému. Cílem je odhalit zranitelnosti, rizika systému a jaké škody by mohly nastat, nebo jaké informace by mohly uniknout. Výsledkem je také ověření funkce a efektivity aplikovaných bezpečnostních opatření [13, 14].

1.4.1 Bezpečnostní analýza

Bezpečnostní analýza má za úkol zajistit důvěrnost dat, jejich integritu a dostupnost. Zkoumá vnitřní zranitelnosti, vnější hrozby, a implementované ochranné mechanizmy. Nejdůležitějším cílem bezpečnostní analýzy je bezpochyby identifikace možné hrozby a potenciálního dopadu. Mimo jiné, má taktéž za úkol analyzovat stávající ochranné systémy, určovat jejich efektivitu a funkčnost. Popřípadě při jejich nedostatečnosti, navrhnout nové. Hrozby si můžeme rozdělit pomocí diagramu na obrázku 1.3 [15].



Obr. 1.3: Typy hrozeb u bezpečnostní analýzy rizik

Bezpečnostní analýzy se pak dělí do těchto pěti skupin:

- přehledová analýza rizik,
- detailní analýza rizik,
- technická bezpečnostní analýza,
- srovnávací analýza stavu,
- počítačová forenzní analýza incidentů.

Zmíněné analýzy výše mají chránit počítačové systémy, zařízení, datová aktiva, služby, aplikace, procesy, nebo informační systém organizace jako celek [16].

1.4.2 Sken zranitelností

Sken zranitelností je základním stavebním kamenem komplexní metodiky bezpečnostního testování. Získáme tak makroskopický obraz stavu zabezpečení. Výhodou je schopnost otestovat i již prakticky zapomenuté, avšak stále aktivní systémy, které často mohou útočníkovi otevřít dveře do privátní sítě. [17]

1.4.3 Penetrační testování

Simuluje reálné útoky, pro vyhodnocení zranitelností, slabin nebo nevhodných konfigurací s cílem potvrdit a případně odhalit další potencionální slabiny. Cílem tohoto testování je objevení zranitelnosti a vyhodnocení jak velkou hrozbu představuje a co by z toho mohl potenciální útočník získat [13]. Penetrační testy se dělí do několika skupin [18]:

- **Manuální testy** – provádí penetrační tester manuálně. Výhodou těchto testů je možnost vytvoření složitých scénářů a tím odhalení chyb, které by program normálně nenalezl. Další nespornou výhodou je fakt, že tester může prezentovat výsledky testu neznalým osobám a vysvětlit jaké bezpečnostní chyby systém obsahuje, jak by tyto chyby mohl útočník využít a jaké škody by mohl napáchat. Za nevýhody manuálního testování pak lze považovat časovou a znalostní náročnost. Na tuto práci je třeba vysoce kvalifikované osoby, které se vyznají nejen v problematice penetračního testování, ale taktéž v technologii platformy, kterou je třeba otestovat.
- **Automatizované testy** – jsou naprogramované scénáře, které kontrolují individuální zranitelnosti. Mezi výhody patří rychlost provedení, škálovatelnost, jednoduchá verifikovatelnost a reprodukovatelnost. Tyto testy zvládne provést i osoba, která není znalá v oblasti penetračního testování. Mezi nevýhody patří nemožnost prezentovat výsledky uživateli v přívětivé formě – je k tomu nutná znalá osoba. Taktéž tyto testy nejsou schopny otestovat všechny zranitelnosti jako složitější scénáře, nebo chyby v logice platformy.
- **Semiautomatické testy** – představují kombinaci manuálních a automatizovaných testů. Vytváří kompromis, který se snaží využít s obou případů to nejlepší. Tento typ testů bývá nejúčinější.

Penetrační testy můžeme taktéž dělit podle znalosti o testovaném systému a to následovně [18, 19]:

- **Black-box testy** – jsou nejčastějším typem testů. Penetrační tester je postaven před hotový systém s tím, že nemá žádnou představu o tom jak je systém

naprogramován. Tento typ testů je nejbližší simulace k reálnému útoku, jelikož útočník taktéž nemá informace o vnitřním fungování. Mezi největší výhody tohoto typu testů je udržení kódu v tajnosti. Dále pak vysoká variabilita, kdy si zadavatel může přizpůsobit testy podle svých představ. Nevýhody těchto testů jsou vysoké nároky na znalosti testera a nemožnost objevení sofistikovanějších chyb.

- **White-box testy** – jsou opakem black-box testů. Tester má přístup k zdrojovému kódu a tím pádem nezkoumá funkčnost sytému, ale jen poskytnutý kód. Hlavní výhodou tohoto typu testování je podrobnější analýza za relativně kratší čas. V případě aplikací je taktéž výhoda možná optimalizace kódu. Mezi nevýhody patří nutná podrobná znalost systému a kódu, kterou musí tester ovládat.
- **Grey-box testy** – jsou kompromisem mezi black-box a white-box testy. Snaží se využít výhody obou zmíněných typů testů. Penetrační tester má v tomto případě přístup k případům, které by u black-box testů neměl. Například: jak se hesla ukládají v databázi; jestli je nasazeno vhodné logování a podobně. Ovšem i tento případ má tu nevýhodu, že tester nemá přístup ke kompletnímu vnitřnímu fungování systému.

1.4.4 Bezpečnostní audit

Bezpečnostní audit komplexně zkoumá a vyhodnocuje zabezpečení systému. Prováděním pravidelných bezpečnostních auditů je možno předcházet bezpečnostním incidentům. Audit identifikuje slabá a zranitelná místa systému. Obsahem bezpečnostního auditu jsou:

- analýza bezpečnostní politiky,
- ověření shody s definovanými požadavky na bezpečnost,
- identifikace a posouzení hlavních bezpečnostních problémů,
- protokol odhalených rizik a problémů,
- plán bezpečnostních opatření,
- aktualizace bezpečnostní dokumentace (směrnice, politika).

Bezpečnostní audit se velice často doplňuje penetračními testy, které prověří fyzicky stav infrastruktury organizace [20].

1.4.5 Etické hackování

Etické hackování je pojem, pod kterým se skrývají všechny hackovací a další asociované techniky počítačových útoků. Stejně jak u penetračního testování je v etickém hackování cílem nalézt bezpečnostní mezery. Rozdíly mezi penetračním testováním a etickým hackováním jsou [21]:

- Penetrační testování je jednou z funkcí etického hackování.
- Etický hacker musí mít komplexní znalosti softwaru, programování a taktéž hardwaru. Oproti tomu penetrační tester může mít znalosti jen v určitém okruhu.
- Etický hacker musí být expert na psaní závěrečného hlášení.
- Etický hacking musí vykonávat odborná, certifikovaná osoba, aby byl výsledek efektivní. Penetrační testy může provádět téměř kdokoliv, kdo vlastní vhodné programové vybavení.
- Etické hackování je po právní stránce složitější.
- Oproti penetračnímu testování je etické hackování časově mnohem náročenější.
- Pro etické hackování je třeba mnohem širší počet přístupů v rámci infrastruktury.

Jak již bylo zmíněno výše, etické hackování provádí etický hacker. Tento pojem však není jednotný, v praxi bývá označován taktéž jako **white hat**. Opak tohoto pojmu je **black hat**, pod kterým se skrývá útočník s nekalými úmysly. Je možné se taktéž setkat s pojmem **grey hat**, jenž představuje osobu, která nemá v úmyslu způsobit škodu, ale jedná bez právního povolení a nevědomosti vlastníka testovaného objektu. Když nalezne nějakou zranitelnost tak ji nahlásí a většinou požaduje nějakou odměnu.

1.4.6 Řízení rizik

Primární cíl řízení rizik je příprava na bezpečnostní audit. Stanovuje cíle, kterých je nutno dosáhnout a porovnává je s tím, jaký je stav v současnosti. V této fázi je nejlepší čas na řešení a nápravu problémů, které zjevně existují. Řízení rizik zahrnuje: zabezpečení IT jako stav bezpečnosti sítě; zotavení po katastrofě; ochranu a převzetí služeb při selhání funkčnosti dat a serverů; obecné zabezpečení; a ochranu dat. Taktéž posuzuje jaké bezpečnostní prvky jsou zahrnuty a jestli jsou potřeba další [22].

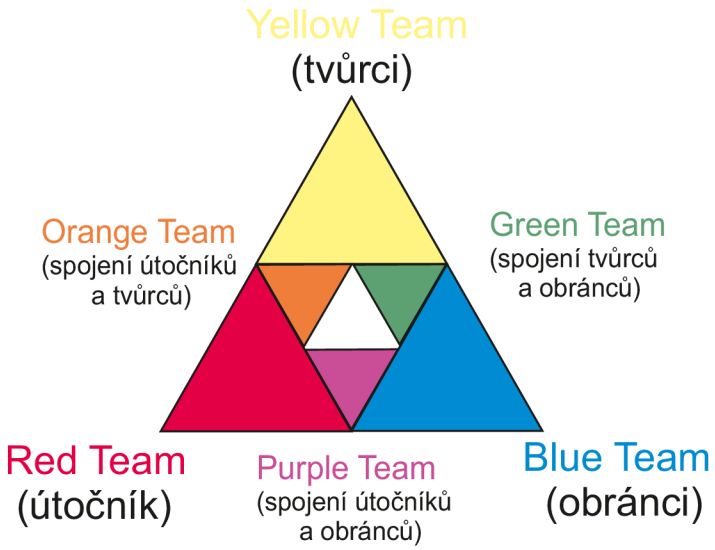
1.5 Bezpečnostní hry

Techniky a nástroje popsané v předešlých kapitolách jsou velice náročné a to hlavně vzhledem k znalostem, které člověk, který testování provádí musí mít. Pro získání těchto znalostí existuje spoustu kurzů, knih a materiálů. Avšak jedna z nejlepších cest jsou bezpečnostní hry. Pomocí těch si může i neznalý člověk zkoušet nástroje, postupy a techniky penetračního testování.

V dnešní době se na internetu můžeme setkat s nespočetným množstvím her zaměřených na bezpečnost. Některé jsou volně dostupné, jiné proprietární (k přístupu

je zapotřebí zaplatit licenci) a některé jsou privátní (dostupné jen na pozvání). V těchto hrách existují turnaje, výzvy a dokonce placené úlohy za jejíž splnění dostane řešitel nějakou finanční odměnu. Dělí se do několika kategorií:

- **CTF** (Capture the flag) je jeden z nejstarších typů bezpečnostních her. CTF úkoly jsou jedny z nejkomplexnějších a jejich složitost je variabilní. CTF problém může být vytvořen pro naprostého začátečníka, ale taktéž pro bezpečnostního experta. Celá tato idea spočívá v nalezení vlajky, která nejčastěji bývá v podobě textu. Řešitel se k tomuto textu dopracuje vyřešením zadané úlohy. Tento typ bezpečnostních her zahrnuje možnosti jako:
  - **Red team/Blue team** – jsou eventy kdy se snaží útočníci (red team) získat vlajku a obránci (blue team) se jim v tom snaží zabránit [27].
  - **Red team** – je styl hry kdy se snaží jednotlivec či skupina ukořistit vlajku [27].
  - Mimo výše zmíněné nejčastější možnosti týmů, existují i další, ty si lze vyjádřit dle obrázku 1.4.



Obr. 1.4: Značení týmů v bezpečnostních hrách

CTF využívá velké spektrum IT oblastí, nějčastěji to však jsou:

- **Programování** – úlohy vyžadují nejčastěji nějakou programovací část k vyřešení. Součástí bývá taktéž reverzní inženýrství. [27].
- **Krypto inženýrství** – nejvíce se blíží k reálným útoků, často je součástí nějaký ransomwarový typ malwaru [27].
- **Exploitace** – úlohy jsou řešitelné exploitací (užitím přetečení bufferu, formátu řetězce, SQL injekce a dalších) a získáním přístupu na běžící instanci [27].

  - **Reverzní inženýrství** – útočník se snaží odkrýt fungování skriptu, programu, aplikace a podobně [27].
- **Wargames** – není až tak obvyklý a účast bývá většinou placená. Klade na řešitele větší nároky, tudíž se doporučuje pro uživatele, kteří se v dané problematice vyznají. Řešiteli je přidělen server a ssh login, úkolem je pak exploitnout daný server.
- **Tag** – velice unikátní typ bezpečnostní hry, jde se s ní setkat jen zřídka. Zde se proti sobě utkají dva týmy, každý tým dostane server s velkým množstvím zranitelností a má jen několik desítek minut na opravu. Po uplynutí této první fáze nastává část druhá, kde se oba týmy utkají proti sobě. Vítězí tým, který získá jako první administrátorské právo na soupeřově zařízení.
- **Battleground** – velice podobný jako **tag**. Ve hře se jen vygenerují dva servery a nepřátelské strany se snaží získat administrátorský přístup na ten druhý. Fáze opravy zranitelností před nastávajícími útoky tedy není obsažena.
- **Server siege** – poměrně nový typ hry, který se spouští na stránce *hackthebox.eu*. Týmy soupeří v tom, který dříve dostane administrátorské privilegium na náhodně vygenerovaném serveru.

Na internetu je možné najít desítky unikátních a nových her. Může to být pouze zranitelná aplikace, která byla vytvořena se záměrem zabavit uživatele řešením zranitelnosti; nebo celá síť, která obsahuje desítky zranitelností. Pro nové uživatele se zájmem v této problematice pak existují i webové aplikace, kde se můžou naučit a vyzkoušet každou zranitelnost zvlášť.

1.6 Možnosti využití OpenStacku

Tato část se zaměřuje na samotné prostředí OpenStacku, které bude použito při praktickém řešení bakalářské práce. Vysvětluje jak platforma funguje a jaké jsou její možnosti v oblasti virtualizace. Taktéž popisuje základní technologie a možnosti jejich využití.

OpenStack je cloudová[6] platforma šířená pod OSS (open-source software). Což znamená, že uživatel může volně studovat jak software funguje a může si ho přizpůsobit dle potřeb. Jde taktéž o právo zasahovat do kódu a upravovat ho, avšak šíření je dovoleno pouze pod stejnou licencí. Tento software má za úkol rozdělovat virtualizovanou výpočetní kapacitu. Má pod kontrolou zdroje cloudu či datacentra. Platforma je navržená tak, aby mohla podporovat různorodé skupiny aplikací. OpenStack služba identit, známá jako **Keystone**, zajišťuje služby pro autentizaci a správu uživatelských účtu, rolí a informací pro cloudovou platformu [23, 24].

[6]Cloud je např množina HW zdrojů, která je spravována jako jeden celek.

Níže budou vysvětleny části OpenStacku, které budou využity k řešení kybernetické hry.

- **API přístup** – slouží jako rozcestník pro jednotlivé části OpenStacku. Jsou zde rozepsané služby a jejich přístupové body s porty, na kterých jsou dostupné. Každý veřejný přístup přes příkazový řádek, nebo přes webovou aplikaci využívá tento API přístup.
- OpenStack **Compute** – poskytován službou **Nova** je komponenta, která umožňuje běh více instancí různých typů napříč jakýmkoliv počtem uživatelům. Umožňuje tvořit vysoce škálovatelné a redundantní služby v cloudovém prostředí. Tato záložka obsahuje tyto podkategorie [25].
  - **Přehled** ve kterém je vypsáno využití prostředků jako například počet instancí, využitých jader virtuálního procesoru, paměti RAM, svazků, IP adres, routerů a další. Toto využití lze filtrovat za určité časové období.
  - **Instance** se berou jako abstrakce již virtuálních strojů. Každá instance musí obsahovat určité náležitosti aby mohla běžet. Tyto náležitosti jsou jméno instance, virtuální procesor, paměť RAM, místo na disku, taktéž musí být přidělena do vybrané sítě. Důležitá vlastnost je taktéž bezpečnostní skupina a přidělený veřejný klíč, aby byl správce schopný se na tento server přihlásit a pracovat s ním. Instance je tvořena z obrazu. V této záložce lze taktéž vytvořit snapshot systému což je záchytný bod.
  - **Obrazy** jsou diskové obrazy, které obsahují operační systém. Tyto soubory lze jednoduše přidat do OpenStacku a následně z nich tvořit Instance.
  - **Klíče** jak již bylo zmíněno slouží pro přístup pro instance. Tyto klíče jsou ve formátu **RSA**, nebo **X509 certifikátu**.
  - **Skupiny serverů** slouží pro sdružování serverů vzhledem k nastavené politice serveru.

- **Síť** – poskytována službou **Neutron** je záložka, která slouží téměř k veškerému nastavení síťování, které OpenStack zajišťuje. Této záložce náleží tyto kategorie:
  - **Topologie sítě**, která slouží jako vizuální zobrazení sítí a zařízení které se v nich nachází a to ve formě topologie nebo grafu.
  - **Sítě** je hlavní pod záložka. Zde lze vidět již vytvořené sítě a je možné je smazat, vytvořit nové, nebo upravit stávající. K nové síti lze přidat parametry: název, zdroj síťové adresy, síťovou adresu, verzi IP a IP výchozí brány.
  - **Router** – má za úkol tvorbu virtuálních směrovačů pomocí jmenných prostorů a iptables[7] pravidel. Představuje prakticky to samé co fyzický router.
  - **Bezpečnostní skupiny** jsou sady IP pravidel které filtrují komunikaci a jsou aplikovány na instance v síti. Uživatelé můžou nastavovat výchozí pravidla pro jejich skupiny a přidávat nové. Každý Openstack systém má již předdefinovanou výchozí skupinu, jenž je aplikována na instanci, která nemá jinou bezpečnostní skupinu [26].
  - **Firewall skupiny** používají iptables, aby aplikovaly firewall pravidla na všechny porty virtuálních strojů a porty routeru, které jsou v projektu.
  - **VPN** (Virtual Private Network) je zabezpečené propojení mezi dvěma sítěmi. Je poskytováno compute formou zvanou cloudpipes, specializovanými instancemi které jsou použity k vytvoření VPN [26].

OpenStack je velice všestranná platforma s velkou škálovatelností a téměř neomezenými možnostmi. Vysvětlení všech jejích částí je v rozsahu této práce nemožné. Prakticky by se v této platformě dala navrhnou celá sít se servery, routery a osobními počítači. Což je taktéž náplň praktické části této práce.

[7] Iptables je stavový firewall v Linuxu.

2 Návrh kybernetické hry

Tato kapitola pojednává o praktické části práce. Na začátku jsou vysvětleny základní informace o projektu. Jaké jsou hardwarové možnosti a jaké informace jsou nutné k připojení k serveru. V dalších částech se již věnuje návrhu a technickému řešení práce.

2.1 Informace o projektu v OpenStacku

Pro účely bakalářské práce byl vytvořen na platformě OpenStack projekt s názvem (cybergames), který má přidělených deset plovoucích IP adres a následující hardwarové zdroje:

- 24 virtuálních CPU,
- 24 GB paměti RAM,
- 150GB diskového úložiště.

Pro připojení k tomuto projektu je nutné nastavení VPN. Tato VPN využívá bezpečnostní rozšíření IP protokolu `IPsec`, který je založen na autentizaci a šifrování. Pro toto připojení je taktéž nutná znalost přihlašovacích údajů, adresy serveru a před sdíleného klíče. Tyto informace (stejně jako přihlašovací údaje do projektu) jsou z bezpečnostních důvodů utajeny.

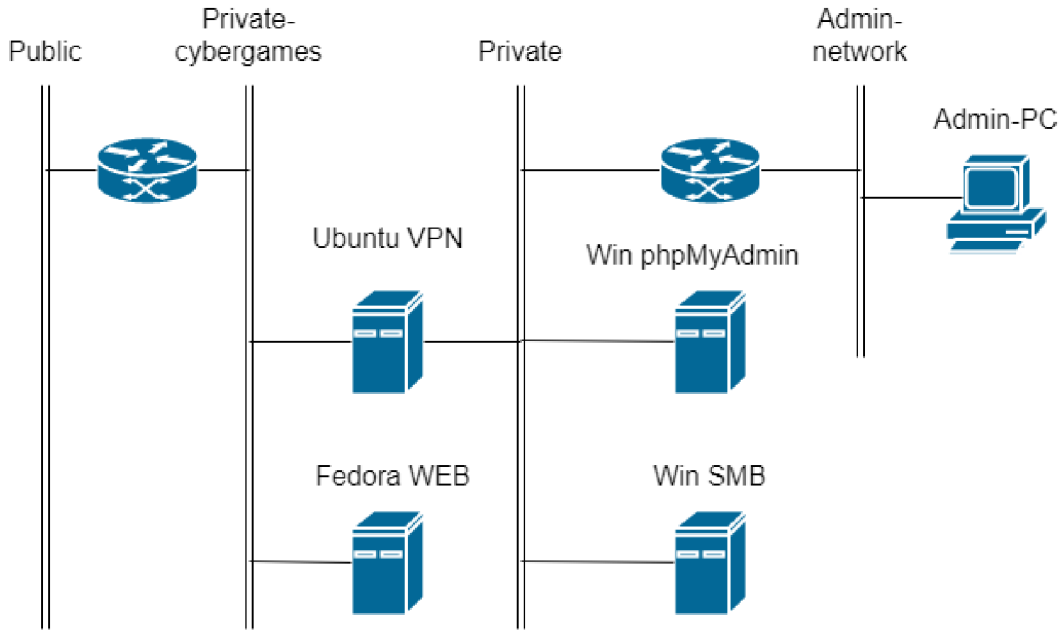
Za touto VPN se nachází dvě sítě. První síť se nazývá síť poskytovatele `Provider Network` a slouží především k propojení virtuální infrastruktury k fyzické síti, případně poskytuje připojení k Intenetu pro VMs nebo kontejnery. Tato síť je v OpenStacku mapována jako `public` a disponuje IP rozsahem 10.0.0.0/20. K této síti je dále vytvořen virtuální směrovač, za kterým je vytvořena síť `cybergames-private` s IP rozsahem 172.16.0.0/24. Tato privátní síť bude sloužit právě k napojení scénářů bezpečnostních her.

2.2 Návrh topologie kybernetické hry

Tato sekce se zabývá obecným návrhem bezpečnostní hry a jejím scénářem. Udává topologii, jenž bude v zadání užita a scénář, kterým se bezpečnostní hra bude řídit.

Pro projekt je plánováno celkem pět zařízení běžících na různých operačních systémech a několik sítí, ve kterých se tyto zařízení budou nacházet. Návrh topologie lze vidět na obrázku 2.1.

Scénář hry je uzpůsoben vzhledem k této topologii. Řešitel bude postaven před první dva servery `Ubuntu VPN` a `Fedora WEB`. Cílem první části bude získat přístup



Obr. 2.1: Topologie sítě bezpečnostní hry

do interní sítě, která se nachází za VPN serverem `Ubuntu VPN`. Toho může být docíleno jen při zdárném řešení první úlohy. Ta bude postavena na jednoduché webové zranitelnosti na serveru `Fedora WEB`.

Po zdárném řešení první části, bude řešitel vpuštěn do privátní sítě `Private`, ve které se nachází další dvě zařízení. Tyto zařízení obsahují zranitelnosti, kterých lze využít pomoci exploitace. Poslední zařízení se nachází v samostatné administrátorské síti, a jedná se o administrátorův počítač. Tato úloha bude postavena na zranitelnosti eskalace práv.

Cílem je taktéž při průchodu najít všechny vlajky, kterých se bude v síti nacházet celkem pět. Každá z vlajek by měla řešitele odměnit za zdárné řešení, některé z úloh či jejich částí.

2.3 Návrh jednotlivých částí kybernetické hry

V této části se zaměříme na jednotlivé části kybernetické hry. Budou zde teoreticky navrhnuty jednotlivé servery a jejich princip. Na úlohu budou vytvořeny 3 rozdílné podsítě, ve kterých se budou zařízení nacházet. V topologii se bude nacházet celkově pět serverů. Tyto zařízení budou postaveny na rozličných operačních systémech a distribucích.

2.3.1 Fedora WEB

Pro tento server jak již z názvu vyplývá, byl vybrán operační systém linux Fedora verze 33. Na tomto operačním systému poběží webový server s podporou skriptovacích jazyků a databází. Problematika tohoto serveru si bude dávat za cíl naučit uživatele odhalit jednoduché webové zranitelnosti, jako SQL injection. Taktéž jejím cílem bude naučit uživatele odhalit zranitelnosti, které nejsou softwarového charakteru. V tomto případě to bude administrátor, který posílá nezabezpečené přístupové údaje do firemní sítě.

Scénář

Tento linuxový server bude sloužit jako úvod do kybernetické hry. Na řešitele bude nahlíženo jako na začátečníka a prvotní server bude mít za cíl otestovat zranitelnost, která byla zmíněna v teoretické části práce. Řešiteli bude přidělena IP adresa serveru, jeho úkolem bude dostat se k vlajce a souboru s přístupem k VPN. Jak již bylo zmíněno na serveru poběží webový server na portu 80, řešitel bude muset tuto skutečnost zjistit pomocí nástroje NMAP. Poté se na tento server připojí, přes běžně dostupný internetový prohlížeč. Po připojení ho přivítá prázdná stránka. Prvním úkolem je najít přihlašovací formulář, jenž se nachází v souboru index.php. Přihlašovací formulář vyžaduje jméno a heslo pro přihlášení. Přes tenhle formulář se řešitel může dostat využitím jedné z nejvíc rozšířených zranitelností – SQL injekcí. Po nabourání do systému bude muset najít vlajku, která se skrývá v příchozích zprávách. Taktéž zde nalezne odkaz na certifikát, který je potřeba k připojení k VPN.

2.3.2 Ubuntu VPN

Pro tento server byl vyprán linuxový operační systém Ubuntu. Na tomto serveru poběží VPN, jenž bude fungovat jako vstup do privátní sítě. Student nebude muset znát žádné informace o serveru, jelikož VPN bude poskytována přes službu `openVPN`. K té se připojí přes příkazovou řádku za využití certifikátu. Tato síť bude obsahovat další virtualizované servery.

Scénář

Tento virtuální stroj je vytvořen čistě za účelem přístupu do sítě `Private`, topologie znázorněné na obrázku 2.1. Jak již bylo zmíněno po ukořistění vlajky na serveru `Fedora WEB`, řešitel taktéž získá konfigurační soubor VPN serveru, který využije k přístupu. Zařízení jinak nebude obsahovat žádnou nyní známou zranitelnost a taktéž se na něm nebude nacházet žádná vlajka.

2.3.3 Windows SMB

Pro tento server byl zvolen operační systém Windows Server 2012 R2. Tento server bude poskytnut bez větších úprav jako ve stavu jeho vydání. Jeho problematika si klade za úkol naučit uživatele pracovat s CVE (Common Vulnerabilitiesand Exposures), což jsou veřejně známé bezpečnostní hrozby a prohloubit jeho znalosti v oblasti exploitace. Především se bude jednat o program metasploit, který je nejčastěji pro takový typ problematiky využívám.

Scénář

Tento server je pokračováním předchozího scénáře, jelikož do sítě, kde se vyskytuje tento server se student dostane, jen při úspěšném vyřešení první úlohy a připojení na VPN server. Tato druhá úloha je již pokročilejšího charakteru. Bude klást na řešitele vyšší nároky, především vzhledem k složitější metodě exploitace. Řešitel bude muset v tomto případě odhalit zastaralou verzi operačního systému, která bude taktéž bez důležitých aktualizací a nalézt slabinu `CVE-2017-0144`. Slabina umožní útočníkovi nad systémem převzít kontrolu. To, ale jen v případě, že zvolí správný exploit. Při úspěšném řešení bude student muset v systému vyhledat správnou vlajku. Ta se bude nacházet v výchozí složce webového serveru.

2.3.4 Windows phpMyAdmin

Tento server je postaven na stejné Windows Server 2012 distribuci jako předchozí `Windows SMB` server. V této úloze student taktéž využije znalosti exploitace, nabyté z předchozí úlohy. Tato exploitace je složitější z důvodu nutnosti nalezení správného exploitu a manipulace s ním.

Scénář

Problematika tohoto serveru spočívá v nalezení nezabezpečeného vstupu do administrace prostředí phpMyAdmin a následně jeho exploitace. Tento scénář bude obsahovat dvě vlajky, vzhledem k délce úlohy. Student získá první vlajku při úspěšném vstupu do phpMyAdmin administrace a druhou při zdařilém exploitaci serveru. Exploitace je založena na zastaralé verzi phpMyAdmin.

2.3.5 Admin-PC

Pro tento server byl zvolen operační systém linux Fedora 27. Rozdíl oproti serveru `Fedora WEB` je ve verzi tohoto systému. V této úloze se student naučí odhalit slabinu systému, díky které bude moci eskalovat práva v sytému. Taktéž se naučí vyhledat

potřebný exploit a pracovat s ním. Tato úloha je v celém průchodu kybernetickou hrou považována za nejsložitější a to hlavně z důvodu složitého nalezení slabiny systému.

Scénář

V této úloze se řešitel připojí na server pomocí nezabezpečeného protokolu telnet. Pro připojení lze využít defaultní přihlášení, které administrátor nakonfiguroval. Po připojení se student dostane na účet hosta, který má omezená práva a nemá možnost se dostat k vlajce. V tomto bodě bude nucen odhalit zranitelnost systému, jenž vyplývá ze zastaralého jádra a celkově z verze operačního systému. Tuto zranitelnost lze najít pod CVE označením `CVE-2017-16995`. Následně řešitel musí vyhledat funkční export, zkopírovat jej do systému a provést za jeho pomocí eskalaci práv.

3 Tvorba kybernetické hry

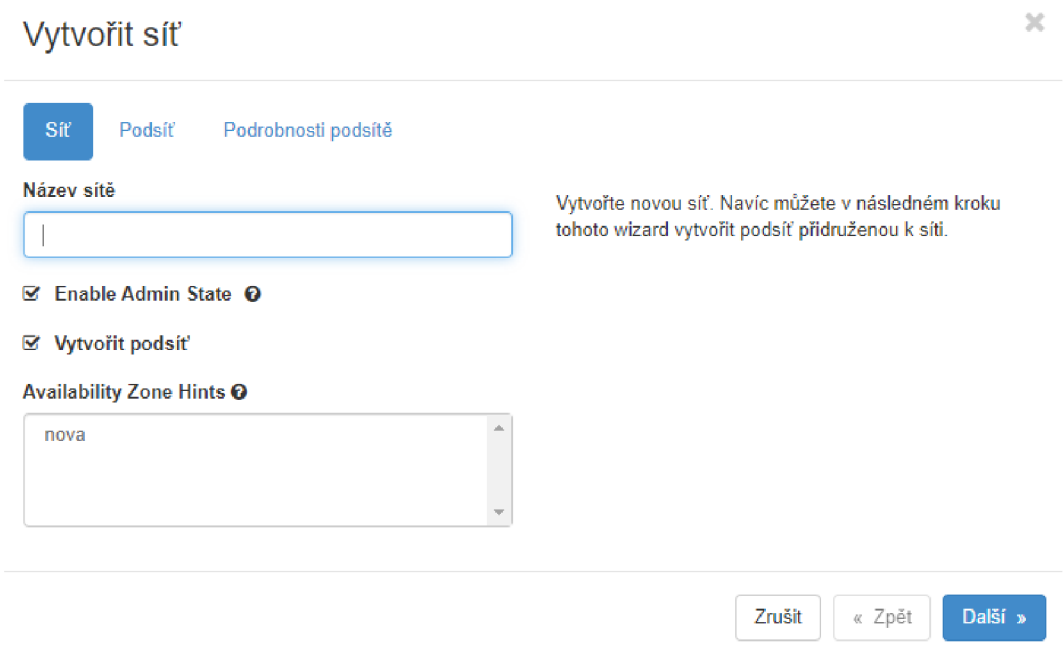
Tato kapitola se věnuje praktické tvorbě kybernetické hry. V první části se zaměřuje na tvorbu v prostředí OpenStack. Vysvětluje jak se vytváří jednotlivé instance. Dále se zabývá tvorbou a propojením síti, do kterých budou jednotlivé instance přiřazeny. Taktéž zde postupně budou zmíněny překážky, které se při tvorbě naskytly.

3.1 Tvorba v prostředí Openstack

3.1.1 Vytváření sítí

Pro tuto úlohu budou vytvořeny tři sítě, stejně jak na diagramu 2.1. Jedná se o sítě `Private-cybergames`, `Private` a `Admin-network`. Jako ukázkový příklad tvorby této sítě byla zvolena síť `Private`, která byla vytvořena následovně.

V hlavním menu se nachází nabídka síť a v ní podnabídka sítě. V této podnabídce pak v pravém rohu je možnost vytvořit novou síť. Po rozkliknutí je otevřen formulář, který je znázorněn na obrázku 3.1.



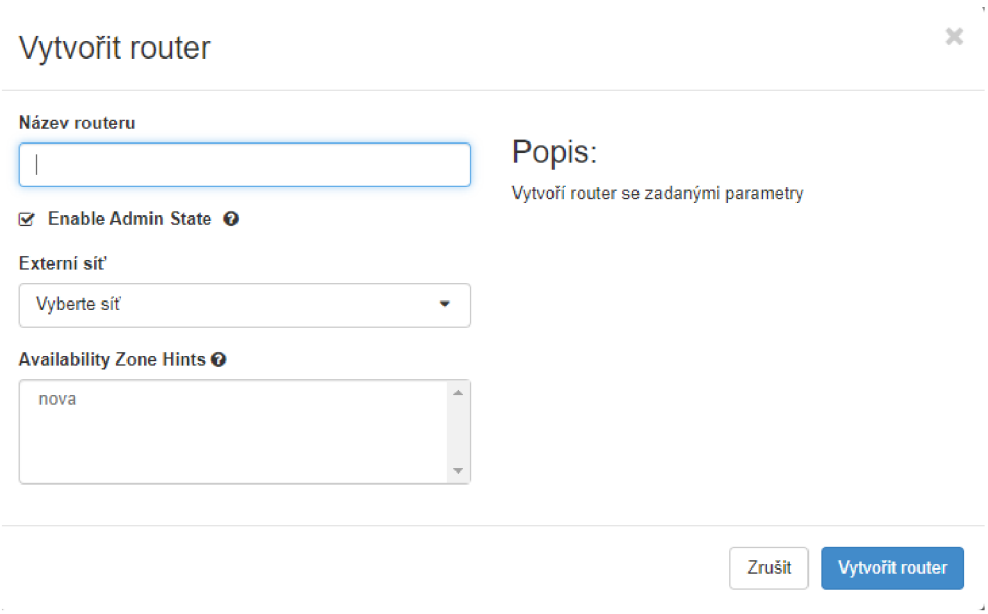
Obr. 3.1: Formulář pro vytvoření nové sítě

V této nabídce je hlavní název sítě, jenž byla pojmenována `Private`. V záložce formuláře byl jinak ponechán zbytek možností na výchozí. V nabídce podsíť se nastavuje název subnetu, síťová adresa, její rozsah a verze internetového protokolu.

Jako poslední možnost je zde výchozí brána, která byla nastavena vždy, jako první adresa v síti. V poslední záložce formuláře, lze nastavit rozsahy adres, jenž budou přidělovány pomocí DHCP (Dynamic Host Configuration Protocol) a které budou přiděleny ručně. Dále zde lze nastavit, které adresy budou použity pro služby DNS (Domain Name System) a cesty hostitele. Po zatrhnutí možnosti `vytvořit` bude síť vytvářena a bude možné ji během několika vteřin využít.

Propojení mezi sítěmi

Propojení sítí může být docíleno hned několika způsoby. Nejjednodušší možnost je vytvoření routeru, ke kterému se k jednotlivým rozhraním připojí vybrané sítě. Toho je docíleno následovně. V hlavním menu v nabídce sítě se nachází možnost routery. V této podnabídce pak v pravém rohu je možnost vytvořit nový router. Tato možnost zobrazí stejný formulář jak na obrázku 3.2.



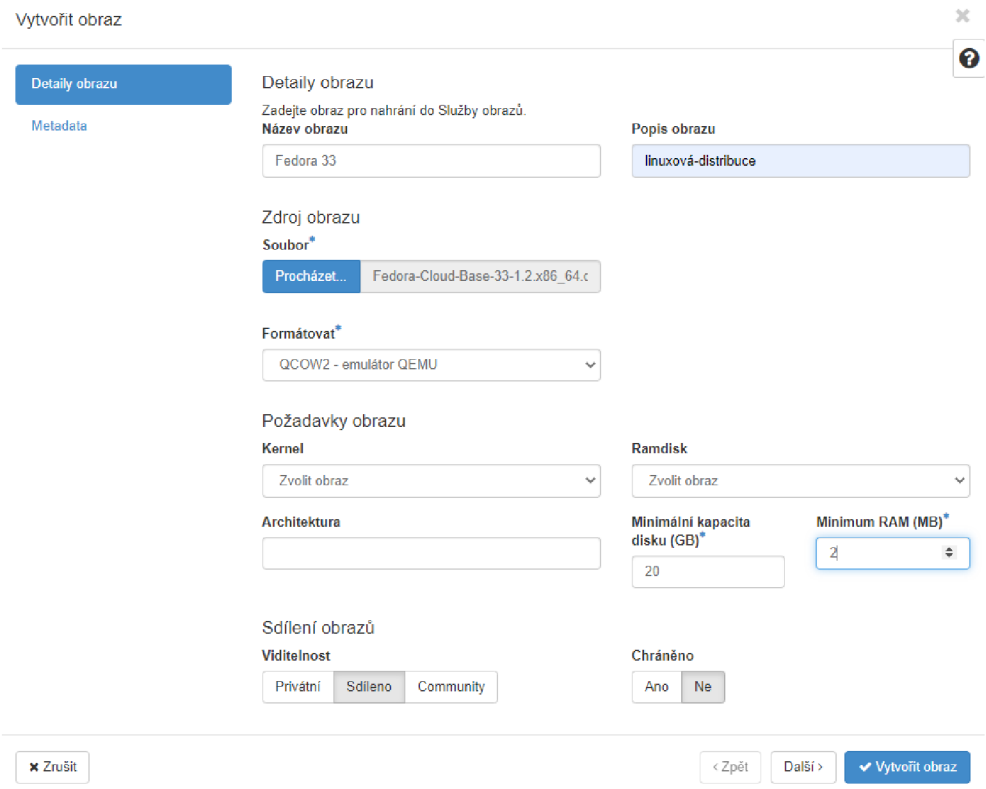
Obr. 3.2: Formulář pro vytvoření nové sítě

Ve formuláři 3.2 je nutné vyplnit jen název routeru. Jeho konfigurace probíhá až po jeho vytvoření. Konfigurace je docíleno v nastavení vytvořeného routeru. Kde v záložce rozhraní, můžou být připojeny vytvořené sítě.

Druhá možnost propojení je přes instanci. K instanci je možno připojit více rozhraní. V tom případě při správné konfiguraci může být propojeno více sítí mezi sebou. Vše záleží na operačním systému a konfiguraci zvolené instance.

3.1.2 Tvorba instance

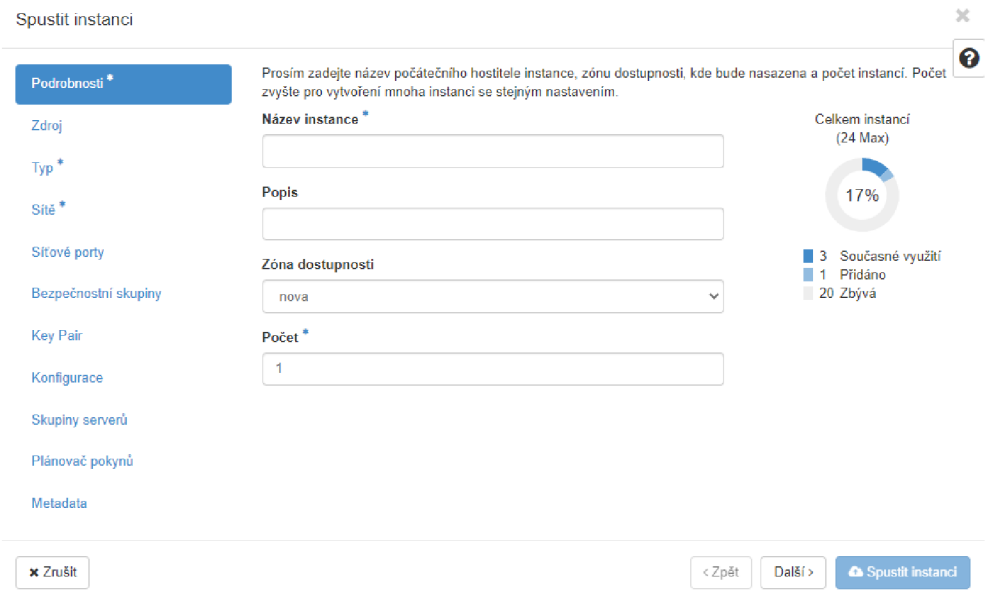
V této sekci bude vysvětleno nahrání obrazu na server, tvorba instance a její nastavení. Tvorba bude vysvětlena na serveru `Fedora WEB`. Pro implementaci tohoto systému do OpenStacku je ideální stáhnout speciální formát QCOW2 (QEMU Emulator), který je uzpůsoben cloudovým aplikacím. Po stažení je nutné tento obraz disku importovat do úložiště OpenStack. Import je proveden ve webovém rozhraní OpenStacku obdobně jako na obrázku 3.3.



Obr. 3.3: Import obrazu disku do OpenStack systému

Na obrázku 3.3 je zobrazena záložka pro přidání obrazu do systému. V tomhle formuláři je nutné vyplnit povinné pole `soubor`, přes který vkládáme stažený obraz. Další povinné parametry jsou formát souboru, jaké minimální místo vyžaduje operační systém na disku a kolik potřebuje paměti RAM. Další důležité parametry jsou `viditelnost` a `chráněnost`. Při změně viditelnosti lze obraz sdílet ostatním projektům, nebo nastavit jako privátní, či sdílet v komunitě. Ostatní parametry si uživatel nastavuje podle jeho preferencí. Po vyplnění formuláře potvrdíme přidání tlačítkem `vytvořit obraz`.

Po přidání obrazu je již možné virtuální stroj vytvořit. Toho lze docílit spuštěním naimportovaného obrazu. Tato akce vyvolá stejný formulář jako na obrázku 3.4.



Obr. 3.4: Vytvoření nové instance v OpenStack systému

Zde je taktéž nutné vyplnit povinné informace, které jsou označeny hvězdičkou. Jedná se o `název instance` a `typ instance`, která určuje jak velké hardwarové zdroje budou instanci přiděleny. Další povinná položka je `síť`, která určuje, do které sítě zařízení bude připojeno. Volitelné pole je zase možno doplnit dle preferencí uživatele. Nakonec je vytvoření nutné potvrdit tlačítkem `spustit instanci`. Po vytvoření, je ještě nutné v záložce instance virtuálnímu stroji přidělit plovoucí IP adresu pro vzdálené připojení. Taktéž zde lze sledovat stav tohoto serveru a nahlédnout do konzole, nebo do logu.

Typ instance

Jak již bylo zmíněno typ instance udává, jaké množství hardwarových zdrojů, bude serveru přiřazeno. Pro tuto úlohu bylo vytvořeno několik šablon, podle kterých můžeme zdroje přiřadit. Typy šablon jsou sepsány níže:

- **cyber-server 1** – pro který je přidělen 1GB RAM, 1 virtuální procesor a 10GB systémového disku.
- **cyber-server 2** – pro který je přidělen 2GB RAM, 2 virtuální procesor a 20GB systémového disku.
- **cyber-server 3** – pro který je přidělen 4GB RAM, 4 virtuální procesor a 30GB systémového disku.

3.2 Tvorba serverů a její problémy

Tato sekce vysvětluje jak byly vytvořeny jednotlivé servery kybernetické hry. Uvádí použité nástroje a postupy. Taktéž popisuje problémy, které musely být při jejich tvorbě řešeny.

3.2.1 Fedora WEB

Použité programy, nástroje a přidělený typ instance

- **Fedora 33** je linuxová inovativní open-source distribuce založena na RPM[1]. Fedora je především známá díky tomu, že je podporována komunitou napříč celým světem a to jak pokročilými techniky, tak i nezkušenými nováčky. Společně pak vytvářejí software pro uživatele tohoto systému.
- **Apache2** je HTTP (Hypertext Transfer Protocol) serverový projekt, který zajišťuje open-source HTTP server pro moderní operační systémy.
- **PHP** je skriptovací jazyk určený pro webové aplikace. Pro potřeby tohoto serveru bude použita verze 7.4.12.
- **mySQL** je databázová open-source služba, pro plnou správu a kontrolu nad databázovými službami. Umožňuje vytváření nativních cloudových aplikací.
- **Typ instance** pro tento server byl zvolen **cyber-server 3**.

Tvorba serveru

Jak již bylo zmíněno, jako první je třeba danou instanci vytvořit. Tuto tvorbu jsme si již popsali v předchozí kapitole 3.1.2.

Pro připojení k instanci byl použit WSL (Windows Subsystem for Linux), jenž umožňuje nainstalovat jádro linuxu pod windows a následně používat linuxové nástroje. Po instalaci je možné se přes SSH (Secure Shell) vzdáleně na virtualizovaný stroj připojit. Pro připojení není nutné zadávat heslo, jelikož to je zprostředkováno pomocí certifikátu, který byl nastaven při vytváření virtuálního stroje.

Jako další byl nainstalován webový server spolu s podporou jazyků php a mySQL, jenž jsou třeba pro správnou funkčnost webové stránky. Toho bylo docíleno příkazy:

```
dnf install httpd -y
systemctl start httpd.service
dnf -y install https://rpms.remirepo.net/fedora/remi-release-33.rpm
dnf config-manager --set-enabled remi
dnf module reset php
dnf module install php:remi-7.4
dnf install mysql-community-server
systemctl enable mysqld.service
systemctl start mysqld.service
```

[1] Balíčkovací systém v Linuxu.

Po této instalaci je ještě nutná konfigurace, která byla uskutečněna následným způsobem:

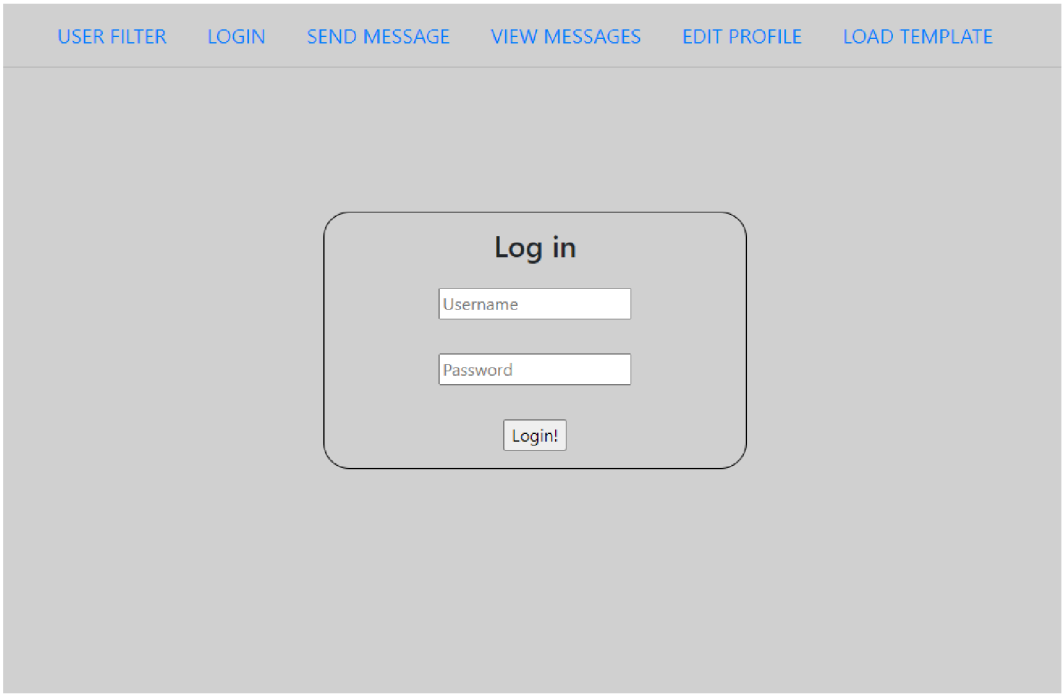
```
SELECT * FROM uzivatele WHERE jmeno = $_POST['jmeno'] grep 'docasne heslo' /var/log/mysqld.log |
    tail -1 mysql_secure_installation
```

Dále již bylo možné se připojit na vytvořený mySQL server a to příkazem:

```
mysql -u root -p
```

Nasazení webové stránky

Pro webovou stránku byl zvolen již existující projekt, který lze nalézt volně dostupný na githubu https://github.com/jadz/php-sploits. Při nasazení bylo nutné importovat databázi, čehož bylo docíleno za pomoci souboru db.sql. Tento soubor obsahuje mySQL kód, který při spuštění vytvoří námi požadované tabulky, relace a vloží do databáze data. Dále bylo třeba naimportovat soubory webové stránky na server do složky /var/www/html. Po importu byla ještě provedena mírná vizuální obměna kvůli přehlednosti a větší realističnosti, viz obrázek 3.5.



Obr. 3.5: Nový vzhled webové stránky

Na závěr musely být přidány do webové aplikace zprávy, které obsahovaly vlajku a certifikát. Tyto zprávy byly vloženy do příchozích zpráv administrátora.

Problémy a jejich řešení

V této úloze se naskytlo hned několik problémů. První při tvorbě databáze, kdy nebylo možné se do ní po instalaci přihlásit. Tento problém byl vyřešen přidáním parametru "skip-grant-tables"do souboru /etc/my.cnf. Ten dovolí připojení bez nutnosti znát heslo. Tohle však není ideální stav z důvodu bezpečnosti, proto v databázi byl nakonfigurován nový uživatel příkazy:

```
ALTER USER 'root'@'localhost' IDENTIFIED BY 'HackTheWorld456*';
FLUSH PRIVILEGES;
```

Na závěr byl vymazán parametr skip-grant-tables ze souboru my.cnf a služba mySQL byla restartována.

Jako druhý problém v projektu byly samotné webové stránky. Tento projekt byl postaven na starší verzi PHP a to zapříčinilo jeho nefunkčnost. Proto bylo nutné některé syntaxe v kódu opravit.

3.2.2 Ubuntu VPN

Použité programy, nástroje a přidělený typ instance

- **Ubuntu 20.04.2** je open-source linuxová distribuce. Je to taktéž nejnovější verze tohoto operačního systému.
- **OpenVPN server** je open-source nástroj, který dokáže vytvořit VPN tunel mezi klientem a serverem.
- **Typ instance** pro tento server byl zvolen **cyber-server 2**.

Tvorba serveru

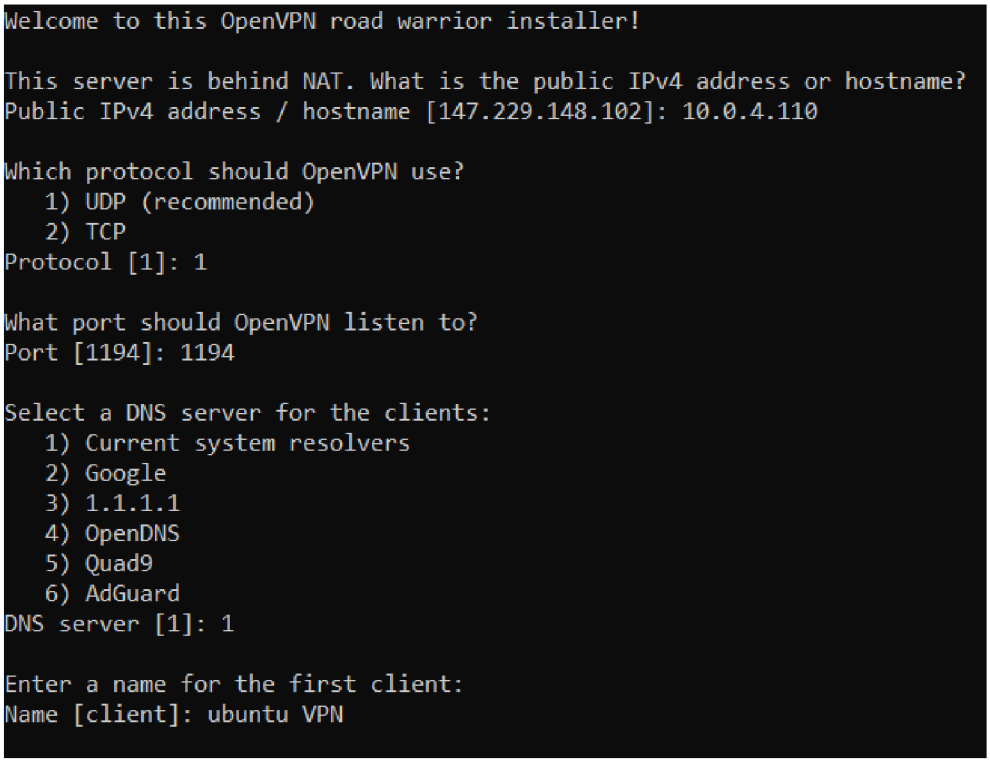
Jako první bylo třeba stáhnout obraz serveru. Ten najdeme pro naše potřeby volně dostupný na webových stránkách https://cloud-images.ubuntu.com/. Přesný typ obrazu byl zvolen `bionic-server-cloudimg-amd64.img`. Tvorba instance je identická kromě nastavení rozhraní, kde musí být stroj nakonfigurován tak, aby propojoval sítě `Private-sybergames` a `Private`, kde se nachází další zařízení.

Po vytvoření instance a přístupu k ní je třeba nainstalovat OpenVPN službu. Toho bylo docíleno následujícími příkazy.

```
sudo apt-get update
wget https://git.io/vpn -O openvpn-install.sh
sudo chmod +x openvpn-install.sh
sudo bash openvpn-install.sh
```

Tyto příkazy jako první nainstalují aktualizace a následně stáhnou volně dostupný skript z gitu. Ten při spuštění nainstaluje OpenVPN službu. Tento skript umožňuje zjednodušený průchod instalací OpenVPN, kde je nutné vyplnit základní

informace o serveru. Pomocí tohoto skriptu se dají následně do služby přidávat noví uživatelé, či jednotlivé uživatele mazat. Taktéž s jeho pomocí lze službu VPN odinstalovat. Instalace skriptu vypadá totožně jak na obrázku 3.6.



Obr. 3.6: Instalace OpenVPN

Problémy a jejich řešení

Tvorba tohoto serveru proběhla jinak, než bylo v plánu. Jako první bylo testováno spuštění VPN serveru na linuxové distribuci Fedora. Instalace měla být provedena za pomocí příkazů a služba měla být nainstalována manuálně. Toho však nebylo přes vysoké množství pokusů dosaženo. Proto bylo přistoupeno na instalaci VPN služby na distribuci Ubuntu za pomocí již napsaného skriptu. Ten celou instalaci zásadně zjednodušil.

3.2.3 Windows SMB

Použité programy, nástroje a přidělený typ instance

- **SMB (Server Message Block)** je komunikační protokol pro sdílení souborů, tiskáren, seriových portů a další komunikace.
- **Windows Server 2012 R2** je windows distribuce založena na standardní verzi windows 10. Slouží jako síťový serverový operační systém.
- **Typ instance** pro tento server byl zvolen **cyber-server 2**.

Tvorba serveru

Jako první je nutné nainstalovat obraz zvoleného systému, jenž lze najít volně ke stažení na stránce cloudbase.it. Po stažení, byl tento obraz vložen do openstack systému a stejně jak v předchozích případech z něj byla vytvořena instance. Změna v tomto operačním systému je ta, že k této instanci můžeme přistupovat v grafickém rozhraní, při načtení konzole dané instance. Není nutné se na instanci připojovat přes SSH.

Po této prvotní iniciaci, již lze přejít ke konfiguraci a nutným přípravám. Jako první, bylo vyžadováno odinstalování nezbytných aktualizací. Jelikož tyto aktualizace na sebe navazují a bylo by velice obtížné odinstalovat jen ty nutné, tak byly odinstalovány všechny. Toho bylo docíleno generováním seznamu všech aktualizací a jeho uložením do textového souboru. K tomu byl využit příkaz:

```
wmic qfe get hotfixid > c:\list.txt
```

Následně byl tento soubor modifikován v textovém editoru, tak aby nahradil řetězec kb prázdným místem. Na ploše byl vytvořen soubor s příponou .bat a do něj vložen script níže:

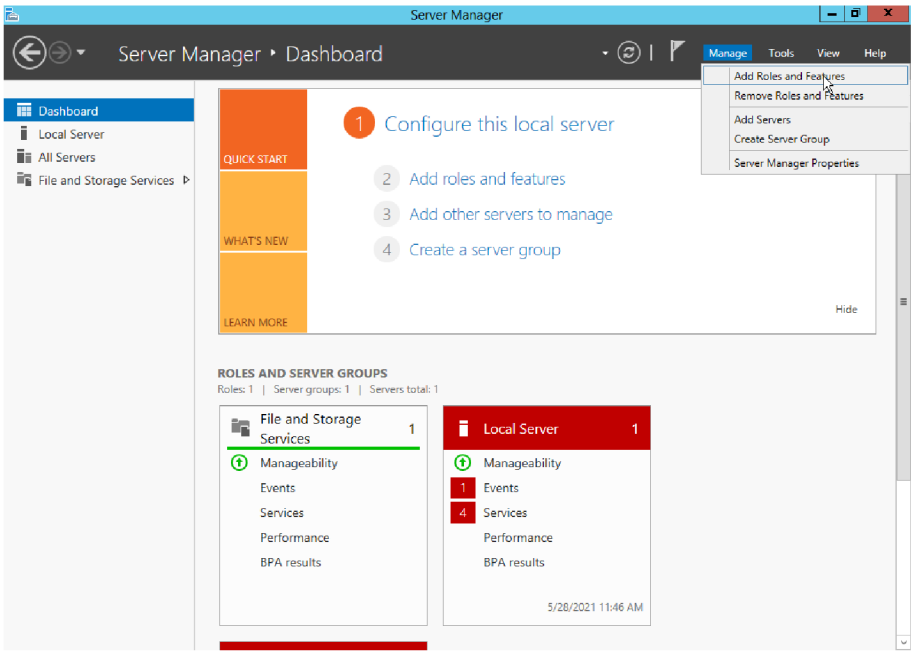
```
@echo off
for /f %%i in ('type c:\list.txt') do (
echo "Uninstalling KB%%i"
wusa /uninstall /kb:%%i /quiet /norestart)
shutdown /rt
```

Poté byl script spuštěn příkazy:

```
cd Users\administrator\Desktop
nazevsouboru.bat
```

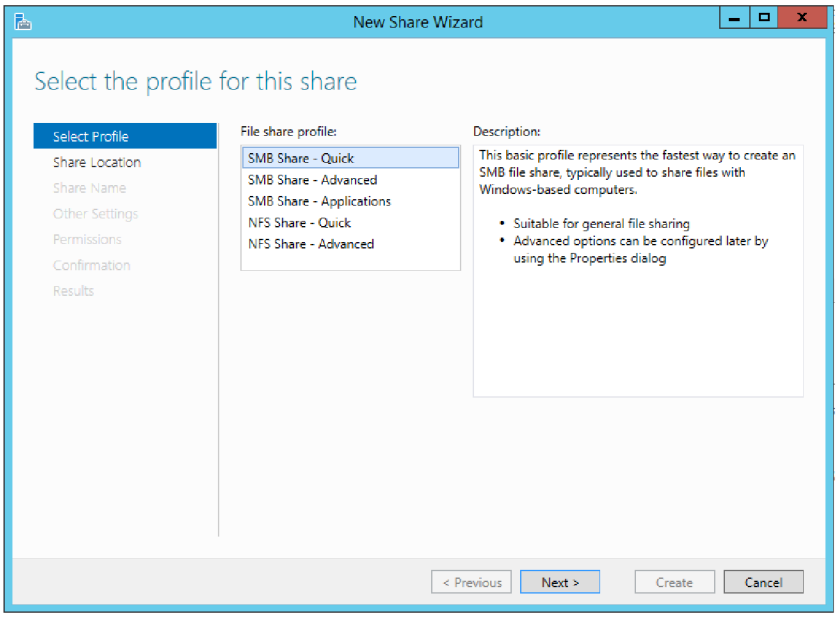
Po této odinstalaci bylo taktéž nutné danou službu povolit a nainstalovat sdílení. Toho bylo docíleno ve správci serveru. Tento správce vypadá jako na obrázku 3.7.

Kliknutím na `Add Roles and Features` zobrazíme nabídku, kterou je nutno projít. V této nabídce v `Server Roles` přidáme `File and Storage Services` a následně ve `Features` zaklikneme námi požadovanou SMBv1 službu.



Obr. 3.7: Správce Windows 2012 serveru

Ve správci serveru, byla následně sdílena nová složka přes protokol SMBv1. Toho bylo docíleno ve správci serveru v záložce `File and Storage Services/Shares`. Zde jednoduše pomocí pravého tlačítka na myši byla vyvolána nabídku, která provede tvorbou nového sdílení viz obrázek 3.8.



Obr. 3.8: Nové sdílení

V průběhu tvorby je nutné složku pojmenovat a nastavit práva pro uživatele `host`. Zbytek nabídek může být ponechán jako výchozí.

Vlajka na tomto serveru byla umístěna na plochu administrátora.

Problémy a jejich řešení

Teoretická tvorba tohoto serveru byla velice jednoduchá. Avšak při jeho praktické instalaci se vyskytlo hned několik problémů. První problém byl, kdy při odinstalaci aktualizací, bylo třeba restartovat systém. Při tomto restartu, si však tyto aktualizace systém sám znovu nainstaloval. Tato skutečnost byla vyřešena zakázáním aktualizací v registru Windows. Toho bylo docíleno tak, že byl změněn registr `Explorer` z hodnoty 1 na 0. Tento registr se nachází v cestě:

```
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer
```

Další problém se vyskytl při samotném testování exploitace serveru, kdy daná služba nebyla stabilní a nešla exploitovat. To bylo vyřešeno aktivováním účtu hosta a změnou postupu exploitace. Účet hosta byl aktivován v příkazové řádce příkazem:

```
net user guest /active:yes
```

Poslední problém se naskytl při převzetí kontroly nad serverem. Jelikož by bylo složité získávat nad strojem reverzní shell, tak na něm byl aktivován RDP (Remote Desktop). To řešiteli umožní jednoduché připojení k serveru. Aktivace RDP je možná v menu:

```
Control Panel\System and Security\System.
```

V tomto menu zvolíme `Advanced system settings` a povolíme vzdálené připojení zatrhnutím `Allow remote connections to this computer`.

3.2.4 Windows PHPMYADMIN

Použité programy, nástroje a přidělený typ instance

- **XAMP v** je multiplatformní balíček, který obsahuje software jako Apache, mySQL a podobné.
- **Windows Server 2012 R2** je windows distribuce založena na standardní verzi windows 10. Slouží jako síťový serverový operační systém.
- **Typ instance** pro tento server byl zvolen **cyber-server 2**.

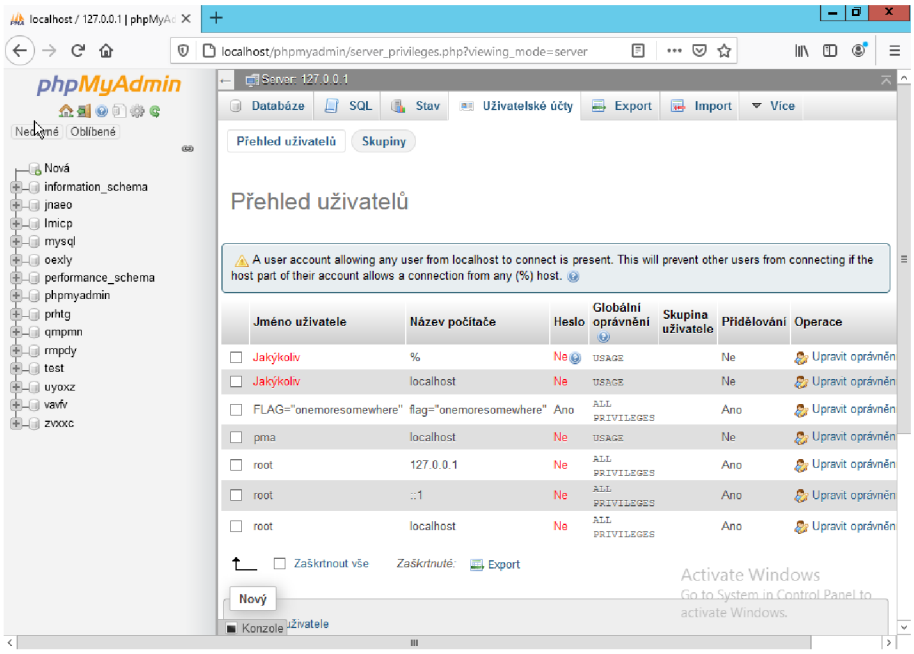
Tvorba serveru

Jelikož tento server je stejné distribuce jako `Windows server SMB`. Tak byl vytvořen ze stejného obrazu a přistupuje se k němu obdobně. V tomhle případě bylo nutno

předejít stejné možnosti exploitace, jako v posledním případě. Proto byla ze serveru odstraněna služba SMB a ponechány veškeré aktualizace.

Jelikož zranitelnost tohoto systému se zakládá na staré verzi phpMyAdmin 4.8.1, bylo nutné tuto verzi nainstalovat. Jako nejjednodušší možnost, bylo ji stáhnout již jako součást balíčku programů XAMP. XAMP tento program obsahuje ve verzi 7.2.6 a je volně dostupný na stránce https://sourceforge.net. Po stažení stačí projít jednoduchou instalaci a není třeba řešit složitější postupy.

Po instalaci je nutné program XAMP spustit a zapnout v něm moduly Apache a MySQL. Po spuštění lze již načíst administraci Apache serveru a modul phpMyAdmin přes prohlížeč. Toho docílíme zadáním do URL prohlížeče adresu 172.0.0.1 což je lokální host. Do administrace prostředí se vstupuje přes výchozí přihlášení, které nebude vzhledem k charakteru úlohy změněno. Administrace phpMyAdmin je znázorněno na obrázku 3.9.



Obr. 3.9: Administrace prostředí phpMyAdmin

Vlajky na tomto serveru byly umístěny do administrace phpMyAdmin a do kořenové složky webového serveru.

Problémy a jejich řešení

Na tento server byly prvotní pokusy nasadit Apache, mySQL a phpMyAdmin, jako samostatné programové vybavení. Toho však nebylo docíleno vzhledem k problému

s verzemi programů. Jako alternativa byl zvolen program XAMP, který v konečném důsledku fungoval bez větších problémů.

3.2.5 Admin PC

Použité programy, nástroje a přidělený typ instance

- **Telnet** je protokol používaný pro vzdálené připojení pomocí textového rozhraní.
- **Fedora 27** je stejná linuxová distribuce, jako byla použita v části 3.2.1. Rozdíl je v její verzi, jelikož novější verze neobsahuje žádnou známou zranitelnost, kterou by jsme mohli využít na tento typ útoku.
- **Typ instance** pro tento server byl zvolen **cyber-server 2**.

Tvorba serveru

Jako první je nutné najít vhodný obraz distribuce Fedory ve vydání 27. Ten je volně dostupný na stránce https://archives.fedoraproject.org. Konkrétně byl použit obraz `Fedora-Cloud-Base-27-1.6.x86-64.qcow2`. Po stažení byla vytvořena instance. V tomto případě provádíme konfiguraci přes SSH ve WLS.

Jelikož tento server bude sloužit pro vyzkoušení útoku na eskalaci práv, tak musí být umožněn přístup na tento server. Byl proto zvolen nejznámější typ vzdáleného připojení a to přes telnet. Pro přístup byl taktéž nakonfigurován uživatel s přihlašovacím jménem `host` a heslem `host`. Toho bylo docíleno příkazy:

```
sudo yum install telnet--server xinetd --y
sudo systemctl enable telnet.socket
sudo systemctl start telnet.socket.
sudo adduser host
sudo passwd host
```

Po této instalaci již bylo možné se na server vzdáleně připojit. Avšak pro zdárné řešení je nutno přidat do systému nástroje, které by řešitel mohl eventuálně použít. Jedná se o příkazy `nano` a `gcc`. Příkaz nano slouží jako editor souboru a příkaz gcc k překladu zdrojového kódu.

Pro ztížení úlohy byly taktéž na serveru povoleny další porty jako HTTP a FTP. Ty mohou řešitele zmást a navést jiným směrem.

Na konec úlohy byla umístěna vlajka do soukromé složky administrátora serveru.

4 Měření HW náročnosti kybernetické hry

V této kapitole je měřena hardwarová náročnost, která je závislá na hraní kybernetické hry. Také je v ní popsáno, jaké minimální zdroje jsou nutné pro běh operačních systémů a které zdroje jsou využity při hraní této kybernetické hry. V závěru kapitoly jsou nabyté informace porovnány se skutečnými zdroji, které byly v praktické části virtuálním strojům přiřazeny.

4.1 Testování kybernetické hry studentem

Z důvodu testování funkčnosti a hardwarového měření, musela být hra otestována nezávislým studentem. Tento student neměl žádné předešlé znalosti v penetračním testování a exploitacích. Vycházel čistě z informací, obsažených v návodu k cvičení a znalosti práce s operačním systémem linuxové distribuce.

Toto řešení úlohy probíhalo v rozpětí devadesáti minut a to od 17:30 do 19:00. Průběh hraní hry byl zachytáván monitorujícím nástrojem `Prometheus` a zachycené data, byly virtualizovány ve webovém prostředí nástrojem `Grafana`.

Řešení úlohy studentem proběhlo bez větších komplikací. Studentovi byl poskytnut virtuální stroj s distribucí `Kali linux` a VPN přístup k OpenStack serveru. Připojení a první sken pomocí nástroje NMAP, proběhl bez problémů, avšak nedodržoval vypsanou metodologii a snažil se bez předešlého prozkoumání služeb exploitovat, což nevedlo ke zdárnému konci. Dále si vyžádal první dvě nápovědy a s nimi úlohu dokončil. Tato první úloha zabrala dvacet minut.

Po dokončení první části se zvládl připojit do privátní sítě přes VPN server. Tuto síť oskenoval a našel další dva virtualizované servery. Servery řešil chronologicky z výsledku NMAP skenu. Jako první před ním stál `Windows SMB` server. Tento server zabral nejvíce času, jelikož se student seznamoval s nástrojem Metasploit. Na server byly potřeba tři nápovědy a řešení zabralo třicet minut.

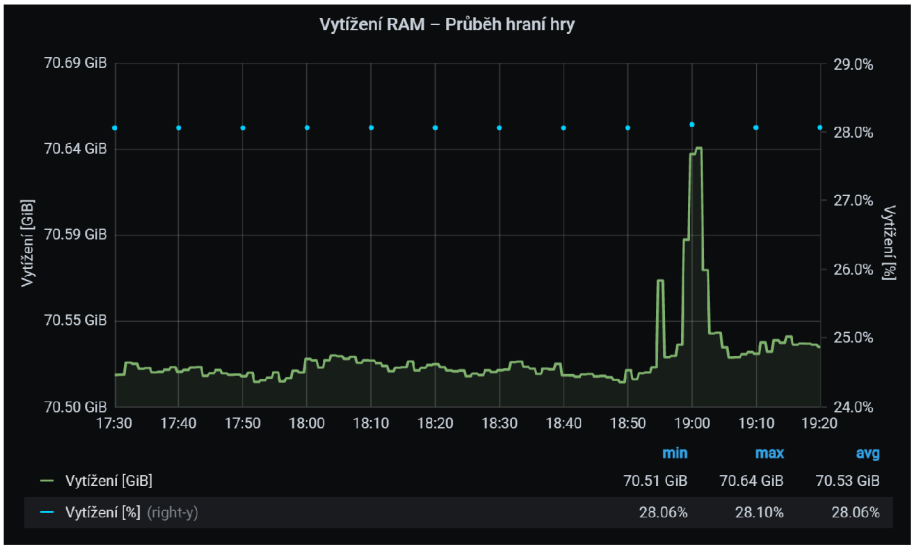
Druhý server `phpMyAdmin`, který funguje na podobném principu byl méně problematický. Byla zde potřeba jen jedna nápověda, která studenta navedla na exploitaci zastaralé aplikace phpMyadmin. Student zvládl vypracovat tuto část během dvaceti minut.

Poslední část cvičení server `Admin PC`, který je považován za nejsložitější byl pro studenta velice problematický. Pro vyřešení použil všechny čtyři nápovědy. Zvládl však úlohu vyřešit ve zbylém čase. To je dvacet minut.

Na závěr ohodnotil cvičení jako přínosné, avšak složité. Líbila se mu možnost otestovat metody hackování na reálných zařízeních.

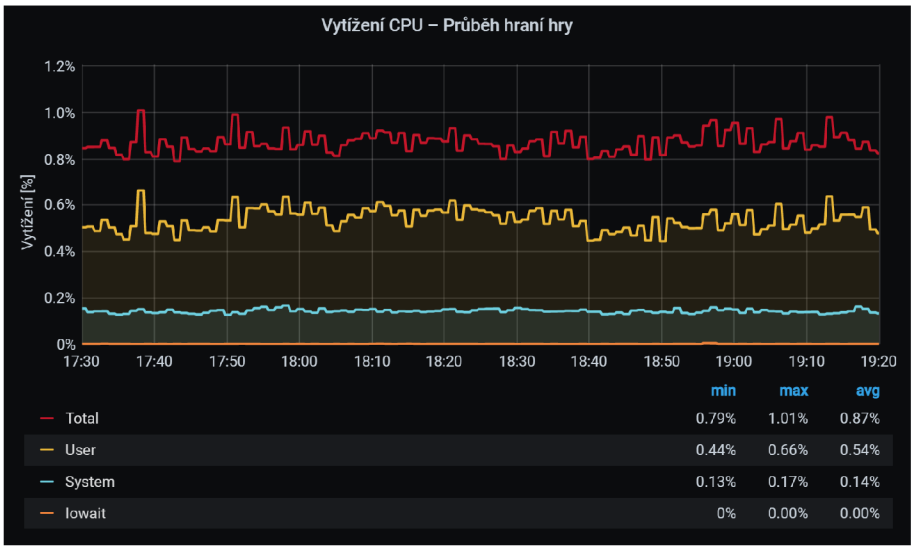
4.2 Měření HW při hraní kybernetické hry

Tyto měření budou postupně graficky znázorněny a popsány.

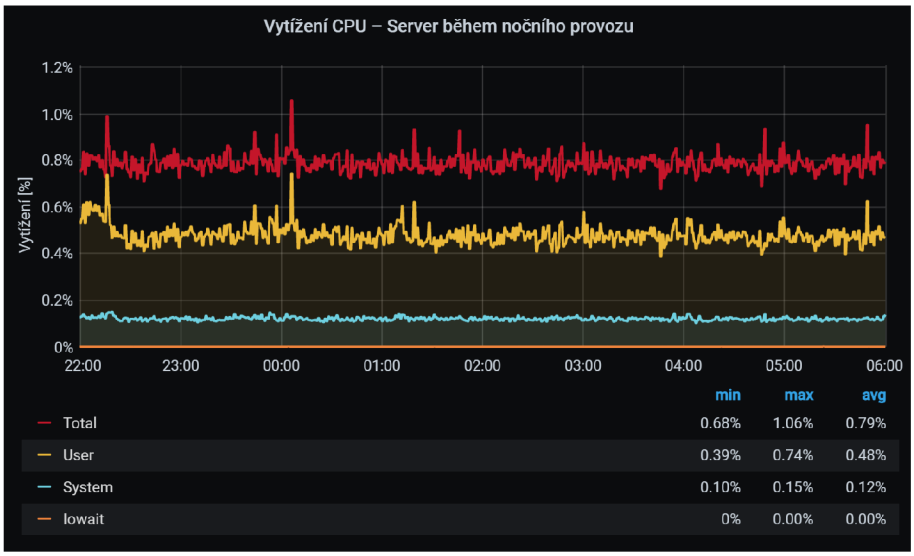


Obr. 4.1: Využití RAM při hraní kybernetické hry

Na obrázku 4.1 lze vidět využití paměti RAM. V závislosti na čase po dobu hraní hry lze vypozorovat, že v průběhu hraní se vyskytly výchylky až ke konci hraní. V této části byla využívána služba telnet a probíhalo na virtuálním stroji `Admin PC` k eskalaci práv. Tyto skutečnosti pak mohly, zapříčinit tuto výchylku.

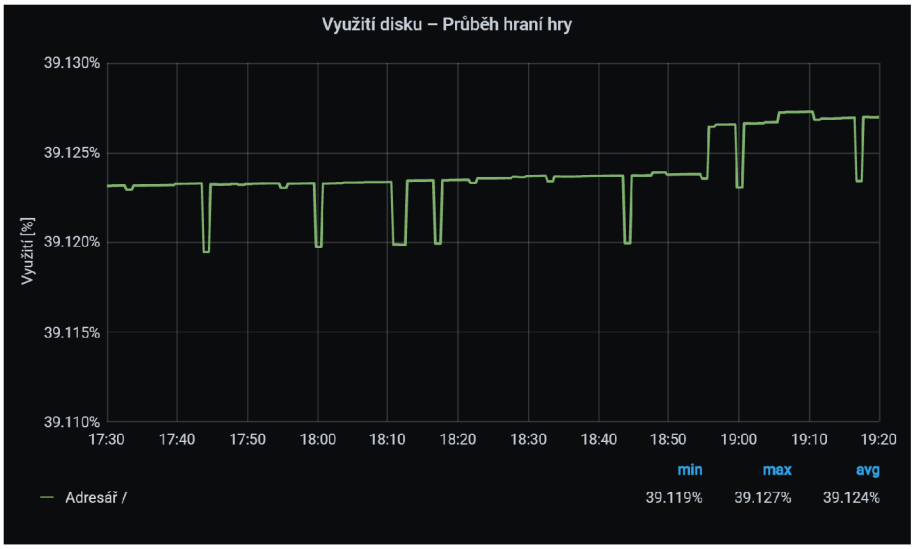


Obr. 4.2: Využití CPU při hraní kybernetické hry



Obr. 4.3: Využití CPU v klidovém stavu serveru

Jak lze na obrázku 4.2 vypozorovat, při průběhu hraní hry nedošlo k žádnému většímu pozorovatelnému vychýlení ve vytížení procesoru. Toto vytížení si můžeme porovnat oproti klidovému stavu, jenž můžeme vidět na obrázku 4.3. Při odečtení průměrné hodnoty vytížení CPU od průměrně hodnoty vytížení při hraní hry, je získána hodnota 0,08%. Což je téměř nepatrný rozdíl.



Obr. 4.4: Využití disku při hraní kybernetické hry

Taktéž na tomto grafu 4.4 nelze vypozorovat větší změny. Drobné výchylky se objevovali v průběhu řešení. Tyto výchylky jsou však minimální a není nutné se jimi

zabývat. Menší výkyv zase přišel ke konci úlohy, kdy je nutné stáhnout exploit a ten spustit. To mohlo zapříčinit větší využití disku.

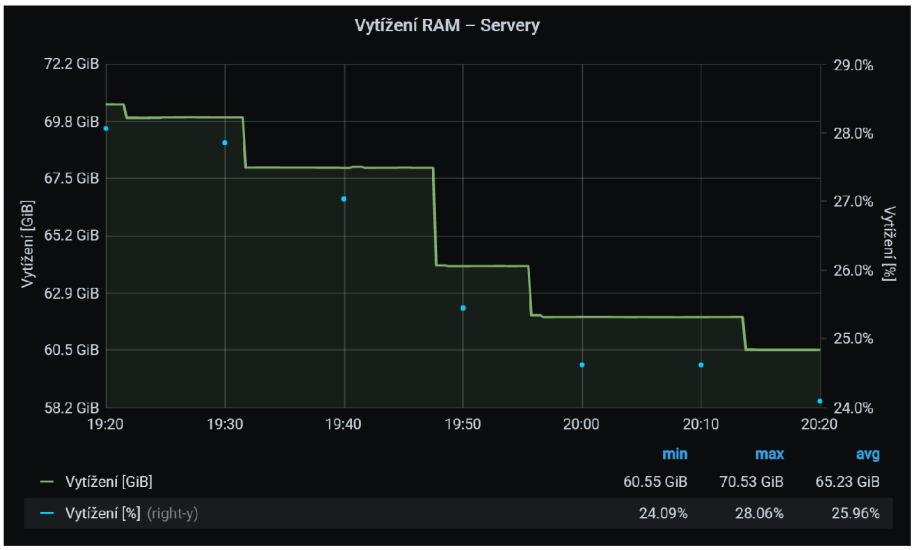
Obr. 4.5: Síťový provoz při hraní kybernetické hry

Tento graf 4.5 je ten nejzajímavější a je možné se z něj dozvědět hned několik informací. Největší výchylky tohoto grafu byly způsobeny připojováním se na služby virtuálních strojů a spouštěním exploitů. Taktéž poslední výchylka znázorňuje připojení se za pomocí služby telnet.

4.3 Měření hardwarové náročnosti serverů

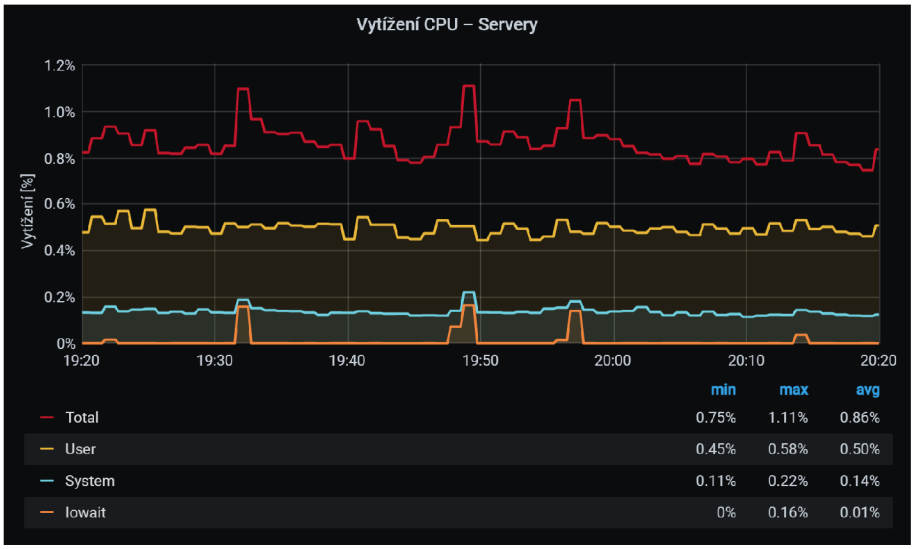
Hardwarová náročnost serverů byla měřena tak, že byly jednotlivé servery vypnuty a byly z nich uvolněny veškeré zdroje. Tento proces se v angličtině nazývá `shelve`. Pro účely této práce bude proces nazýván jako archivace. Následně byly porovnány stavy před a po tomto procesu. Servery byly archivovány v následujícím pořadí:

1. **Admin PC**
2. **Windows phpMyAdmin**
3. **Windows SMB**
4. **Ubuntu VPN**
5. **Fedora WEB**



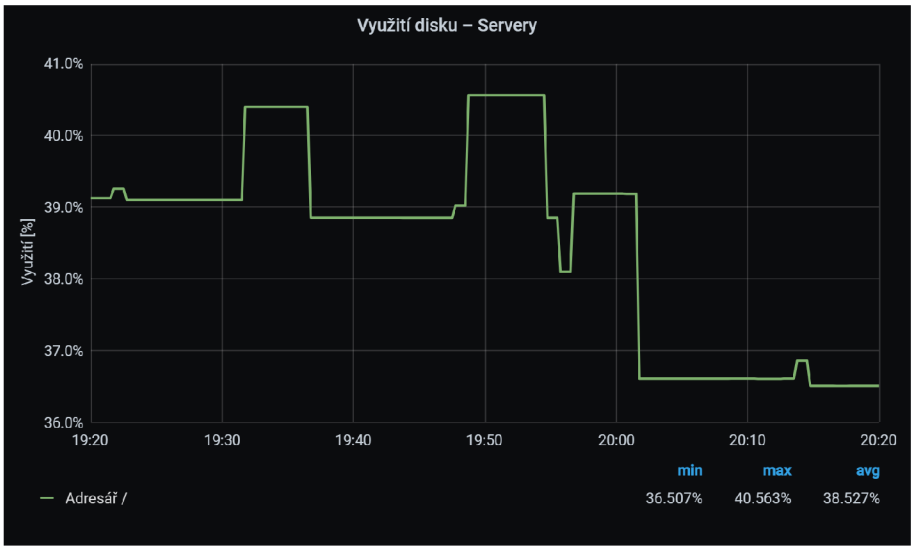
Obr. 4.6: Vytíženost RAM při uspávání instancí

Z grafu 4.6 lze jednoduše vyčíst, kdy byly jednotlivé instance archivovány. Graf je klesajícího charakteru, jelikož při archivace instance se uvolní paměť, kterou lze následně využívat pro jiné účely. Celkově virtuální stroje spotřebovaly deset gigabitů paměti. Nejvíce paměti pak vyžadují servery `Windows phpMyAdmin`, `Windows SMB` a `Ubuntu VPN`.



Obr. 4.7: Využití CPU při uspávání instancí

Na grafu 4.7 využití procesoru lze taktéž vidět drobné výchylky. Ty jsou dány ukládáním instance. Tyto výchylky jsou nejvíce znatelné pro servery `Windows SMB` a `Ubuntu VPN`.



Obr. 4.8: Využití disku při uspávání instancí

Z grafu 4.8 vidíme, že servery využívají zhruba čtyři procenta disku. Nejnáročnější servery se jeví stejně jako v předešlých případech `Windows phpMyAdmin`, `Windows SMB` a `Ubuntu VPN`.

Obr. 4.9: Vytížení sítě při uspávání instancí

Z grafu 4.9 lze jednoduše vyčíst, že zdrojově nejnáročnější instance zmíněné již několikrát výše, vyžadují více zdrojů. To včetně síťového provozu, kde vidíme rychlost odesílání až 73 MiB. Zatím co méně náročné instance vyžadují kolem 18 až 38 MiB.

4.4 Shrnutí měření

Měření prozradilo hned několik skutečností. Jako nejdůležitější fakt, že nejnáročnější virtuální servery jsou `Windows phpMyAdmin`, `Windows SMB` a `Ubuntu VPN`. To se dalo předpokládat vzhledem k tomu, že Windows distribuce obsahuje mimo jiné grafické rozhraní a jeho obraz je největší. `Ubuntu VPN` je náročnější vzhledem k jeho účelu a taktéž nebyl zvolen nejmenší možný obraz této distribuce. Nejvíce zdrojů je v úloze přiděleno virtuálnímu serveru `Fedora WEB`, což rozhodně není vhodné a doporučuje se vzhledem k tomu zdroje omezit.

Přidělené zdroje jednotlivým virtuálním serverům jsou v tabulce níže:

Název serveru	CPU	RAM (GiB)	Disk (GiB)
Admin PC	1	1	10
Win phpMyAdmin	2	2	20
Win SMB	2	4	40
Ubuntu VPN	2	2	20
Fedora WEB	4	4	30

Reálné zdroje, které virtuální servery využijí jsou v tabulce níže:

Název serveru	CPU	RAM (GiB)	Disk (GiB)
Admin PC	nelze	0,51	0,194
Win phpMyAdmin	nelze	2,14	8,14
Win SMB	nelze	4	6,1
Ubuntu VPN	nelze	2	12,07
Fedora WEB	nelze	2,8	0,86

Z tabulek lze vyčíst, že zdroje nejsou úplně ideálně rozloženy a v budoucnu by mělo dojít k úpravě. Taktéž by neměl nastat jakýkoliv problém při replikaci zařízení tak, aby úlohu mohlo řešit více studentů zároveň.

Závěr

Hlavním cílem práce bylo navrhnout kybernetickou hru, která bude sloužit jako úvod do bezpečnostního testování. V první kapitole práce seznamuje se související teorií. Jsou zde vysvětleny pojmy informační, kybernetická a počítačová bezpečnost. Na tyto pojmy následně navazují hrozby a zranitelnosti (v České republice a ve světě), se kterými se lze v dnešní době nejčastěji setkat. Dále jsou zde vysvětleny základní postupy a techniky bezpečnostního testování, které jsou taktéž v dnešní době hojně využívány. Velmi dobrá metoda, jak edukovat veřejnost proti takovýmto hrozbám, či zranitelnostem je právě formou zábavy pomocí bezpečnostních her. V teorii jsou proto nejznámější typy bezpečnostních her uvedeny a vysvětleny, na jakém principu fungují. Na závěr této kapitoly jsou popsány možnosti využití OpenStacku v laboratoři VUT Brno.

Na základě nabytých znalostí v možnosti využití OpenStack systému a typů bezpečnostních her byla navržena laboratorní úloha. Úloha obsahuje strukturovanou síť s několika virtuálními stroji, které běží na různých platformách a obsahují známé zranitelnosti. Hra je navržena obdobně jako **CTF**, kde se řešitel snaží dopátrat vlajky splněním úkolu. Odměna řešitele vlajkou přichází v bodě, kdy převezme kontrolu nad zařízením a bude schopen se jí v systému dopátrat. Části hry jsou zamýšleny tak, aby postupně zvyšovaly nároky na řešitele a ten byl nucen zkoumat danou problematiku více do hloubky.

V realizaci této úlohy bylo potřebné se především naučit s platformou OpenStack a operačními systémy, které byly v úloze využity. Následně mohlo dojít k vytváření infrastruktury na zvolené platformě a konfiguraci jednotlivých virtuálních serverů.

V poslední části byl vypracován návod pro laboratorní cvičení, který je ve dvou vyhotoveních a to pro žáky a cvičícího.

Literatura

[1] *Proč řešit kybernetickou bezpečnost.* [online]. V Ostrava: AUTOCONT, 2017 [cit. 2020-11-11]. Dostupné z URL: <https://www.autocont.cz/aktuality/openspace/kyberneticka-bezpecnost/proc>

[2] *Co je compliance.* [online]. V Praze 3: Wolters Kluwer a.s, 2020 [cit. 2020-11-11]. Dostupné z URL: <https://www.incompliance.cz/cz/compliance>

[3] PCI DSS [online]. V Praze: TAYLLORCOX, c2019 [cit. 2020-11-11]. Dostupné z URL: <https://www.tayllorcox.cz/audit/ostatni-certifikace/pci-dss-compliance>

[4] *What is cybersecurity, and why is it needed.* [online]. Utah: Braintrace, c2020 [cit. 2020-11-24]. Dostupné z URL: <https://braintrace.com/what-is-cybersecurity-and-why-is-it-needed/>

[5] FRUHLINGER, J. *What is information security ? Definition, principles, and jobs.* [online]. Framingham: IDG Communications, c2020 [cit. 2020-11-24]. Dostupné z URL: <https://www.csoonline.com/article/3513899/what-is-information-security-definition-principles-and-jobs.html>

[6] HUDEC, L. *Úvod do informačnej bezpečnosti.* [online]. Bratislava [cit. 2020-11-24]. Dostupné z URL: <http://www2.fiit.stuba.sk/~lhudec/PIS/1_tema.ppt>

[7] *Zpráva o stavu kybernetické bezpečnosti ČR - 2019.* [online] NÚKIB, 2020 [cit. 2020-11-11]. Dostupné z URL: <https://www.nukib.cz/download/publikace/zpravy_o_stavu/NUKIB_ZSKB_2019.pdf>

[8] KOLOUCH, J. *Cybercrime.* V Praze: CZ.NIC, 2016. ISBN 978-80-88168-15-7.

[9] MALINA, L. *Útoky DDoS a testování bezpečnosti a výkonnosti sítě.* Moodle.vutbr.cz [online]. Brno: Vysoké učení technické v Brně, 2019, 2019 [cit. 2020-11-15]. Dostupné z URL: <moodle.vutbr.cz>

[10] *OWASP Top Ten.* [online] Maryland: OWASP Foundation, c2020 [cit. 2020-11-14]. Dostupné z URL: <https://owasp.org/www-project-top-ten/>

[11] *XML External Entity (XXE).* [online] SOOM.cz, c2003–2020 [cit. 2020-11-14]. Dostupné z URL: <https://www.soom.cz/lexikon-webovych-zranitelnosti/3--XML-External-Entity-XXE>

[12] KÜMMEL, R. *Cross-Site Scripting v praxi.* Zlín: Tigris spol. s r o., 2011. ISBN 978-80-86062-34-1.

[13] *Bezpečnostní testy.* Bdo-it [online]. Praha: BDO IT, 2014 [cit. 2020-11-03]. Dostupné z URL: <http://bdo-it.cz/cz/bezpecnostni-testy>.

[14] *What is Security Testing?* Guru99 [online]. Ahmedabad: Guru99 Tech Pvt, 2020 [cit. 2020-11-03]. Dostupné z URL: <https://www.guru99.com/what-is-security-testing.html>

[15] Danel, R. *Analýza rizik.* [online] In: . Ostrava: VŠB – TU Ostrava [cit. 2020-11-15]. Dostupné z URL: <http://homel.vsb.cz/~dan11/bis/BIS_Danel_analyza_rizik.pptx>

[16] Bezpečnostní analýza [online]. Praha 2: Risk Analysis Consultants, 2014 [cit. 2020-11-10]. Dostupné z: < https://www.rac.cz/rac/homepage.nsf/CZ/SS/$FILE/RAC%20Bezpecnostni%20analyza_Datasheet_CZ_141015.pdf >

[17] *Skenování zranitelností webových aplikací a interní infrastruktury.* [online] V Brně: EO SECURITY, 2016 [cit. 2020-11-10]. Dostupné z URL: <https://eo-security.cz/sluzby/skenovani-zranitelnosti/>

[18] SELECKÝ, M. *Penetrační testy a exploitace.* V Brně: Computer Press, 2012. ISBN 978-80-251-3752-9.

[19] *PTWA: Software a bezpečnost.* [online] SOOM.cz, c2003–2020 [cit. 2020-11-26]. Dostupné z URL: <https://www.soom.cz/clanky/1190--PTWA-Software-a-bezpecnost>

[20] *Bezpečnostní IT audit.* [online] Praha: Blue Partners, c2016 [cit. 2020-11-26]. Dostupné z URL: <https://www.bluepartners.cz/cz/slovnik/bezpecnostni-it-audit/>

[21] *Penetration Testing Vs. Ethical Hacking.* [online] Haidarábád: Tutorials Point India Limited, c2020 [cit. 2020-11-26]. Dostupné z URL: <https://www.tutorialspoint.com/penetration_testing/penetration_testing_vs_ethical_hacking.htm>

[22] *How Does an IT Audit Differ From a Security Assessment?* [online] Austin: SolarWinds Worldwide, c2020 [cit. 2020-11-27]. Dostupné z URL: <https://www.dnsstuff.com/it-security-audit-vs-security-assessment>

[23] *Jak funguje OpenStack a šest důvodů, proč v něm mít cloud* [online]. Brno: Master Internet s.r.o, c2020 [cit. 2020-11-29]. Dostupné z URL: <https://www.master.cz/blog/co-je-openstack-jak-funguje-vyhody-openstacku/>

[24] *Cloudová platforma OpenStack.* [online]. Brno: CCB, spol. s r.o., 2018 [cit. 2020-11-29]. Dostupné z URL: <https://www.linuxexpres.cz/software/cloudova-platforma-openstack>

[25] JACKSON, K; CODY, B; EGLE, S. *OpenStack Cloud Computing Cookbook.* 3rd edition. Birmingham: Packt Publishing, 2015. ISBN 978-1-78217-478-3.

[26] FIFIELD, T; DIANE, F; ANNE, G; LORIN, H; JONATHAN, P; EVERETT, T; JOE, T. *OpenStack Operations Guide.* Sebastopol: O'Reilly Media, 2014. ISBN 9781491946954.

[27] BROWN, E. *Security essentials*: *What is capture the flag hacking?* [online]. AT&T, 2019 [cit. 2020-12-04]. Dostupné z URL: <https://cybersecurity.att.com/blogs/security-essentials/capture-the-flag-ctf-what-is-it-for-a-newbie>

Seznam symbolů, veličin a zkratek

CTF	Capture the flag
CVE	Common Vulnerabilitiesand Exposures
DDoS	Distributed Denial of Service
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name System
DoS	Denial of Service
GDPR	obecné nařízení na ochranu osobních údajů
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
ISECOM	Institute for Security and Open Methodologies
OML	open methodology license
OSS	open-source software
OSSTMM	Open Source Security Testing Methodology Manual
OWASP	Open Web Application Security Project
PCI-DSS	Payment Card Industry Data Security Standard
SMB	Server Message Block
QCOW2	QEMU Emulator
PHP	HyperText Preprocessor
RDP	Remote Desktop
SSH	Secure Shell
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
VPN	Virtual Private Network
WSL	Windows Subsystem for Linux

XML eXtensible Markup Language

XSS Cross-site scripting

A Hra kybernetické bezpečnosti na platformě OpenStack - verze pro studenty

Úvod do kybernetické hry

Celý večer sedíš u počítače, díváš se na vtipné videa s kočkami a nahlas přemýšlíš co se tak mohlo pokazit. Jak je možné, že tě jen tak ze dne na den vyhodí se slovy, "Již nejste potřeba, naše firma šetří finance a vy jste pro nás zbytečná položka na seznamu". Několik let se jim staráš o bezpečnost a oni se takhle odvděčí. Víš, že to takhle nemůžeš nechat. Existuje jediná možnost a to ukázat jim co jsi zač. Nabourej se do počítače správce sítě a ukaž mu, že jsi mnohem víc potřebný, než si vůbec dokáže představit.

Cíl

Cílem laboratorní úlohy je seznámit studenty s problematikou bezpečnostního a penetračního testování. Zejména se bude jednat o bezpečnost webových aplikací a exploitace. Student se naučí pracovat s běžně používanými nástroji jako jsou NMAP či Metasploit. Absolvováním této laboratorní úlohy student získá základní přehled v dané oblasti a dokáže řešit problémy, jenž jsou vystavěny na obdobném principu jako v této laboratorní úloze.

Úkoly

1. Projděte si teoretický úvod a seznamte se s nástroji NMAP a Metasploit.
2. V operačním systému Kali Linux nakonfigurujte server VPN a připojte se k OpenStack serveru.
3. Získejte přístup do systému prvního virtuálního stroje a postupujte dál v úloze.
4. Získejte všechny vlajky z napadených systémů.

A.1 Teoretický úvod

A.1.1 Bezpečnostní zranitelnosti a exploitace

V této laboratorní úloze se budeme věnovat jen určitým typům zranitelností a exploitací. Vzhledem k tomu je tento teoretický úvod přizpůsoben.

SQL injekce

Injekce je nejčastější typ útoku na webovou aplikaci. Využívá vložení škodlivého kódu, přesněji databázového dotazu, který je pak zaslán interpretu[1]. Interpret může tento dotaz přeložit nesprávně a může tak dojít k úniku, či ztrátě dat, popřípadě ke spuštění nežádoucího kódu.

V praxi pak tato zranitelnost může vypadat tak, že v popředí se nachází tento standardní přihlašovací formulář požadující od uživatele jméno a heslo. Tento formulář je naprogramovaný tak, že v popředí aplikace se nachází tento HTML[2] (HyperText Markup Language) kód.

```
<form action='index.php' method="post">
<input type="text" name="jmeno" required="required"/>
<input type="password" name="heslo"/>
<input type="submit" value="Submit"/>
</form>
```

Formulář má za úkol zaslat uživatelem zadané jméno a heslo do pozadí aplikace, PHP[3] (HyperText Markup Language) souboru, jenž obsahuje tento databázový dotaz.

```
SELECT * FROM uzivatele WHERE jmeno = $_POST['jmeno']
AND heslo = md5($_POST['heslo']);
```

Zde si můžeme všimnout, kde nastává problém. Zadané jméno a heslo se totiž vkládá přímo do dotazu. Jelikož neproběhne kontrola, dotaz může útočník jakkoliv kompromitovat. V konečném důsledku může takový kompromitovaný dotaz vypadat například následovně.

```
SELECT * FROM uzivatele WHERE jmeno = 'administrator'
AND heslo = '' or 1=1--'
```

V tomto případě zadal uživatel údaje jako jméno: `administrator` a heslo: `' or 1=1--`. Podmínka `or 1=1` bude vždy splněna a pokud takový dotaz nebude ošetřen, lze se přihlásit na jakéhokoliv uživatele v této situaci jako administrátor. Ostatní znaky slouží k zachovaní správné syntaxi SQL, kde dvě pomlčky označují komentář. Tato syntaxe se může lišit vzhledem k použitému dotazovacího jazyku a kódu.

Exploitace

Eploitace programu je základní technika etického hackingu a hackingu obecně. Počítačové programy jsou tvořeny komplexními příkazy, které se postupně spouštějí a říkají počítači co má dělat. Exploitace takového programu je v podstatě postup

[1] Program, který překládá a spouští zdrojový kód řádek po řádku.
[2] Značkovací jazyk, používaný pro tvorbu webových stránek.
[3] Skriptovací jazyk pro tvorbu pozadí aplikace, její logiky.

jak přimět počítač, aby vykonal příkazy, které útočník požaduje. A to i v případě, že běžící program byl navrhnut, aby tomuto zabránil.

Jelikož program může dělat skutečně jen to k čemu je navržen, tak bezpečnostní mezery jsou vlastně chyby v návrhu programu, implementaci či v prostředí, kde program běží. Vzhledem k tomuto faktu je nutné počítat s tím, že k překonání takové bezpečnostní mezery je třeba komplexních znalostí informatiky a kreativní mysli, která takovou bezpečnostní chybu dokáže odhalit.

V této laboratorní úloze jsou všechny zranitelnosti vytvořeny tak, aby student nebyl nucen tyto exploitace vytvářet ani složitými způsoby zjišťovat, jestli je daná komponenta zranitelná. Všechny informace jsou jednoduše vyhledatelné a to ať už v programu metasploit, tak na internetu. Chyby jsou vytvořeny na bázi zastaralých služeb systému, operačního systému či nedostatečném zabezpečení [1].

A.1.2 NMAP

Tato část má za úkol vysvětlit základní techniky skenování sítě pomocí nástroje NMAP (Network MAPper). NMAP je open-source nástroj pro skenování sítě a bezpečnostní auditing. Tento nástroj je využíván nejčastěji síťovými administrátory, penetračními testery, ale samozřejmě taktéž narušiteli počítačových sítí. NMAP pomocí obyčejných IP paketů dokáže zjistit, kteří hostitelé jsou v síti k dispozici, jaké služby nabízejí a jaké operační systémy používají. Taktéž dokáže zjistit typ filtrů paketů, jaký firewall se používá a další. Tento nástroj byl navržen předně pro rychlé skenování rozsáhlých sítí, ale funguje dobře taktéž vůči jednotlivým hostitelům.

Příkaz NMAP může obsahovat celkově tři vstupy a to typ, parametry a cíl skenu. Avšak nutně musí být vyplněn pouze cíl skenu, bez kterého nebude příkaz fungovat. Základní syntaxe vypadá následovně.

```
nmap [ <Typ skenu> ...] [ <Parametry> ] { <cíl skenu> }
```

Typ skenu uvádí základní informaci o skenování. To je použitý typ paketu a na co se má sken zaměřit. Mezi nejčastější typy patří:

- **-sn** – nalezení hostů, ale nehledá otevřené porty.
- **-sS** – SYN sken.
- **-sT** – TCP sken.
- **-sU** – UDP sken.
- **-O** – detekce operačního systému.

Parametry udávají pokročilé možnosti skenu. Těchto parametrů je velké množství, proto jsou uváděny jen ty základní.

- **-sC** – spouští výchozí scripty.
- **-sV** – detekce verzí služeb.
- **-Pn** – předpokládá že všichni hosti jsou online.

- **-p** – skenuje jednotlivě vyčtené porty popřípadě rozsah.
- **-p-** – skenuje všechny porty.

Cíl skenu se zadává jako IP adresa cíle. Může být dán jako samotná IP adresa, nebo jako rozsah adres. Taktéž můžeme využít parametry pro specifikování cíle.

- **-iL** – vyčítá IP adresy z textového souboru.
- **-iR** – vybírá náhodné cíle v rozsahu.
- **--exclude** – vynechá ze skenu zadané adresy.
- **--excludefile** – vyčítá IP adresy z textového souboru a ty vynechá ze skenu.

Příklady skenování pomocí NMAP

Zde budou ukázány příklady užití a to od těch nejjednodušších až po ty více komplexní. Pro testovací potřeby můžete sken zkoušet na lokálním hostu, to je adresa 127.0.0.1.

Oskenování lokálního hosta 127.0.0.1 za užití výchozího skenu a parametrů.

```
nmap 127.0.0.1
```

Spuštění TCP skenu na všechny hosty v rozsahu 127.0.0.0 až 127.0.0.255. Následné zjištění jestli je spuštěná na jejich straně nějaká ze služeb v tomhle případě SSH, telnet, HTTP, HTTPS nebo Battle.net.

```
nmap -sV -p 22,23,80,443,6112 127.0.0.1/24
```

Spuštění skenu zařízení s předpokladem, že je dostupný. Využije k tomu defaultní skripty a detekuje verze služeb, které na něm běží [2].

```
nmap -Pn -sC -sV 127.0.0.1
```

A.1.3 Metasploit

Metasploit je open-source nástroj pro vývoj, testování a použití exploitů. Primární účel tohoto nástroje je penetrační testování, vývoj a výzkum zranitelností. Poskytuje možnost exploitace jednotlivých cílů a následný monitoring, nahrání souborů či vytvoření takzvaně zadních vrátek. Využívá techniky, které jsou vypsány níže.

- **Autentizační útoky**
- **Útoky hrubou silou**
- **SQL injekce**
- **Exploitace**

Metasploit moduly

- **Auxiliary** jsou moduly, které obsahují užitečné programy jako fuzzier, skenery, nástroje na SQL injekci.

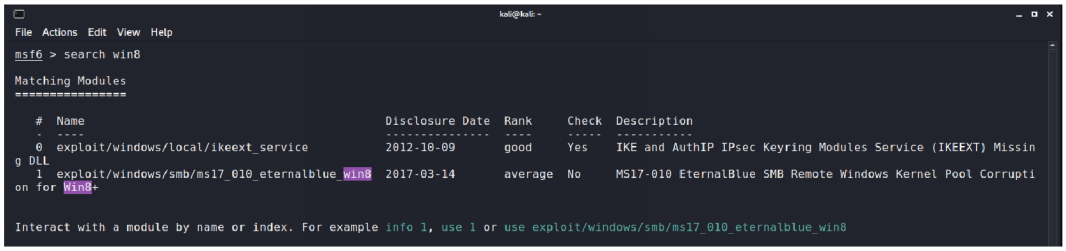
Obr. A.1: Moduly metasploitu

- **Encoders** zajišťují, že payloady dorazí do cíle v pořádku a uceleně.
- **Exploits** jsou moduly, které se spouští na cílovém zařízení.
- **Nops** jsou instrukce napsané v assembleru.
- **Payloads** obsahují sety instrukcí, které jsou odeslány na cílové zařízení. Můžou obsahovat od kousků kódů, až po malé aplikace.
- **Post** umožňují testerovi shromažďovat data z napadených počítačů.

Výpis jednotlivých modelů a obsah naleznete na obrázku A.1.

Základní příkazy metasploitu

- **search** – Jelikož metasploit obsahuje přes 2000 exploitů, tak pro nalezení toho nejvhodnějšího je nejjednodušší použít příkaz `search`. Pro vyhledání exploitu pro win8 vypadá příkaz jako na obrázku A.2.



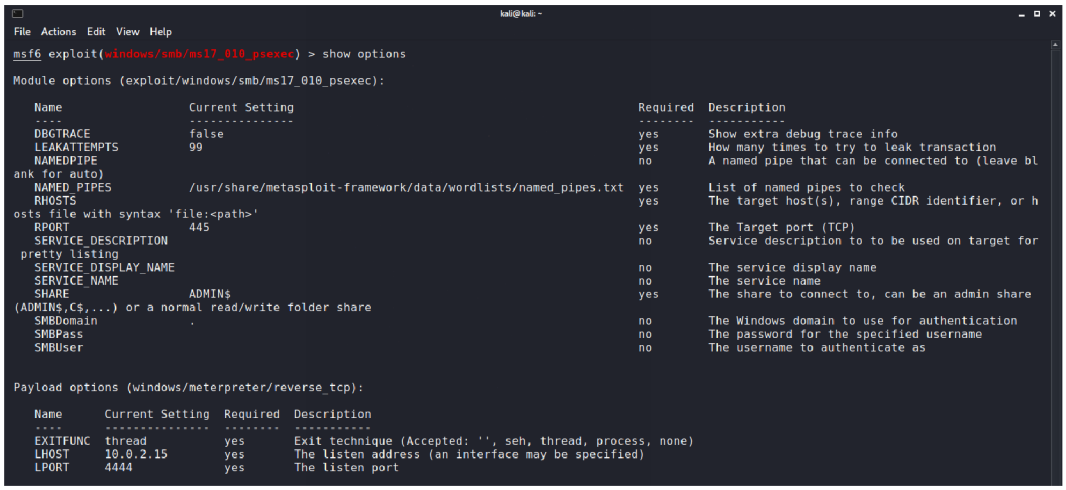
Obr. A.2: Vyhledání exploitu pomocí search

- **help** – Tento příkaz vypíše příkazy, které lze v nástroji použít.

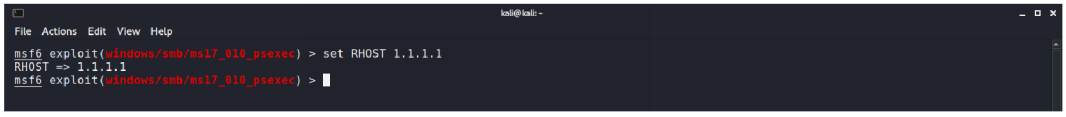
- **use** – Příkaz use se používá k aktivování určitého modulu. Pro aktivování například exploitu pro zranitelnost ms17_010, bude příkaz vypadat jako na obrázku A.3.

Obr. A.3: Aktivování exploitu pomocí příkazu use

- **show options** a **set** – Tyto příkazy slouží pro zobrazení a nastavení parametrů zvoleného exploitu. Příklad zobrazení a nastavení je na obrázku A.4 a A.5. Mezi základní parametry patří RHOST, LHOST, RPORT a LPORT, které udávají adresu hosta, cíle a jejich porty.



Obr. A.4: Zobrazení parametrů exploitu



Obr. A.5: Nastavení parametrů exploitu

- **db_nmap** – Je příkaz stejný jak klasický NMAP vysvětlený výše.

A.2 Postup řešení

A.2.1 Potřebné nástroje.

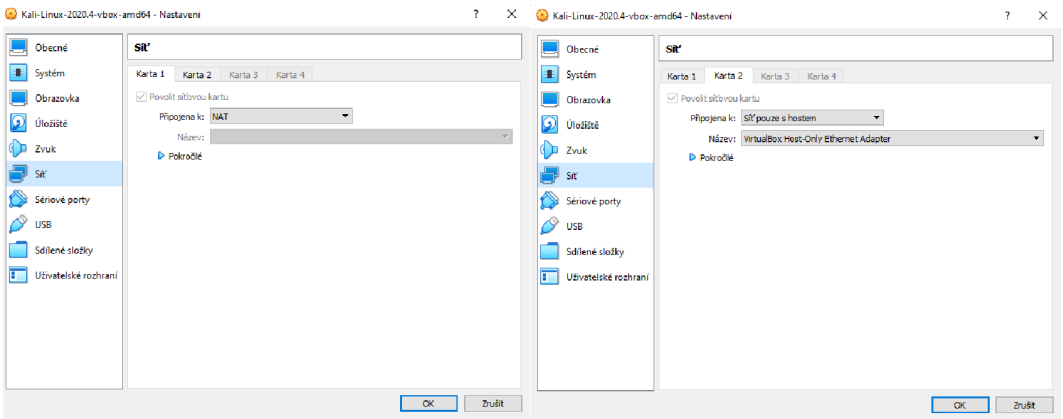
K provedení laboratorní úlohy budeme potřebovat následující nástroje.

1. **Virtualizovaný Kali Linux ve Virtualboxu**
2. **Přístupové údaje k OpenStack serveru**
3. **Připojení k internetu.**

A.2.2 Nastavení VPN serveru

Pro připojení k openstack serveru a řešení úlohy, je nutné mít ve Virtualboxu správně nastavené síťové rozhraní virtuálního stroje. Rozhraní je nastaveno tak, aby byly aktivovány dvě síťové karty s tím, že jedna je připojena k NAT a druhá k síti pouze s hostem, stejně jak na obrázku A.6. Nastavení poslouží taktéž v pozdější části úlohy, kdy bude nutné využívat tzv. VPN chaining. VPN pro připojení k OpenStack serveru nastavíme v Kali linuxu v záložce Síťových připojení -> VPN. Pro VPN nastavujeme parametry rozhraní ethernet, adresu serveru, typ sítě, před sdílený klíč a přihlašovací údaje. Tyto údaje jsou v rámci bakalářské práce utajeny a budou studentovi sděleny před vypracováním úlohy.

Po připojení na server VPN vyzkoušíme ping na zařízení s IP adresou 10.0.0.1. Při zdařilém průchodu je VPN nastavena správně.



Obr. A.6: Nastavení síťového rozhraní virtuálního stroje

A.2.3 Metodika řešení

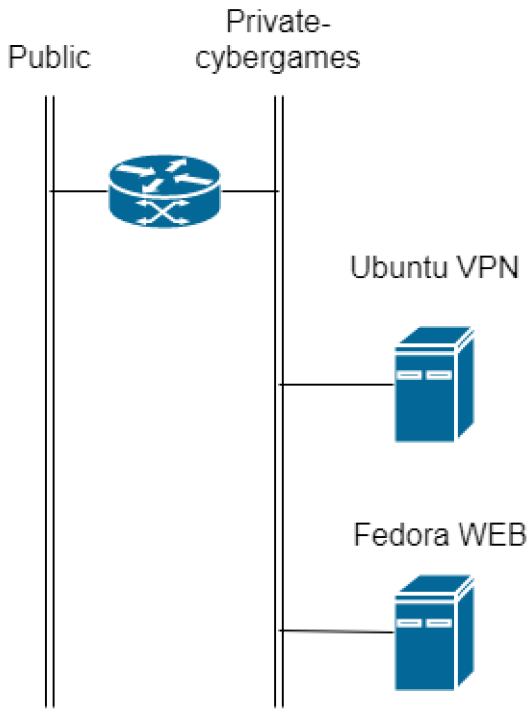
Pro postup v laboratorní úloze se doporučuje následná metodologie.

1. Sken zařízení a sítě pomocí nástroje NMAP.

2. Zjištění základních informací o jeho operačním systému a službách.
3. Kontrola jestli dané informace znamenají potenciální zranitelnost.
4. V metasploitu vyhledat exploit, případně ho najít na internetu.
5. Získat kontrolu nad systémem a najít vlajku.

A.2.4 Postup úlohou

Po přípojení a prvotním skenu sítě, nacházíš dvě zařízení. Tyto zařízení jsou zobrazeny na obrázku A.7.



Obr. A.7: Topologie sítě

Fedora WEB

Musíš se dostat na server `Fedora Web`. tam jsou informace, které potřebuješ k připojení k privátní síti. K té vede cesta přes `Ubuntu VPN`. Tento server slouží čistě jako VPN. Nemělo by smysl zde hledat nějaké další informace. Navíc je to zařízení, které si tvořil ty, takže víš, že neobsahuje žádnou zranitelnost.

Řešte servery dle metodologie vypsanou výše. Postupujte úlohou, zapisujte si nalezené vlajky a zranitelnosti do tabulky, kterou lze nalézt níže. Na tomto zařízení se nachází jedna vlajka a certifikát, který slouží k připojení do privátní sítě firmy. Při zaseknutí se na některé z částí můžete požádat vyučujícího o nápovědu.

Ke každé úloze jsou vypracovány nápovědy, za které se strhává dle jejich váhy určitý počet bodů.

Ubuntu VPN

Podařilo se ti dostat k důležitým informacím, které si potřeboval. Konečně se můžeš připojit do privátní sítě a pokračovat dál v úloze. Moc dobře si vzpomínáš jak na to. Stačí certifikát stáhnout do soukromé složky a přes terminál spustit příkaz.

```
sudo openvpn cerifikat
```

Prohlédneš si záznam tohoto připojení a vidíš, jaká síť se za VPN serverem nachází. Tuto síť oskenuješ a nacházíš další dvě zařízení. Obě obsahují jednu nebo víc informací, které potřebuješ.

Windows phpMyAdmin

Tenhle systém již znáš, víš že na něm běží webová služba a také, že se o něj hodně dlouhou dobu nikdo nestaral. Zkoušíš nejjednodušší typy útoku a hlídáš si verzi služeb, které na tomto serveru běží.

Windows SMB

Tenhle server bude oříšek. Hned po skenu pozoruješ, že zde není moc možností jak server napadnout. Po chvilce však nacházíš možnou slabinu.

Admin PC

Už jen tento server a máš vyhráno. Tohle bude nejtěžší část, ale pro tebe to určitě nebude žádný problém.

Zařízení a nalezené vlajky

Do tabulky níže zapisujte nalezené informace.

IP ADRESA	ZRANITELNOST	VLAJKA

A.3 Seznam zkratek

HTML HyperText Markup Language

NMAP Network MAPper

PHP HyperText Markup Language

Literatura

[1] ERICKSON, Jon. *Hacking: The art of exploitation.* 2ND EDITION. San Francisco: No Starch Press, 2008. ISBN 1-59327-144-1.

[2] LYON, Gordon. *NMAP network scanning. Sunnyvale: Insecure.Com LLC.* , 2008. ISBN 978-0-9799587-1-7.