Univerzita Hradec Králové
Fakulta informatiky a managementu
Katedra informačních technologií

Využití Kali Linux pro analýzu zranitelností

Bakalářská Práce

Autor: Vyacheslav Novak
Studijní obor: Aplikovaná informatika

Vedoucí práce: Mgr. Josef Horálek, Ph.D.

Hradec Králové červen 2022

Prohlášení:

Prohlašuji, že jsem bakalářskou/diplomovou práci zpracoval samostatně a s použitím uvedené literatury.

V Hradci Králové dne 28.06.2022 Vyacheslav Novak

Poděkování:

Rád bych poděkoval vedoucímu mé bakalářské práce pánu Mgr. Josefu Horálkovi, Ph.D. za vedení bakalářské práce a za jeho rady.

Anotace

V této bakalářské práci se provede analýza dostupných nástrojů v Kali Linux určených pro analýzu zranitelností, popíše se funkcionalita a případy použití každého nástroje a zranitelnosti pro jejichž objevení byl tento nástroj vytvořen. V praktické části bakalářské práce bude demonstrováno praktické využití daných nástrojů v podobě deseti řešených úloh.

Tato bakalářská práce bude také obsahovat informace o Kali Linux, o důležitosti kybernetické bezpečnosti, běžných zranitelnostech a způsobech ochrany proti nim.

Annotation

Title: Using Kali Linux for analyzing vulnerabilities

In this Bachelor Thesis, the author will analyze the available tools in Kali Linux for vulnerability analysis, provide a description of the functionality and use cases of each tool as well as the vulnerabilities that the tool was made to discover. The practical use cases of the given tools will be demonstrated in the practical part of the Bachelor Thesis.

The Bachelor Thesis will also contain information about Kali Linux, the importance of cybersecurity, common vulnerabilities and ways to protect against them.

Obsah

1 Úvod

Každým rokem přibývá více a více kybernetických útoků. Kybernetické útoky se zvyšují nejen v počtu, ale i v rozsahu a způsobují větší škody. V roce 2021 se stal jeden z největších kyberútoků v USA, kde Colonial Pipeline, největší plynovod v USA, byl zasažen útokem ransomware, který způsobil zastavení provozu plynovodu, nedostatky paliva ve východních státech USA a vyhlášení nouzového stavu v některých státech USA. „Společnost [The Colonial Pipeline Company] zaplatila výkupné požadované hackerskou organizací (4,4 milionu dolarů nebo 75 bitcoinů)“ (převzato z **[1]**)

Tento útok byl možný kvůli nedodržování základních zásad kyberbezpečnosti a dalo by se mu jednoduše zabránit.

Definice pojmu zranitelnost: Pod pojmem zranitelnost v kontextu této práce se uvažují zranitelnosti v počítačových systémech. Podle **[2]**, zranitelnosti se nachází v software, firmware, hardware, nebo služebních komponentách.

Analýza zranitelností je prvním krokem v procesu **VAPT – Vulnerability Assessment and Penetration Testing** (česky ohodnocení zranitelností a penetrační testování) (zpracováno podle **[3]**).

Penetrační testování je „Metodika testování, ve které se hodnotitelé, kteří obvykle pracují pod určitými omezeními, pokoušejí obejít nebo porazit bezpečnostní prvky informačního systému.“ (Převzato z **[4]**)

Na začátku práce se představí operační systém Kali Linux, jeho možnosti a návod na instalaci a aktualizaci. Tento operační systém bude použít při praktické části práce. Protože práce se zabývá analýzou zranitelností, bude provedena rešerše o zranitelnostech a budou také představeny některé běžné zranitelnosti a způsoby, jak jim zabránit. Pak následuje teoretická část, kde se bude mluvit o nástrojích Kali Linux, které jsou určeny pro analýzu zranitelností. Budou popsány jejich hlavní možnosti a účely. V praktické částí budou použity výše uvedené nástroje a jejich funkčnost se ukáže na příkladech ve formě deseti úloh, u kterých bude vždycky uveden konkrétní cíl a návod na jeho splnění.

2 Metody

V této kapitole budou popsány metody využité při tvorbě této bakalářské práce.

V průběhu práce byla použitá pouze zahraniční literatura. Důvod tomu je, že v angličtině je mnohem více materiálu než v češtině, a také skutečnost, že popsaná v této práce témata nejsou charakteristická pouze pro českou republiku – témata popsaná v práci jsou univerzální pro všechny země světa. Z angličtiny byl autorem přeložen všechen text do češtiny.

V kapitole Zranitelnost byla provedena rešerše o pojmu zranitelnost a selektivní rešerše o vybraných pojmech spojených se zranitelnostmi. Při této rešerší byla čerpaná informace z vědeckých článků a důvěryhodných webových stránek, jako například stránka NVD organizací NIST, která patří americké vládě. Podobná rešerše se provedla v kapitole Analýza Zranitelností, ve které byl popsán aktuální stav řešení této problematiky.

V teoretické části byly popsané vybrané nástroje pro analýzu zranitelností dostupných v Kali Linux. U každého nástroje byla dodržována jednotlivá struktura: krátký popis nástroje, kde se nachází v Kali Linux, ukázka rozhraní, hlavní možnosti (a případně základní parametry), možnosti použití. Informace potřebná pro psaní teoretické části byla převzata hlavně z oficiálních dokumentací uvedených nástrojů.

Všechny nástroje uvedené v teoretické části byly pak použity v praktické části. Praktická část probíhala ve formě 10 úloh, přičemž u každé úlohy byl stanoven určitý cíl, který byl pak dosažen autorem zvoleným nástrojem. Pro instalaci Kali Linux byla použita verze virtuálního stroje, která byla instalována s pomocí software VMWare Horizon. Protože zmíněné nástroje potřebují cíl, který budou zkoumat, byly použité speciální nástroje pro cvičení nástrojů penetračního testování: webová aplikace OWASP Juice Shop, webová stránka scanme.nmap.org a operační systém Metasploitable.

V teoretické a praktické části byly použité obrázky ukazující rozhraní nástrojů a práci s nimi. Ty obrázky byly vytvořené autorem na výše uvedeném virtuálním stroji Kali Linux.

3 Kali Linux

„Kali Linux je open-source Linux distribuce založená na Debian zaměřená na různé úkoly v oblasti informační bezpečnosti, jako je penetrační testování, bezpečnostní výzkum, počítačová forenzní analýza a zpětné inženýrství.“ **[5]**

Penetrační testování, podle **[6]**, zahrnuje simulaci skutečných útoků za účelem posouzení rizika souvisejícího s potenciálním narušením bezpečnosti. Při penetračním testování testeři nejen odhalí zranitelnosti, které by mohli útočníci využít, ale také zneužívají slabá místa, pokud možné, aby mohli posoudit, co by mohli útočníci získat po úspěšné exploataci.

Smysl penetračního testování je nalezení a opravení slabin systému dříve, než to bude moci udělat útočník.

Kali Linux je bezplatná a dnes nabízí více než 600 nástrojů pro účely penetračního testování. „Kali Linux je nesporný průmyslový standard, Open-source platforma pro penetrační testování.“ **[7]**

Kali Linux se také liší od jiných distribucí pro penetrační testování následujícími možnostmi (zpracováno podle **[8]**):

- Široká podpora bezdrátových zařízení.
- Odpovídání standardu Filesystem Hierarchy Standard, což umožňuje uživatelům Linuxu snadno najít binární soubory, podpůrné soubory, knihovny atd.
- Podporuje zařízení s procesorem architektury ARM (Advanced RISC Machine) – menší počítače jako například Raspberry PI.
- Je plně přizpůsobitelná, včetně jádra. Uživatelé mohou vytvořit vlastní ISO soubor Kali, což umožňuje například změnit prostředí pracovní plochy (Xfce, Gnome, MATE atd.) na jiný.

Kali Linux má také bohatou oficiální webovou stránku, kde lze najít všechny informace o této distribuci včetně dokumentaci **[9]** a diskusního fóra **[10]**. Adresa webové stránky je **[11]**.

Etika použití nástrojů Kali Linux: V praktické části bakalářské práce se budou používat nástroje Kali Linuxu pro analýzu zranitelností. Tyto nástroje potřebují určitou IP-adresu jako cíl. Webový server, který je hostován na této adrese, bude oběti analýzy zranitelností.

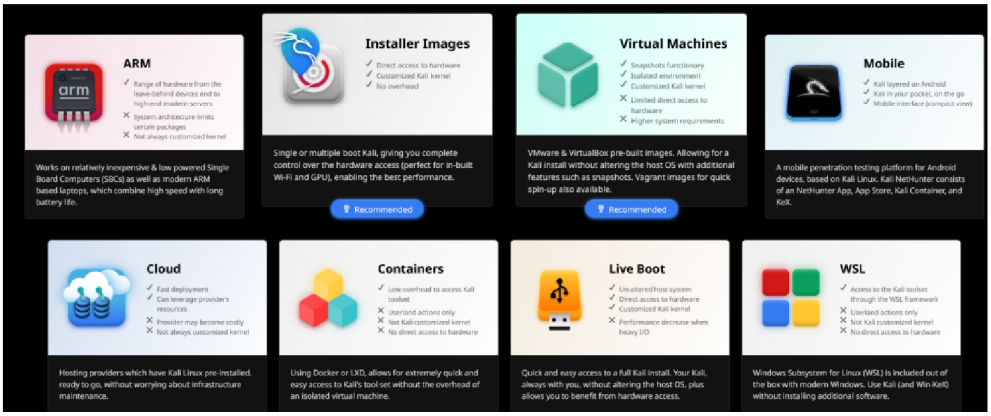
I když se provádí pouze analýza zranitelnosti, považuje se to za kyberútok, což je nelegální bez souhlasu majitele cílové aplikace a vede ke trestnímu činu.

Proto je třeba buď použití nástrojů Kali Linux na vlastním webovém serveru, nebo jejich použití na existující webově stránce, která je speciálně vytvořena pro účel vyzkoušení a procvičení nástrojů analýzy zranitelností.

[12] doporučuje použit aplikaci **OWASP Juice Shop**. Je to moderní web aplikace, která schválně obsahuje mnoho zranitelností pro účely penetračního testování. Tato aplikace se ale musí nastavit jako vlastní webový server. Návod na její nastavení se ukáže v praktické části bakalářské práci.

3.1 Instalace Kali Linux

Oficiální webová stránka Kali Linux nabízí 8 různých způsobů instalace distribuce, v závislosti na zařízení a účelech použití Kali:



Obrázek 1 Dostupné verze Kali. Zdroj: Autor

- **ARM** – Verze pro počítače s procesorem architektury ARM. Jsou to většinou mobilní zařízení jako mobilní telefony a tablety nebo mikropočítače jako Raspberry PI. V kontextu Kali pod pojmem ARM jde spíše o mikropočítače než mobilní telefony. ARM procesory mají výhodu nízké spotřeby energie.
- **Installer Images** – Nejvýkonnější a nejvíce přizpůsobitelná verze. Toto je instalační soubor ve formě ISO souboru, který se instaluje jako typický operační systém. Tato verze je dobrá pro dlouhodobé použití, ale pro její instalaci je třeba si buď nahradit stávající OS nebo nastavit dual-boot (2 operační systémy nainstalované současně na jednom počítači). Pokud se uživatel tím nechce zabývat, je lepší si nainstalovat verzi Virtual Machines nebo Live Boot.
- **Virtual Machines** – verze pro virtuální stroj. Umožňuje instalovat Kali Linux izolovaně od hlavního operačního systému. Má vyšší požadavky na systém a horší výkon než verze Installer Images.
- **Mobile** – Verze pro mobilní telefony. Obsahuje speciální verzi Kali Linux – Kali Nethunter. Ta má speciálně přizpůsobené rozhraní pro mobilní zařízení, které umožňuje uživatelům jednoduše pracovat s konfiguračními soubory s pomocí webového rozhraní. Obsahuje několik speciálních funkcí jako změna MAC adresy Wifi, nástroje pro útoky na Bluetooth a další.
- **Cloud** – Umožňuje si pronajmout Kali Linux na cloudu – v současné době pouze na Amazon AWS. Ta je předem instalována a není třeba se starat o údržbu infrastruktury. Je vytvořená hlavně pro krátkodobé použití.
- **Containers** – Umožňuje nastavit kontejner s použitím software jako Docker. Kontejnery jsou zapouzdřené prostředí, ve kterých lze spustit aplikace, izolované od hlavního operačního systému. Tuto verzi je dobré použit, pokud uživatel nechce instalovat celý operační systém. Má ale nevýhody – neumožňuje přímý přístup do

hardware a bude mít komplikace s příchozím připojením k nástrojům spuštěným v kontejneru.

- **Live Boot** – Přenosná verze, kterou lze instalovat na USB nebo CD či DVD disk. Výhoda je přenosnost a izolovanost od hlavního operačního systému. Může ale mít problém s výkonem.
- **WSL** – WSL je software dostupný na Windows, který umožňuje nainstalovat Linux distribuci na operačním systému Windows ve formě kontejneru – podobně jako virtuální stroj. Vůči virtuálnímu stroji má ale výhodu lepšího výkonu a lepší integraci se systémy Windows.

V této bakalářské práci se používá verze **Virtual Machines**.

Pro instalaci této verzi je třeba si nejdříve nainstalovat software VirtualBox nebo VMWare, které jsou dostupné zadarmo, stáhnout si z oficiální stránky Kali odpovídající soubor podle toho, jestli se používá 32 nebo 64bitový OS a podle nainstalovaného software pro virtuální stroje (VirtualBox nebo VMWAre), pak rozbalit instalační zip soubor a v software virtuálního stroje si zvolit rozbalenou složku s Kali. Defaultní login a heslo virtuálních obrazů Kali je "**kali/kali**".

3.2 Aktualizace Kali Linux

Kali Linux se neustále vyvíjí, a proto je doporučeno operační systém pravidelně aktualizovat. **[11]** doporučuje zkontrolovat aktualizací jednou za několik týdnu, nebo pokud je potřeba instalovat novou verzi nástroje anebo pokud byla vydána bezpečnostní aktualizace.

Návod na aktualizaci:

„Pro aktualizaci Kali, je třeba se nejprve ujistit, že `/etc/apt/sources.list` je správně vyplněn:

```
kali@kali ~  cat /etc/apt/sources.list
#     See     https://www.kali.org/docs/general-use/kali-linux-sources-list-
repositories/
deb http://http.kali.org/kali kali-rolling main contrib non-free

# Additional line for source packages
# deb-src http://http.kali.org/kali kali-rolling main contrib non-free
kali@kali ~
```

Poté je možné spustit následující příkazy, které nás povýší do poslední verzi Kali.“

```
kali@kali ~  sudo apt update
kali@kali ~  sudo apt full-upgrade -y
```

(převzato z **[13]**)

4 Zranitelnost

V této kapitole se provede rešerše nad tématem zranitelnost. V této rešerši bude definován pojem zranitelnost, budou popsány kategorie a způsoby kategorizace zranitelností, jejich životní cyklus a jeho fáze, databáze zranitelností a pojmy CVE, CWE, CVSS atd. příčiny výskytu zranitelností a skenery zranitelností.

Zranitelnost: MITRE definuje zranitelnost jako "vada v software, firmware, hardware či služební komponentě, vyplývající ze slabosti, která může být využitá, což způsobí negativní dopad důvěrnosti, integritě, nebo dostupnosti zasažené komponenty či komponent." **[2]**.

Bohužel neexistuje přesná definice zranitelnosti a každý zdroj popisuje pojem zranitelnost trochu jinak. **[14]** například cituje odlišné definice zranitelnosti od třech různých zdrojů. Každá z nich se liší, ale všechny definice se shodují s tím, že zranitelnost je slabost v určité IT infrastruktuře, která může být někým využita pro určité výhody a škodí této infrastruktuře.

Kategorizace zranitelností: Zranitelnosti lze zařadit do obecných kategorií, podle toho, kde se zranitelnost vyskytuje. **[15]** zařazuje zranitelnosti do následujících kategorií:

- **Hardware**: Citlivost k vlhkosti, prachu, přírodním katastrofám zašpinění nebo i zranitelnosti ve firmware.
- **Software**: Jakákoli zranitelnost v software, příklady jsou SQL injekce, Cross-Site Scripting, injekce http hlaviček aj.
- **Sít**: Nezabezpečená architektura sítě, útoky Man in the Middle, defaultní přihlašovací údaje a nechráněné komunikační linky.
- **Personál**: Zranitelnosti způsobené personálem, úmyslně nebo nikoliv. Příklady: stažení malware prostřednictvím emailových příloh, nedodržování zásad bezpečnosti, špatná správa hesel.
- **Fyzické místo**: Případy, kdy je fyzická oblast má nespolehlivý zdroj energie nebo je vystavená přírodním katastrofám.
- **Organizační**: Nesprávné vnitřní kontroly, chybějící audit, a chybějící plány kontinuity, zabezpečeností či plány reakce na incidenty.

Zranitelnosti lze také kategorizovat do konkrétnějších kategorií podle typu zranitelnosti, jako Cross-Site Request Forgery, SQL injekce atd. Takovou kategorizací se zabývají **databáze zranitelností**, které mají vlastní systémy kategorizaci zranitelností. Příkladem takového systému je **CWE** (Common Weakness Enumeration, česky společný výčet slabostí). O CWE a databázích zranitelností se bude podrobněji mluvit později v této kapitole.

Zdroje Zranitelností: Neexistuje jeden kompletní seznam všech možných zdrojů zranitelností, a ty se také liší podle oblasti jejich vzniku (hardware, software, sítě atd.) nejčastěji zmíněné zdroje zranitelností jsou podle **[16]**: "vady v návrhu, špatná správa zabezpečení, nesprávná implementace, zranitelnosti v internetových technologiích, povaha

činnosti útočníka, složitost opravy zranitelných systémů, omezení efektivity reaktivních řešení a sociální inženýrství“.

Databáze zranitelností: Databáze zranitelností jsou veřejně dostupné seznamy všech veřejně známých zranitelností. Dvě nejznámější databáze jsou dnes **NVD** (National Vulnerability Database, česky národní databáze zranitelností) a **CVE** (Common Vulnerabilities and Exposures, česky běžné zranitelnosti a expozice). NVD je vytvořeno NIST – americkým národním institutem norem a technologie. CVE je vytvořeno společností MITRE. Systém CVE je **synchronizovaný** s NVD a poskytuje NVD svůj seznam zranitelností. NVD na druhou stranu poskytuje další informaci o zranitelnostech ze seznamu CVE, jako ohodnocení zranitelností pomocí systému **CVSS**. CVE lze najít na **[17]** a NVD na **[18]**.

Cílem těchto databází je sbírat informace o objevených zranitelnostech, kategorizovat je, určit jejich dopady a rizika a určit způsoby prevence těchto zranitelností. Každá zranitelnost má přidělené unikátní identifikátor a kategorie.

CWE: “CWE poskytuje společný jazyk pro diskusi, hledání a řešení příčin zranitelností zabezpečení software nalezených v kódu, designu nebo architektuře systému. Každý jednotlivý CWE představuje jeden typ zranitelnosti. CWE je v současné době spravován společností MITRE Corporation.” **[19]**. Seznam CWE se nachází na webové stránce MITRE: **[20]**.

CVSS (Common Vulnerability Scoring System, česky systém ohodnocení běžných zranitelností): CVSS je systém, který je určen pro ohodnocení závažnosti zranitelností. “CVSS jsou počítány na základě několika metrik, které přibližují snadnost zneužití zranitelnosti (možnost vzdáleného přístupu, složitost přístupu a potřeba autentizace), dopad zranitelnosti na důvěrnost, integritu a dostupnost a další faktory. Skóre, stejně jako celkové hodnocení závažnosti, jsou v rozmezí od 0 do 10, přičemž 10 je nejzávaznější” **[21]**.

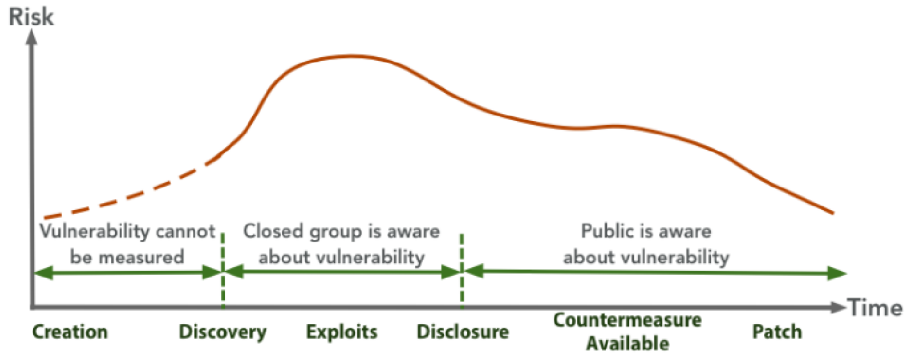
Životní Cyklus Zranitelnosti: Každá zranitelnost má životní cyklus. Ten se skládá z fází – bodů v čase, kterými prochází každá zranitelnost. Každá fáze definuje stav a riziko zranitelnosti. **[22]** definuje následující fáze:

- **Birth (Narození)**: Výskyt softwarového defektu či vady.
- **Discovery (Objevení)**: Zranitelnost je objevena.
- **Disclosure (Odhalení)**: Objevitelé mají možnost odhalit podrobnost o zranitelnosti vývojáři nebo široké veřejnosti.
- **Correction (Korekce) - „patching“**: Zranitelnost je opravena vydáním softwarové modifikaci prodejci software nebo vývojáři.
- **Publicity (Propagace) a Scripting – exploatace**: Zranitelnost může být zveřejněna několika způsoby. Kdokoli se středními dovednostmi může úspěšně zneužít novou zranitelnost.

- **Death (Smrt)**: Tento stav nastane, když byla zranitelnost opravena nebo útočníci ztratili zájem.

[22] také uvádí, že stavy exploatace, odhalení a korekce nejsou fixní a že exploatace a korekce mohou nastat dříve, ve stejné době, nebo později než stav odhalení.

Neexistuje shoda mezi různými zdroji o názvech fází životního cyklu zranitelností. Jejich smysl však zůstává stejný. **[23]** představuje následující graf, který ukazuje jednotlivé fáze životního cyklu zranitelnosti a závislost rizika zranitelností na čase:



Obrázek 2 Životní cyklus zranitelností. Zdroj: [23]

Skenery zranitelností: "Skenery zranitelností jsou populární proaktivní technologie zabezpečení informace. Prohledávají systémy a sítě pro výskyt známých chyb a pak vytváří zprávu obsahující všechny výsledky vyhledávání, které může jednotlivec nebo podnik použít k posílení své bezpečnosti." **[24]**. Tyto nástroje používají uvedené výše databáze zranitelností jako NVD a CVE a jsou velmi důležité, protože "...odhalují zranitelnosti dříve, než má narušitel příležitost je zneužít." **[24]**. Některé populární skenery zranitelností budou popsány podrobně v průběhu této bakalářské práce.

4.1 Běžné zranitelnosti

Tato kapitola představí 10 nejčastějších kategorií zranitelností, které byly zvoleny na základě **[25]** - seznam top 10 kategorií zranitelností roku 2021 od společnosti **OWASP** (Open Web Application Security Project). OWASP je nezisková nadace, která vyvíjí software pro kyberbezpečnost.

Kategorie Zranitelností v seznamu OWASP Top 10 jsou zvoleny na základě statistických dat a průzkumů. K určení pozice kategorie zranitelností v seznamu se používá míra výskytu místo frekvence – což znamená procento testované populace, které je ovlivněno určenou zranitelností. K poskytování dat nutných pro vytvoření tohoto seznamu přispívali společnosti jako GitLab, Appsec Labs, HackerOne, PenTest-Tools aj.

Každá kategorie odpovídá určitým **CWE** (Common Weakness Enumeration), což je seznam typu softwarových a hardwarových typů slabostí. O CWE se mluví v předchozí kapitole – Zranitelnost.

Následuje seznam nejčastějších 10 kategorií zranitelností – od nejhorší do nejméně špatné (zpracováno podle **[25]**):

1. **Rozbité řízení přístupu**: Jde o špatně nastavená oprávnění. Řízení přístupu znamená, že každý uživatel má určitá oprávnění a jeho dostup k některým částem aplikace může být omezen. Zranitelnosti, které mohou dovést k rozbitému řízení přístupu jsou například špatná konfigurace CORS, která umožňuje dostup do API z nedůvěryhodných zdrojů, manipulace s metadaty jako cookies či JSON Web Token nebo obcházení prověrek kontroly dostupu ruční modifikací URL adresy či použitím nástroje modifikujícího API požadavky.
2. **Kryptografická selhání**: Nedostatečná či neexistující ochrana citlivých dat jako osobních údajů, hesel, čísel platebních karet atd. Může vzniknout kvůli přenosu dat v prostém textu místo použití šifrování, použití starých šifrovacích algoritmů, které jsou dnes už příliš slabé, použití zastaralých hash funkcí jako MD5 či SHA1, použití slabých šifrovacích klíčů atd.
3. **Injekce**: Aplikace je zranitelná vůči injekci, když se zadaná uživatelem data nejsou ověřena či filtrována a když se dotazy k databázi píšou ručně v kódu. Pak může útočník napsat vlastní dotaz či příkaz, například SQL dotaz, předat ho cílové aplikaci a vynutit jí ten dotaz spustit. Existují různé typy injekcí, nejčastější jsou SQL injekce, injekce příkazu operačního systému či injekce v ORM (Object Relational Mapping) systému.
4. **Nezabezpečený design**: Kategorie zranitelností, kterou lze vyjádřit jako chybějící či neefektivní design. Nezabezpečený design **není** nezabezpečená implementace, a tak není zdrojem všech ostatních kategorií zranitelností v tomto seznamu. Nezabezpečená implementace může existovat i při zabezpečeném designu. Některé příklady nezabezpečeného designu jsou například vygenerování chybových hlášení s citlivými daty nebo použití systému otázek a odpovědí při procesu zotavení

ztracených uživatelských údajů. Dalším příkladem je chybějící ochrana proti botům na webových stránkách.

5. **Chybná konfigurace zabezpečení**: Tuto kategorii zranitelnosti může mít aplikace, pokud například používá defaultní účty či hesla, má zapnuté zbytečné funkce jako porty, služby, účty či oprávnění, nemá nastavené nebo nejsou zapnuté nejnovější funkce zabezpečení, nebo software není aktualizovaný nebo je zranitelný. Konkrétním příkladem je zapnutý výpis adresářů na serveru, což umožňuje útočníku najít například Java třídy na adresářů serveru, stáhnout je, pak je dekompilovat a pomocí dekompilovaného zdrojového kódu najít chybu řízení přístupu.
6. **Zranitelné a zastaralé komponenty**: Pod pojmem komponenty se rozumí operační systém, webový server, DBMS, API, knihovny a další software. Je třeba znát verzí všech těchto komponent, pravidelně je aktualizovat a skenovat zranitelnosti v nich.
7. **Selhání identifikace a autentizace**: Jde o slabinách při zabezpečení přihlašovacích údajů uživatele a potvrzení identity uživatele. Slabiny této kategorii mohou existovat, pokud aplikace povolí útoky na hesla jako Brute Force či Dictionary útok (tím, že povolí mnohé pokusy o přihlašování se z jednoho zdroje), povolí slabá a často používaná hesla, ukládá hesla v prostém textu bez šifrování či slabě šifrovaná hesla, nemá systém vícefaktorového ověřování, nedeaktivuje korektně session ID při odhlášení se či po době nečinností aj.
8. **Selhání integrity software a dat**: V případech kdy se stahují soubory jako pluginy či knihovny z externích zdrojů, často pomocí nástrojů jako npm či Maven, existuje tato zranitelnost v aplikaci, pokud tyto soubory mohou být staženy z nedůvěryhodných zdrojů. Dalším případem jsou aplikace, které mají zapnutou funkci automatické aktualizace a stahují tyto aktualizace bez ověření integrity. Příkladem je útok na společnost SolarWinds, ve kterém útočníci vložili zlomyslný kód do určité aktualizace software SolarWinds Orion Platform, která byla pak distribuována mnohým nadnárodním společnostem a vládním agenturám, které ji používaly. Je to jeden z největších útoků, které využily slabinu tohoto typu.
9. **Selhání bezpečnostního protokolování a monitorování**: Protokolování a monitorování jsou určeny pro detekci porušení. Protokolování a monitorování se počítá za nedostatečné, pokud nejsou protokolovány události jako přihlášení, neúspěšná přihlášení a transakce vysoké hodnoty, pokud jsou protokoly uloženy lokálně, pokud protokoly nejsou monitorovány pro podezřelou aktivitu aj.
   Protokoly také nesmí být viditelné uživateli, neboť jinak je aplikace zranitelná vůči úniku informací.
10. **Server-Side Request Forgery (SSRF)**: Nastane, když webová aplikace načítá vzdálené prostředky bez ověření uživatelem zadané URL adresy. Útočník tak může aplikaci přimět k odeslání vytvořeného požadavku na neočekávaný cíl, a to i v případě, že je chráněna bránou firewall, sítí VPN nebo jiným typem seznamu řízení přístupu (ACL). Tato kategorie má relativně nízkou míru výskytu, ale jejich výskyt se zvyšuje, protože v moderních webových aplikacích se stává načítání URL adresy běžným scénářem.

4.2 Prevence běžných zranitelností

V seznamu kategorií zranitelností OWASP Top 10, který byl použit v předchozí kapitole, jsou u každé kategorií zranitelnosti také popsány způsoby její prevence. V této kapitole budou uvedeny tyto způsoby – u každé kategorii několik vybraných způsobů (ne všechny možné) (Zpracováno podle **[25]**)

1. **Rozbité řízení přístupu**: Je třeba dodržovat zásadu "Deny by default" ve všech případech kromě veřejných prostředků. Deny by default znamená, že uživatelům a jiným subjektům, kterým není výslovně povolen přístup, je přístup odepřen. Je třeba deaktivovat výpis adresářů webového serveru. Výpis adresářů je speciální webová stránka, která vypisuje všechny soubory, které se nachází na webovém serveru. Je třeba zrušit stavové session ID po odhlášení se a po dobu nečinnosti a minimalizovat dobu života JWT tokenů.
2. **Kryptografická selhání**: Data, která je zpracována, uložená či přenášená aplikací musí být klasifikována a je třeba určit jaká data jsou citlivá. Pak tyto citlivé daty je třeba zašifrovat, přitom používat silné algoritmy, protokoly a klíče a nepoužívat zastaralé kryptografické funkce jako MD5, SHA1 atd. Nesmí se zbytečně ukládat citlivá data a je třeba jej co nejdříve zlikvidovat. Dále je třeba vypnout caching (ukládání do mezipaměti) pro odpovědi, které obsahují citlivá data atd.
3. **Injekce**: Pro prevenci zranitelností typu injekce se musí používat tzv. parametrizované SQL dotazy, což znamená, že se SQL dotazy předem připraví, a vynechají se v nich některé hodnoty (parametry), které uživatel pak předá aplikaci. Jiná možnost je používat ORM (všechny moderní ORM systémy používají parametrizované dotazy). ORM (Object Relational Mapping) je systém, ve kterém se píšou SQL dotazy ve formě kódu použitého vývojářem programovacího jazyka místo SQL kódu.
4. **Nezabezpečený design**: Lze zabránit těmto zranitelnostem použitím knihoven či komponent se zabezpečenými designovými vzory, omezit spotřeby zdrojů uživatelem či službou, používat modelování hrozeb pro kritickou autentizaci, řízení přístupu atd. Je třeba také psát jednotkové testy a integrační testy, pomocí kterých lze ověřit, že kritické části aplikace jsou odolné vůči modelům hrozeb.
5. **Chybná konfigurace zabezpečení**: Je třeba odstranit všechny zbytečné funkce, komponenty, dokumentace atd. Vytvořit automatický proces, který ověří konfigurace a nástroje ve všech prostředích. Pak je třeba identicky nakonfigurovat prostředí vývojové, QA a produkční, ale s různými přihlašovacími údaji. Tento proces by měl být automatizován pro minimalizaci úsilí potřebného k nastavení nového zabezpečeného prostředí.
6. **Zranitelné a zastaralé komponenty**: Musí se odstranit všechny nepoužívané funkce, závislosti, soubory, dokumentace a komponenty. Neustále zaznamenávat verzí všech komponent na straně klienta a serveru a jejich závislosti. Nepřetržitě monitorovat zdroje jako Common Vulnerability and Exposures (CVE) a National

Vulnerability Database (NVD) pro zranitelnosti v komponentách. Všechny komponenty je třeba stahovat pouze z oficiálních a důvěryhodných zdrojů.

7. **Selhání identifikace a autentizace**: Implementovat vícefaktorovou autentizaci pro prevenci automatických útoků na hesla, nenasazovat produkty s výchozími přihlašovacími údaji, implementovat kontroly slabých hesel. Hesla musí odpovídat standardům NIST 800-63b sekci 5.1.1, které popisují podmínky, které musí být splněny pro silné heslo. Selhané pokusy přihlášení musí být omezeny, ale opatrně, aby nevznikl scénář Denial of Service (odmítnutí služby).
8. **Selhání integrity software a dat**: Lze zabránit použitím mechanismů jako digitálních podpisu pro ověření toho, že software pochází z očekávaného zdroje a nebyl změněn. Je třeba zajistit, že nástroje pro správu závislostí jako npm či Gradle využívají pouze důvěryhodné zdroje. Změny kódu a konfigurací musí procházet proces recenze pro minimalizací šancí toho, že zlomyslný kód či konfigurace bude zaveden do pipeline software.
9. **Selhání bezpečnostního protokolování a monitorování**: Všechny selhání u přihlášení, řízení přístupu, a ověření vstupu na straně serveru musí být protokolovány s dostatečným množstvím uživatelského kontextu, aby bylo možné identifikovat podezřelé či škodlivé účty, a protokoly mají být uloženy dostatečně dlouho, aby bylo možné provést opožděnou forenzní analýzu. Formát protokolů musí být kompatibilní s formátem vyžadovaným software pro správu protokolů a data protokolů musí být správně zakódovaná, aby se zabránilo injekcím či útokům na systémy protokolování a monitorování.
10. **Server-Side Request Forgery (SSRF)**: Prevenci tohoto typu zranitelnosti lze dosáhnout ze síťové vrstvy a z aplikační vrstvy. U síťové vrstvy je třeba vynutit Deny by Default zásady ve firewallu (viz bod 1.) nebo nastavit pravidla řízení přístupu k síti tak, aby veškerý internetový provoz mimo zásadního byl blokován. Je třeba také segmentovat funkci vzdáleného přístupu ke zdrojům v samostatných sítích.
    U aplikační vrstvy je potřeba ověřovat všechna data přijatá od klienta, neposílat klientovi nezpracované odpovědi, zakázat přesměrování http a vynutit schéma URL, porty a cíle pomocí seznamu povolených.

4.3 Analýza zranitelností

Tato kapitola se zabývá rešerší nad téma analýza zranitelností, představí pojem analýza zranitelnosti, její fáze, současné techniky a možný tvar analýzy zranitelnosti v budoucnosti.

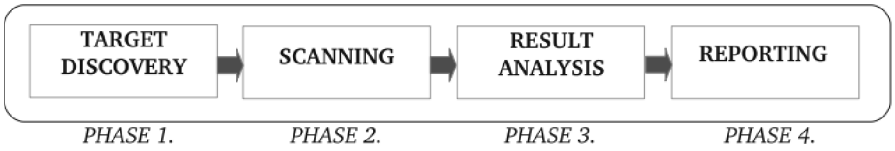
Pojmy „analýza zranitelností“ a „ohodnocení zranitelností“ se používají jako synonyma.

„Ohodnocení zranitelností je proces, který vymezuje, identifikuje a klasifikuje bezpečnostní mezery (zranitelnosti) na počítačích, sítích nebo komunikační infrastruktuře. Kromě toho může analýza zranitelností odhadnout účinnost navrhovaných preventivních opatření a vyhodnotit jejich skutečnou účinnost po jejich implementaci.“ **[26]**.

Analýza zranitelností je v dnešní době velmi důležitý aspekt kyberbezpečnosti, neboť bez schopnosti organizace či jednotlivce identifikovat zranitelnosti na jejich infrastrukturách nebudou schopni je opravit dříve, než je využije útočník. Analýza zranitelností je první část ve **VAPT** (Vulnerability Assessment and Penetration Testing, česky ohodnocení zranitelností a penetrační testování), což je „urážlivý způsob obrany kybernetického majetku organizace“ **[3]**.

Analýza zranitelností se skládá z čtyř fází (zpracováno podle **[3]**):

- **Target Discovery** (zjištění cíle): Jde o sbírání klíčové informaci o cíli, která pomáhá vygenerovat představu o bezpečnostní infrastruktuře cíle, což pomáhá při rozhodování o strategiích testů běhu programu. Sbírá se informace o sítích, podsystémech, převzatých technologiích, systémových prostředcích, aplikacích, komunikační infrastruktuře atd.
- **Scanning** (skenování): Po úspěšném zjištění cíle se provádí sken celého systému pro zranitelností, které jsou přítomné na cíli. Používají se různé nástroje a techniky se snahou najít libovolné bugy či skuliny, jako například otevřené porty, slabá hesla, nechtěné služby atd. Skenování, které se provádí jsou skenování portů, sítí, web aplikací aj. Skenování se ukončí seznamem potenciálních zranitelností na cíli.
- **Result Analysis** (analýza výsledků): Tester analyzuje seznam zranitelností z předchozího kroku. Cílem testera v této fázi je upřednostnění zjištěných zranitelností na základě jejich závažnosti a dopadu. Výsledky z kroku skenování trpí velkým množstvím „false positives“ (falešné poplachy) a ty se musí zjistit a odstranit. Výsledkem je finální optimalizovaný seznam upřednostněných zranitelností.
- **Reporting** (hlášení): V této fázi se tester zaměří na dokumentaci provedených operací a získaných výsledků v celém procesu hodnocení zranitelností. V této fázi tester vygeneruje zprávu o provedené práci tím, že zařadí seznam zranitelností spolu s jejich úrovněmi závažnosti a dalšími podrobnostmi. Tento seznam může organizace dále použít k žádosti o nápravu nalezených zranitelností.



Obrázek 3 Fáze analýzy zranitelností. Zdroj: [3]

V dnešní době existuje dva hlavní způsoby analýzy zranitelností:

- **Ruční**: Je to ruční provádění výše uvedených fází analýzy zranitelností, bez použití jakýchkoliv pomocných nástrojů speciálně vytvořených pro tento účel. „Tester podrobuje cíl různým testovacím případům a ručně sleduje odezvu a její variace. Pokud se zjistí, že variace není v souladu s pravidly chování, je software považován za zranitelný" **[3]**. Problém s ručním způsobem je doba provedení výše uvedených testů.
- **Automatický**: Dnes velmi běžný způsob analýzy zranitelností. Provádí se s využitím speciálních nástrojů, které pomáhají zranitelnosti identifikovat a vygenerovat výpis výsledků. Je to velmi pohodlný způsob a trvá mnohem rychleji než ruční způsob, výsledky ale nemusí být vždy perfektní.

S rozvojem umělé inteligenci a spojených s ní pojmů vstoupí tento obor i do domény kyberbezpečnosti. V poslední době lze najít literaturu, ve které jsou navrženy způsoby analýzy zranitelností s využitím umělé inteligence, hlubokého učení, strojového učení či neuronových sítí. Příkladem je **[27]** - AutoVAS, ve kterém je představen automatizovaný systém pro analýzu zranitelností založený na hlubokém učení. „AutoVAS se skládá z fáze učení, která trénuje model pomocí datové sady a fáze detekce, která používá trénovaný model ke zjištění, zda je vstupní zdrojový kód zranitelný.“ (převzato z **[27]**).

[28] představuje přehled literatury o rešerši na téma umělé inteligenci pro analýzu zranitelností.

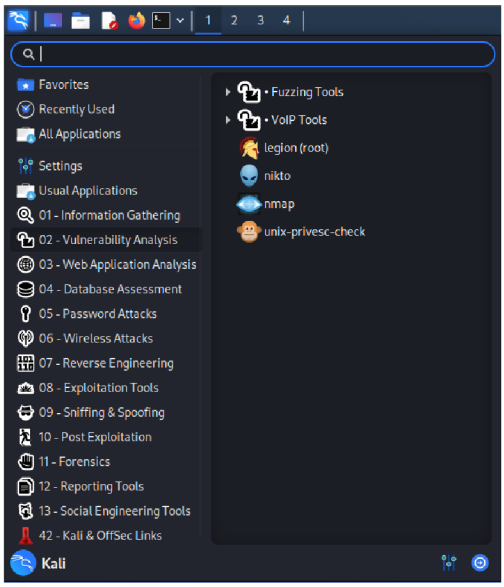
5 Nástroje Kali Linux

Aktuální verze Kali Linux **2022.3** nabízí více než 600 nástrojů pro účely penetračního testování. Tento počet stále roste i s tím, že vývojáři Kali se snaží odstranit redundantní nástroje. Nástroje jsou rozděleny na 13 kategorií:

- **Sbírání informaci** – Průzkum cíle a sbírání jakékoli informaci o cíli pomocí například skenu sítě a portů, analýzy internetových protokolů jako SMTP a SNMP, analýzy DNS atd.
- **Analýza zranitelností** – Tato bakalářská práce se zabývá nástroji této kategorii. Jde o zkoumání cílového systému se snahou najít předem známé typy zranitelností a určit, zda je možné na cílovém systému tyto zranitelnosti využit.
- **Analýza webových aplikací** – Podobně jako analýza zranitelnosti, ale pouze pro webové aplikace jako servery. Je tam například nástroj wpscan, který analyzuje webové aplikace vytvořené s pomocí WordPress.
- **Posoužení databází** – Hledání zranitelností v databázích jako SQL injekce, slabá hesla atd.
- **Útoky na hesla** – Zahrnuje nástroje, které se snaží odhadnout hesla s pomocí existujícího slovníku slov nebo s pomocí náhodných kombinací znaků.
- **Bezdrátové útoky** – Útoky na bezdrátové systémy jako Wifi a Bluetooth.
- **Reverzní inženýrství** – Reverzní inženýrství je proces, ve kterém se pokouší transformovat zkompilovaný software zpět do jeho zdrojového kódu. Tato kategorie zahrnuje jak nástroje, které se snaží tuto transformaci udělat automaticky, tak i hex-editory, které jsou určeny pro ruční manipulaci kódu kompilované aplikace v hexadecimálním formátu.
- **Nástroje pro exploataci** – Dalším krokem po sbírání informací a analýze zranitelností je jejich exploatace – využití zranitelností a získání výhody z toho. Nástroje v této kategorii umožňují právě toto.
- **Sniffing & Spoofing** – Sniffing (šňupání) je proces monitorování a zaznamenávání datových paketů, které tečou po sítí. Spoofing (falšování) znamená, že komunikace útočníku by vypadala tak, jako by pocházela z důvěryhodného zdroje. Zdrojem může být například IP adresa, DNS server, email atd.
- **Post Exploatace** – Po úspěšné exploataci zranitelnosti lze provést akce pro další exploataci systému. Například po získání dostupu do vzdáleného počítače, lze s pomocí post exploataci tento dostup udržovat.
- **Forenzní nástroje** – Nástroje pro výpis a zotavení dat v cílovém systému. Některé nástroje se používají pro soubory na souborovém systému, jiné pro výpis dat z internetového provozu.
- **Reportovací nástroje** – Tyto nástroje pomáhají vygenerovat zprávu, ve které jsou popsané výsledky práce jiných nástrojů.
- **Nástroje sociálního inženýrství** – Sociální inženýrství znamená exploatace lidského chování s cílem získání důvěrné informace. Příkladem je Phishing, ve kterém se

rozesílají emailové zprávy, které vypadají tak, jako kdyby pocházely z oficiálního zdroje, například z určité sociální sítě. Tyto emaily zpravidla obsahují zlomyslný odkaz nebo přihlašovací okno, které jsou využity pro krádež citlivých údajů oběti.

Tato bakalářská práce se zabývá analýzou zranitelností. Většina potřebných nástrojů pro tento účel se nachází v kategoriích **analýza zranitelností** a **analýza webových aplikací**.



Obrázek 4 Kategorie nástrojů Kali a nástroje analýzy zranitelností. Zdroj: Autor

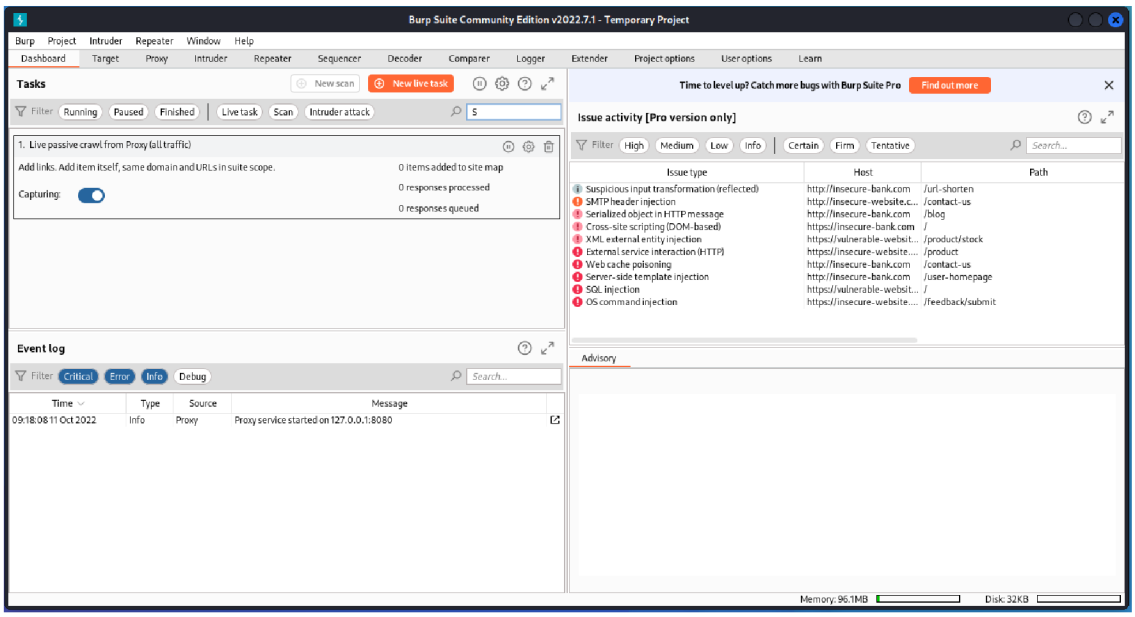
5.1 Burp Suite

Burp Suite je jeden z nejrozsáhlejších nástrojů pro testování bezpečnosti web aplikací a dnes už umí najít zranitelnosti i v mobilních aplikacích a API. Byl vyvinut v roce 2004 Dafyddem Stuttardem. Burp Suite je vlastně kolekce nástrojů a obsahuje nástroje nejen pro analýzu zranitelností, ale i pro jiné účely, jako například útoky na hesla.

Burp Suite je dostupný ve 3 verzích – Community, Professional a Enterprise. Community verze je dostupná zadarmo a je nainstalována v Kali Linux. Professional a Enterprise jsou placené verze a obsahují některé funkce, které nejsou v Community edition, jako například automatické skenování zranitelností.

V Kali Linux lze najít Burp Suite v kategorii Web Application Analysis pod názvem burpsuite. Oficiální stránka Burp Suite je **[29]**.

Burp Suite má také vlastní rozhraní – nemusí se používat přes terminál.



Obrázek 5 Rozhraní Burp Suite. Zdroj: Autor

Hlavní možnosti Burp Suite:

Burp Proxy: Je to důležitá část práce s Burp Suite. Burp Proxy funguje jako proxy server mezi prohlížečem a cílovým serverem a umožňuje zachytit a manipulovat přicházející nebo odcházející provoz. Burp Proxy lze použit buď se zahrnutým prohlížečem uvnitř Burp Suite, nebo lze ho nastavit na vlastní prohlížeč jako Firefox nebo Chrome.

Repeater: Repeater umožňuje opakovaně posílat http požadavky s určitými modifikacemi a analyzovat přicházející odpovědi.

Intruder: Intruder je nástroj umožňující automatizaci útoků proti webovým stránkám na základě uživatelem daného payloadu. Umožňuje například sbírat užitečnou informaci, spustit Brute Force útoky na hesla a nalézt různé typy zranitelností.

Target: Umožňuje vytvořit mapu webové stránky. Uživatel Burp Suite musí otevřít Burp prohlížeč, otevřít cílovou webovou stránku a ručně otevírat každý existující odkaz atd. Tento proces vyplní mapu cílové stránky. Existuje i automatické vytvoření takovéto mapy pomocí funkce Crawler v Burp Scanner.

Scanner: Scanner je velmi výkonný nástroj, který automaticky analyzuje webovou stránku pro zranitelnosti. Obsahuje také funkci Crawler, která umožňuje automatické vytvoření mapy cílové stránky. Scanner bohužel není dostupný v Community verzi Burp Suite.

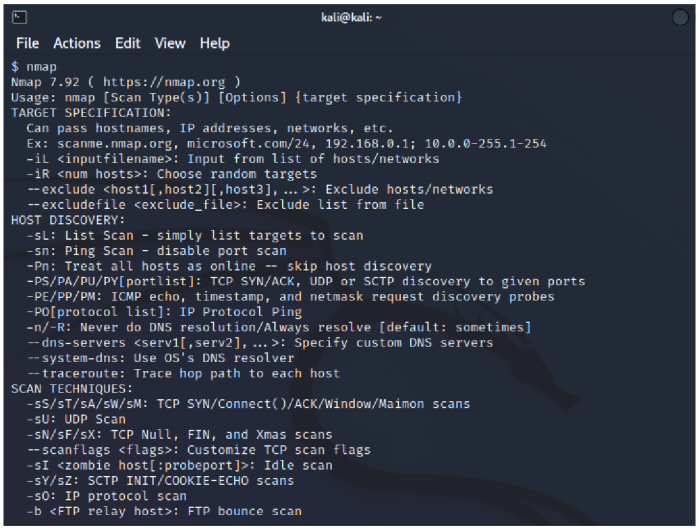
Burp Suite obsahuje mnoho dalších možností. Informaci o nich lze najít na jeho oficiální dokumentaci: **[30]**.

Možnosti využití: Burp Suite je velmi obsáhlý nástroj a má spoustu funkcí. Některé příklady jeho možností jsou: Sbírání informaci o cílové webové stránce, útoky na hesla s cílem získání dostupu k uživatelským účtům, modifikace a posílání modifikovaného http provozu, což umožňuje různé typy exploatací zranitelností, nalezení skrytých funkcí u cíle – zjištění neviditelného obsahu atd.

5.2 Nmap

Nmap (Network Mapper) je open source nástroj a je dostupný zadarmo na Linux, Windows a Mac OS. Je ve vývoji více než 20 let, je jednoduchý na použití a flexibilní, nabízí výkonnou funkčnost a má obsáhlou dokumentaci na jeho oficiální stránce.

V Kali Linux se najde v kategoriích Information Gathering a Vulnerability Analysis a nazývá se nmap. Oficiální stránka: **[31]**.



Obrázek 6 Terminál při spuštění Nmap. Zdroj: Autor

Hlavní možnosti: Nmap umožňuje vytvořit mapu určité sítě pomocí skenování portů a IP adres zařízení v této síti. Pomocí Nmap lze zjistit, jaká zařízení běží na síti, jaké služby nabízí tato zařízení, jaké mají operační systémy a jejich verze, jaké mají filtry packetů a firewally aj.

Na oficiální stránce Nmap lze najít odkazy ke stažení Nmap a jeho pluginů, dokumentaci, knihy o Nmap a další informace.

Nmap také obsahuje uživatelské rozhraní, ve formě samostatného software, který se jmenuje **Zenmap**. Zenmap ale není nainstalován v Kali Linux, a proto se musí stáhnout z oficiální stránky Nmap.

Kromě Zenmap, existují ještě 3 další pluginy pro Nmap:

- **Ncat** – Nástroj pro přenos dat, přesměrování a ladění.
- **Ndiff** – Pro srovnání výsledků skenů.
- **Nping** – Pro generaci packetů a analýzy odpovědí.

Ncat a Nping se musí stáhnout. Ndiff je již integrován v Nmap.

Základní parametry: Při spuštění Nmap pouze s příkazem `nmap` bez dalších parametrů se vypíše seznam parametrů a jejich krátký popis, jak je uvedeno na obrázku výše. Některé důležité parametry jsou uvedené níže:

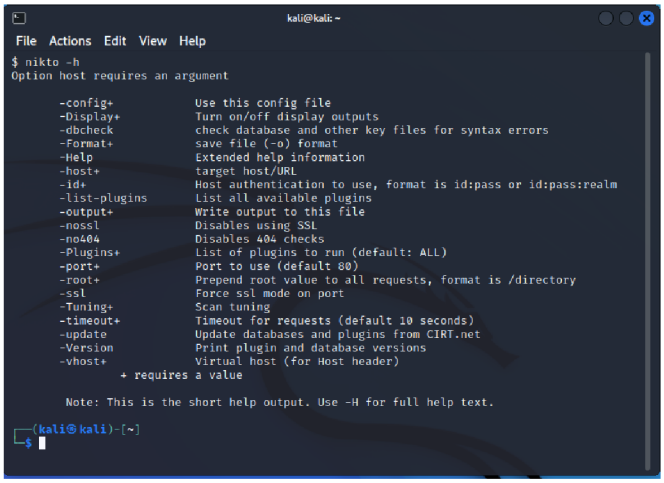
- `-sV`: Zkontroluje otevřené porty se snahou určit služby a jejich verzi.
- `-O`: Zapnutí detekci operačního systému, což určuje použitý OS a některé základní informace o něm.
- `-sC`: Zapnutí některých běžných skriptů. Lze také načíst vlastní skript pomocí `--script`.
- `--traceroute`: Podobný příkazu `traceroute` na Linux či `tracert` na Windows – sleduje cestu od skenovaného stroje k cíli.
- `-A`: Agresivní režim. Je to kombinace všech čtyř výše uvedených parametrů. Nevýhoda agresivního režimu je že zkoumání s agresivním režimem je snadno detekovatelné cílovým systémem.
- `-sS`: "Stealth" (tichý) režim, naopak od agresivního režimu je tichý režim těžko detekovatelný.

Možnosti využití: Nmap je určen hlavně pro sbírání informaci o cílovém systému a pomáhá najít díry v zabezpečení systému.

5.3 Nikto

Nikto byl napsán v jazyce Perl v roce 2001 Chrisem Sullo. Nikto je Open source a bezplatný skener zranitelností na web serverech. Nikto není tichý nástroj a je snadno detekovatelný jeho cílem. Nemá uživatelské rozhraní, ale je velmi jednoduchý na použití.

V Kali se nachází v kategorii Vulnerability Analysis pod názvem nikto.



Obrázek 7 Terminál při spuštění Nikto. Zdroj: Autor

Hlavní možnosti:

Na skenovaném web serveru Nikto je schopen najít zastaralé verze serverů, nebezpečné soubory a programy, zkontrolovat konfiguraci serveru, snaží se určit nainstalované servery a další software a může najít běžné zranitelnosti jako SQL injekci či XSS.

Nikto má další funkce jako podpora SSL, skenování více portů najednou, ukládání výsledků skenů do textových souborů, integrace s Metasploit, což je další software pro penetrační testování atd.

Základní Parametry: Nejdůležitější parametry Nikto jsou `-h <host>` a `-p <port>`, které odpovídají cílovému hostu (IP adrese) a portu respektive. Host se vždycky musí uvádět. Když se neuvede port, bude použit defaultní port 80. Některé další důležité parametry jsou `-ssl`, což umožňuje použití HTTPS, `-Version` pro výpis verzí pluginů a databázi a `-update`, což aktualizuje pluginy a databáze.

Možnosti použití: Nikto pomáhá najít na web serveru problémy a zranitelnosti, které se týkají špatných souborů a problémů s konfigurací, aby bylo možné vědět, co je třeba opravit a tím odstranit existující zranitelnosti.

Dokumentaci a více informaci o Nikto lze najít na oficiální stránce https://cirt.net/Nikto2

5.4 SQLMap

SQLMap je bezplatný Open Source nástroj napsaný v Python, který je vytvořen pro odhalení a exploataci zranitelností **SQL injekce**, o které se psalo v části o běžných zranitelnostech. SQLMAP nemá uživatelské rozhraní – lze ho použit jen přes terminál.

SQLMAP se v Kali Linux najde v kategorii Web Application Analysis a jmenuje se sqlmap. Oficiální stránka SQLMAP je **[32]**.

Obrázek 8 SQLMAP. Zdroj: Autor

Hlavní možnosti:

SQLMap dokáže SQL injekci najít jednoduše a automaticky, a může tuto zranitelnost nejen odhalit, ale i využit, tedy zadat deformovaný SQL příkaz do cílové webové stránky.

Podporuje velký počet databází (MySQL, Oracle, PostgreSQL…), šest různých typů injekce, může se připojit k databázi bez loginu a hesla, pokud je v této DB možnost provést SQL injekci. Umožňuje vyjmenovat databází, tabulky, sloupce a další data, stáhnout si soubory z DB serveru a má spoustu dalších funkcí.

Základní Parametry:

- `-u <URL>`: Cílový URL
- `-p <param>`: Nastaví GET parametr, proti kterému bude použitá SQL injekce.
- `--risk <risk>` a `--level <level>`: Risk a level zvyšují počet testů a počet metod pro nalezení SQL injekci. Tyto 2 parametry ale hodně zvyšují čas potřebný pro nalezení SQL injekci a také riziko toho, že pokusy nalezení SQL injekci budou detekovány cílem. Level má rozsah 1-5 a risk 1-3, přičemž jsou defaultně oba na 1. Pokud se

SQLMAP nepodaří najít zranitelnost SQL injekce, je třeba postupně zvyšovat úrovně risk a level.

- `--batch`: V průběhu práce může SQLMAP zeptat uživatele různé ano/ne otázky o metodách nalezení SQL injekci. S tímto parametrem bude vždycky automaticky zvolená defaultní odpověď ke každé otázce.
- `--method=<method>`: Nastaví http metodu, která bude použita při http požadavcích.
- `--dbms=<dbms>`: Nastaví jméno použité cílem dbms, například postgresql či sqlite. Toto umožňuje hodně snížit dobu zpracování.
- Vyjmenování: Existuje hodně různých parametrů pro vyjmenování, v závislosti na to, co je třeba vypsat. Například `--tables` vypíše tabulky database.

Možnosti použití: Zranitelnost typu SQL injekce umožňuje manipulaci s databází cílového serveru – umožňuje získat dostup k databázi nebo k některým její částem.

5.5 WPScan

WPScan umožňuje analyzovat web aplikaci, která je vytvořená s pomocí **WordPress** a snaží se v této aplikaci najít různé zranitelnosti.

V Kali Linux se najde v kategorii Web Application Analysis pod názvem wpscan. Oficiální stránka WPScan je **[33]**.

5.5.1 WordPress

WordPress je platforma, která umožňuje uživatelům si velmi jednoduše vytvořit webovou stránku bez znalostí programování a nabízí bezplatný hosting webových stránek pod doménou wordpress.com. WordPress je velmi populární a značná část webových stránek na internetu je vytvořena pomocí této platformy.

Bohužel webové stránky na WordPress jsou často cílem různých kybernetických útoků. WordPress je bezpečná platforma, a když se vyskytuje určitá zranitelnost, je rychle opravená vývojáři WordPress. Problémem je ale to, že se WordPress musí pravidelně aktualizovat, a velká část jeho uživatelů aktualizace neprovádí, a tak jsou oběti kybernetických útoků. Kromě toho, uživatelé WordPress si často zvolí slabé heslo, jako 12345, a pak jsou snadným cílem Dictionary nebo Brute Force útoků na hesla.

Většina zranitelností WordPress se ale nachází v jeho pluginech. Tyto Pluginy jsou doplňky k WordPress, které umožňují uživatelům přidat novou funkčnost nebo zjednodušit proces vytvoření webové stránky. Existují zlomyslné pluginy, které jsou většinou dostupné pouze na neoficiálních webových stránkách a mohou obsahovat malware. Důvěryhodné pluginy také mohou mít zranitelnosti. Ty se pak opraví při aktualizaci pluginu, ale stejně jako v případě samotného WordPress, uživatelé těchto pluginů je zřídka aktualizují.

Obrázek 9 WPScan. Zdroj: Autor

Hlavní Možnosti:

WPScan je vytvořen speciálně pro analýzu výše uvedených zranitelností. Na oficiální webové stránce WPScan lze najít dlouhý seznam známých zranitelností od roků 2005: https://wpscan.com/wordpresses

WPScan umí určit nainstalovanou verzi WordPress, instalované pluginy a zranitelnosti v nich. Může vyjmenovat uživatelské loginy a media soubory, zahájit Brute Force útok atd.

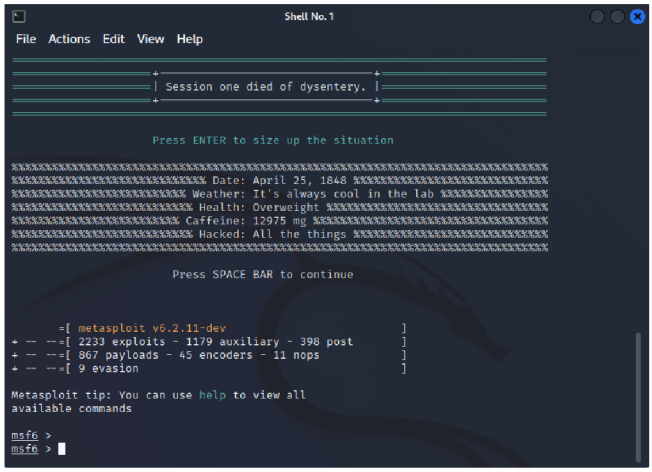
Základní Parametry: Ve WPScan jsou 2 povinné parametry: `--url <url>` a `--api-token <token>`. `--url` je uvedení cílové adresy, kterou bude WPScan zkoumat. Do `--api-token` je třeba napsat uživatelský API token, který lze získat ze stránky WPScan. O API tokenu se bude mluvit v praktické části, v úloze o WPScan. Dalšími důležitými parametry jsou parametry pro vyjmenování. Pro vyjmenování je třeba použit parametr `-e <option>`. Seznam optionů lze najít na dokumentaci WPScan. Příklad: `-e vp` vyjmenuje pouze **zranitelné pluginy**.

Možnosti Použití: WPScan může být použit autory WordPress aplikací, aby mohli vědět, zda jejich aplikace obsahuje zranitelnosti či špatné pluginy.

5.6 Metasploit Framework

Metasploit Framework je rozsáhlá sada nástrojů pro penetrační testování. Metasploit není pouze jedním nástrojem a je kompletní framework nabízející širokou škálu funkcí. Tento framework je napsaný v programovacím jazyce Ruby a je open source.

V Kali Linux lze najít Metasploit Framework v kategorii Exploitation Tools pod názvem metasploit framework. Oficiální stránka: **[34]**.



Obrázek 10 Metasploit Framework – Terminál po vstupu do msfconsole. Zdroj: Autor

Hlavní možnosti:

Metasploit má 2 verze: Metasploit Framework, která je bezplatná a je defaultně instalována v Kali Linux, a Metasploit Pro, která je placená a obsahuje pokročilé funkce.

Komponenty Metasploit Framework (zpracováno podle [35]): Metasploit se dělí na 7 komponent:

- **Exploity** – Hlavní část Metasploitu – exploity využívají zranitelnosti systému a útočí na něj. Aby bylo možné rozhodnout, který exploit je třeba použit, musíme provést rozsáhlý výčet a posouzení zranitelností cílového systému.
- **Payloady** – Payloady představují akce, které budou provedeny po exploataci cílového systému pomocí exploitu. Payloady se dále dělí na 3 kategorie:

  Singles: Zcela samostatné jednotky exploitu, které mají všechno potřebné pro exploataci zranitelnosti na cílovém systému. Mohou ale být moc velké, což způsobí problém v případech omezení velikosti.

  Stagery: Používají se v případech, když hodně záleží na velikosti payloadu. Tyto payloady nemají v sobě shell-kód potřebný pro exploataci a jednoduše nastaví připojení mezi útočícím systémem a cílem.

  Stagy (Stages): Používají se, když se otevírá připojení mezi útočícím systémem a cílem při použití Stagerů, tyto payloady se stahují na cílový systém a obsahují potřebný shell-kód pro exploataci zranitelnosti.

- **Encodery** – Úlohou Encoderů je zamlžit exploity a payloady aby nebylo možné je detekovat bezpečnostními software na cílovém systému.
- **NOPy** – Název je převzat z programovacího jazyka Assembly a znamená **No OPeration**. NOPy vlastně nedělají nic, ale mohou pomoci při psaní exploitů a shell-kódů.
- **Post** – Post (Post Exploatace) je sada modulů, která pomáhá dále proniknout do systému po úspěšné exploataci. S post moduly je možné například: krást cookies a uložená hesla, spustit PowerShell skripty, eskalovat uživatelská oprávnění aj. Post moduly se dělí na kategorie podle cílového operačního systému.
- **Evasion** – Nová kategorie modulů, dostupná od verze Metasploit 5. Tyto moduly modifikují Payloady tak, aby je nebylo možné detekovat bezpečnostním software.
- **Pomocné moduly** – Moduly, které nejsou exploity se počítají jako pomocné moduly. Tyto moduly jsou vytvořeny pro splnění konkrétních a dost jednoduchých účelů, jako například zkoumání sady IP adres pro zjištění běžících FTP serverů na těchto adresách. Na webové stránce https://www.offensive-security.com/metasploit-unleashed/auxiliary-module-reference/ lze najít informaci o všech pomocných modulech v Metasploit.

Kromě těchto komponent, Metasploit také obsahuje tyto 4 nástroje:

- **Msfconsole** – msfconsole je rozhraní příkazového řádku vytvořené pro Metasploit. Aby bylo možné použit příkazy Metasploit, je třeba nejdříve vstoupit do msfconsole v příkazovém řádku. I když existují grafická uživatelská rozhraní pro Metasploit, msfconsole je nejlepší a nejvíce podporované rozhraní.
- **Msfdb** – msfdb je databáze Metasploitu, založená na databázovém systému PostgreSQL.
- **Msfvenom** – msfvenom je utilita pro generování a šifrování payloadů. Je to náhrada utilit msfpayload a msfencode, které existovali v dřívějších verzích Metasploit.
- **Meterpreter** – Meterpreter je speciální payload, který používá DLL injekci v paměti, což znamená, že se nic nepíše do disku a nevytvářejí se nové procesy. Důsledkem toho je to, že tento payload je prakticky neviditelný. Meterpreter také poskytuje útočníkovi interaktivní shell s pomocí kterého může útočník spustit jakýkoli kód na cílovém systému.

Metasploitable: Na oficiální stránce vývojářů Metasploit Framework (Rapid7) lze si také stáhnout zadarma operační systém Metasploitable 2 ve formě virtuálního stroje. Tento operační systém je založený na Ubuntu Linux a podobně jako OWASP Juice Shop, Metasploitable schválně obsahuje zranitelnosti pro účely cvičení funkcí Metasploit Framework. Tento operační systém bude použit v posledních dvou úlohách praktické části.

Možnosti Použití: Metasploit Framework je obrovská kolekce nástrojů a je schopen vyplnit většinou funkcí ostatních nástrojů uvedených výše.

6 Praktická Část

V této části bude prezentováno praktické využití nástrojů analýzy zranitelností dostupných v Kali Linux. Nejdříve se ukáže nastavení speciálního web serveru, který schválně obsahuje zranitelnosti a je vytvořen speciálně pro cvičení výše uvedených nástrojů. Poté se praktická část bude pokračovat formou ukázky deseti řešených úloh.

Každá úloha bude mít určitý cíl a zvolený nástroj, který je vhodný pro dosažení uvedeného cíle. Při každé úloze se budou ukazovat příkazy (pokud se nástroj používá přes terminál) nebo tlačítka a další GUI elementy (pokud má nástroj uživatelské rozhraní), které jsou potřebné pro dosažení cíle, a výsledky těchto funkcí.

Nástroje využité v této části jsou pouze ty, které byly představeny v teoretické části.

6.1 Nastavení web aplikace OWASP Juice Shop

V této části se ukáže návod na nastavení OWASP Juice Shop. OWASP (The Open Web Application Security Project) je organizace, která se zabývá zlepšením softwarové bezpečnosti. OWASP Juice Shop je, webová stránka pro účely procvičení penetračního testování.

OWASP Juice Shop lze najít na oficiální stránce OWASP: https://owasp.org/www-project-juice-shop/ a návod na nastavení na jejich Github: https://github.com/juice-shop/juice-shop

Ze stránky na Github je vidět, že existují různé způsoby nastavení OWASPJS: Formou Docker kontejneru, nasazení na platformu Heroku, ze zdrojového kódu atd. V této práci se použije "Packaged distributions" (balené distribuce).

Návod:

Nejdříve se musí nainstalovat Node.js:

```
sudo apt install nodejs
```

Pak na stránce https://github.com/juice-shop/juice-shop/releases se musí stáhnout verze Juice Shop odpovídající operačnímu systému a verzi Node.

Balíček s Juice Shop se má rozbalit kdekoli na počítači.

V rozbalené složce juice shop je třeba otevřít terminál a napsat příkaz:

```
npm start
```

Webovou stránku pak lze najít na lokální adrese http://127.0.0.1:3000

Pro obrázek výsledné webové stránky viz **příloha 1**.

6.2 Nastavení Burp Proxy na vlastní prohlížeč

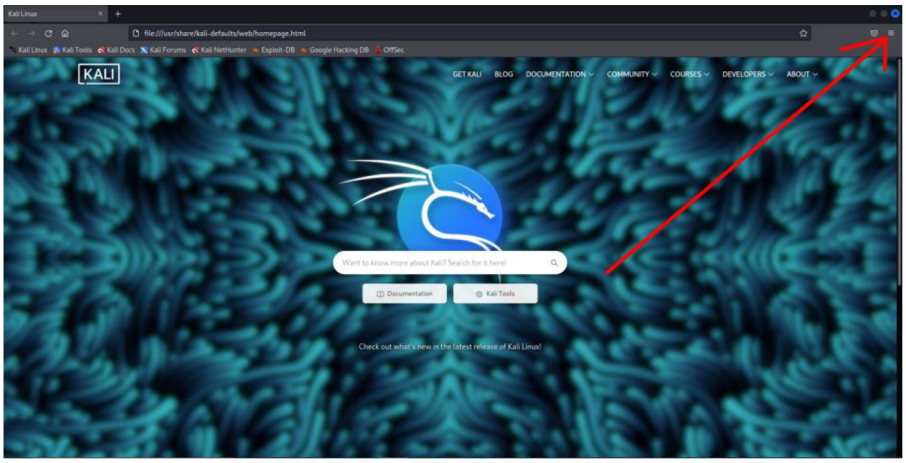
Burp Proxy je nejdůležitější část Burp Suite a umožňuje zachytit a modifikovat síťový provoz mezi prohlížečem a cílovou aplikací. Jednoduše řečeno, Burp Proxy funguje následně (převzato z **[12]**):

1. Uživatel si otevře prohlížeč.
2. Uživatel si zadá URL webové stránky, kterou chce navštívit.
3. Prohlížeč přesměruje požadavek do Burp Suite, který pak předá požadavek cílové webové stránce.
4. Cílová webová stránka odpovídá požadavku a posílá odpověď zpět do Burp Suite, který pak předá odpověď prohlížeči, aby ji bylo možné vykreslit.

Burp Suite má vlastní prohlížeč a funguje s Burp Proxy bez potřeby dalších konfigurací. Zde se ale představí návod, jak nakonfigurovat Burp Proxy s vlastním prohlížečem. V tomto návodu se použije Firefox, ale lze Burp Proxy nakonfigurovat i s jinými prohlížeči.

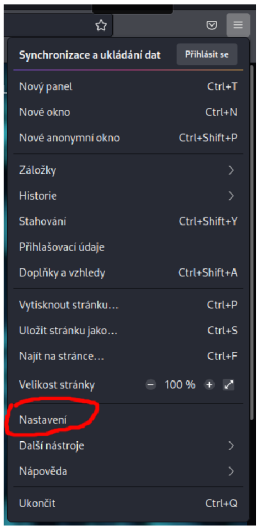
Návod na Konfiguraci Burp Proxy s Firefox:

1. Ve Firefox se musí otevřít hlavní menu aplikace (tzv. “Hamburger Menu” se třemi čárami):



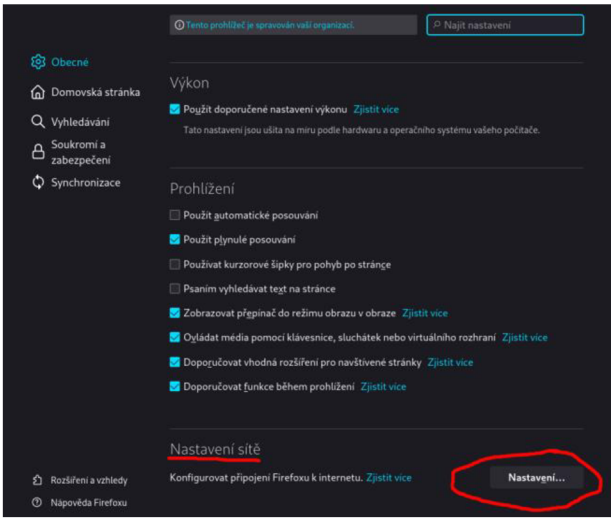
Obrázek 11 Burp Proxy na prohlížeči krok 1. Zdroj: Autor

2. Pak se zvolí Nastavení:



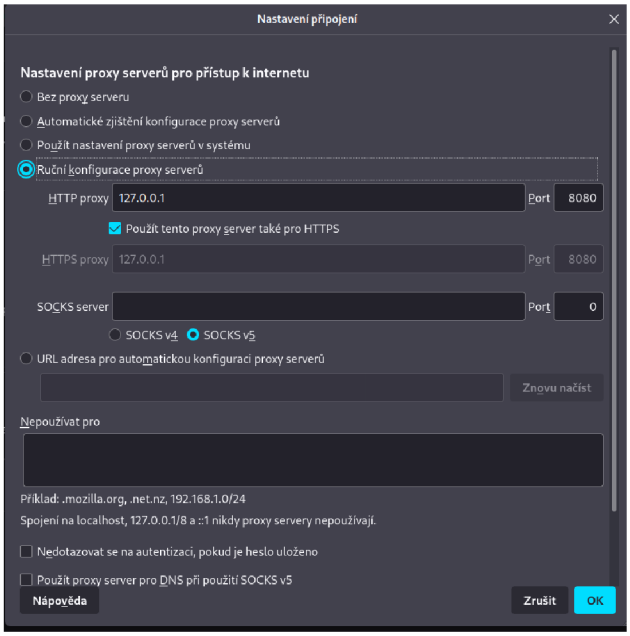
Obrázek 12 Burp Proxy na prohlížeči krok 2. Zdroj: Autor

3. V levé části stránky nastavení se zvolí Obecně, a pak je třeba zmačknout tlačítko Nastavení dole ve složce Nastavení sítě:



Obrázek 13 Burp Proxy na prohlížeči krok 3. Zdroj: Autor

4. V nově otevřeném okénku se zvolí Ruční konfigurace proxy serverů a do horního textového políčka „http Proxy“ se má napsat 127.0.0.1 a hned vpravo v textovém políčku Port se napíše 8080. Pod těmito políčky se musí zvolit zaškrtávací políčko „Použit tento proxy server také pro HTTPS“:



Obrázek 14 Burp Proxy na prohlížeči krok 4. Zdroj: Autor

Po nastavení proxy se už prohlížeč nebude správně fungovat! Proxy se musí nastavit pouze před prací s Burp Suite. Po prací s Burp Suite se musí v okénku z 4. kroku zvolit nahoře možnost „Bez proxy serveru“. Jinak nebude možné pracovat s prohlížečem.

6.3 Úloha 1 – Analýza zranitelností s Nikto

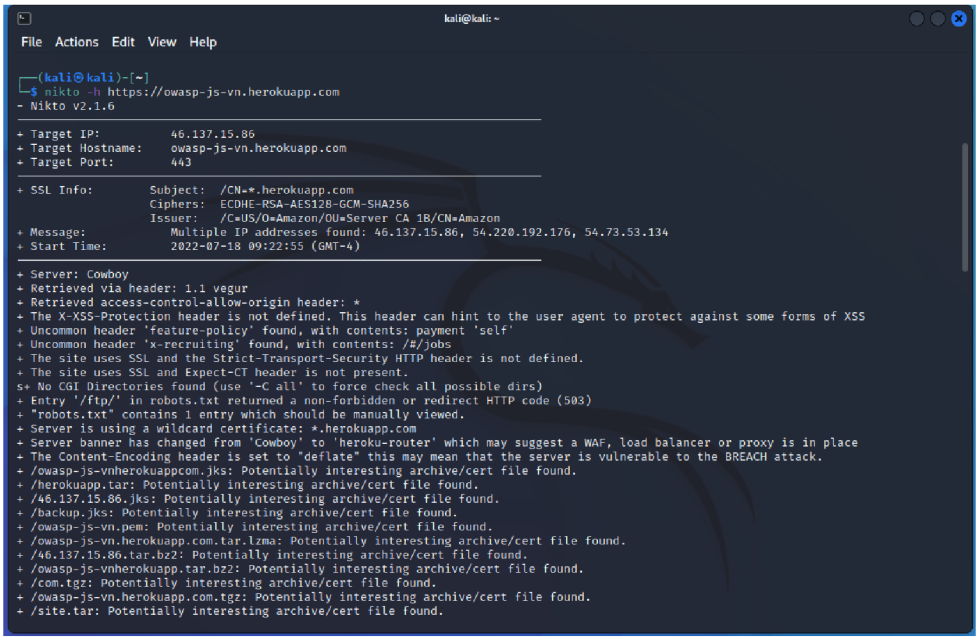
Cílem této úlohy je prozkoumat dříve vytvořenou kopii OWASP Juice Shop s pomocí Nikto a analyzovat výstup s cílem nalezení potenciálních zranitelností, které existují na OWASP Juice Shop.

Pro skenování OWASPJS s pomocí Nikto lze využit příkaz:

```
nikto -h <adresa>
```

V tomto případě byla použitá kopie OWASPJS nasazená na Heroku místo lokálního serveru, což dává lepší výsledky při skenu s Nikto. Heroku je platforma jako služba, která umožňuje vývojářům nasazovat a spravovat jejich aplikace na cloudu. Oficiální stránka Heroku: **[36]**

Výsledek:



Obrázek 15 Výsledek skenu Nikto. Zdroj: Autor

Výstup je velmi dlouhý – tady je uvedena jenom nejdůležitější část výsledku.

Analýza výstupu: První řádky dávají informaci o IP adrese stránky, její názvu, portu, informace o SSL a čas začátku skenu.

Další řádky dávají více různé informace, jako například, který server (serverový software) je použit (Cowboy).

Zajímavější řádek je "**The X-XSS-Protection header is not defined…**" ten říká, že není definována hlavička ochrany proti XSS, a jak je zmíněno ve stejném řádku by ta hlavička mohla zabránit některým formám XSS útoků.

Další zajímavý řádek je "**The site uses SSL and the Strict-Transport-Security HTTP header is not defined.**" Podle **[37]** http Strict Transport Security (HSTS) je zlepšení bezpečnosti, kde prohlížeč automaticky přesměruje všechny http požadavky na https. Pokud tato ochrana neexistuje a uživatel zadá URL adresu stránky bez prefixu http/https nebo s prefixem http místo https na začátku, může být terčem některých útoků.

Řádek "**Entry '/ftp/' in robots.txt returned a non-forbidden or redirect http code (503)**" říká, že na webové stránce existuje tajná adresa, ke které by uživatel neměl mít dostup, ale dostup má. Pokud si uživatel zadá adresu http://127.0.0.1:3000/ftp, dostane dostup ke složce obsahující různé soubory, které by neměly být vidět běžnému uživateli.

Poslední důležitý řádek je "**The Content-Encoding header is set to "deflate" this may mean that the server is vulnerable to the BREACH attack**" to znamená, že webová stránka je zranitelná vůči BREACH útoku kvůli další nedefinované hlavičce. U BREACH útoku, podle **[38]**, útočník může v některých specifických případech získat dostup k přihlašovacím údajům určitého uživatele.

Vyhodnocení úkolu: S pomocí základního skenu nástroje Nikto byl vyplněn úkol, jehož cílem bylo nalezení potenciálních zranitelností na OWASPJS. Ve výstupu výsledků skenu byly nalezeny některé řádky, které pravděpodobně naznačují potenciální zranitelnosti, nejdůležitější z nich jsou uvedeny výše.

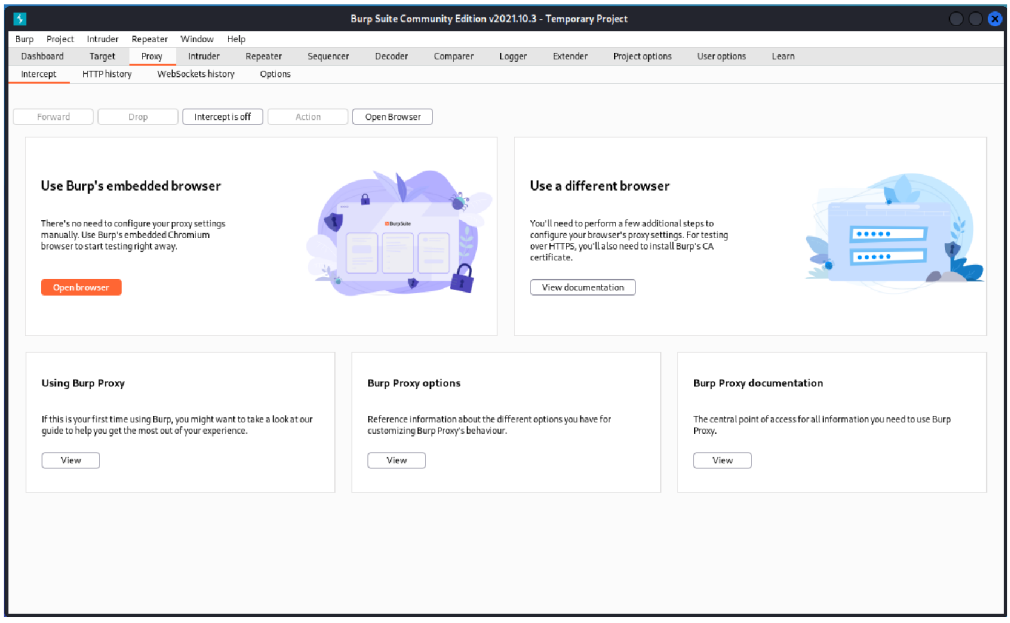
6.4 Úloha 2 – Burp Suite Intercept

V této úloze bude představena funkce Intercept u Burp Suite, která ve spojení s Burp Proxy umožňuje zachytit a modifikovat provoz mezi klientem a webovou stránkou.

Cílem této úlohy bude využit funkcí Intercept pro modifikaci http požadavku takovým způsobem, aby bylo možné vytvořit uživatelský účet s administrátorským oprávněním.

Pro využití Burp Proxy se musí buď používat prohlížeč Burp Suite nebo nastavit Burp Proxy na vlastní prohlížeč, jak je uvedeno v sekci 5.2. Jednodušší je používat prohlížeč Burp Suite.

Nejdříve se musí otevřít Burp Suite. Ten se zeptá, zda je třeba vytvořit nový projekt nebo otevřít existující. Po zmačknutí Next se zeptá na konfiguraci, zase se nemusí nic dělat a pouze zmačknout Start Burp. Po otevření hlavního okna Burp Suite je třeba zvolit složku Proxy a pod ní složku Intercept:



Obrázek 16 Složka Intercept u Burp Suite. Zdroj: Autor

V tomto novém okně se zvolí Open Browser. V otevřeném prohlížeči je třeba zadat adresu OWASPJS. Na webové stránce OWAPJS je třeba zmačknout v pravém horním rohu Account a Login, a v nově otevřené stránce dole “Not yet a customer?” Tohle otevře stránku registrace. Všechna políčka se musí vyplnit (email atd. nemusí být skutečné). Před zmačknutím tlačítka registrace je ale třeba se vrátit zpět do Burp Suite a zmačknout “Intercept is off”. Po zmačknutí tlačítka Register se v Burp Suite objeví http požadavek:

Obrázek 17 Zachycený požadavek v Burp Suite. Zdroj: Autor

Horní polovina požadavku není moc zajímavá. Dole ale lze vidět údaje o nově vytvořeném účtu. Ve zvoleném místě mezi těmito údaji, ve stejném bloku lze napsat nový řádek:

```
„role“: „admin“,
```

Obrázek 18 Modifikovaný požadavek v Burp Suite. Zdroj: Autor

Pak se musí zmačknout tlačítko Forward v Burp Suite. Toto předá modifikovaný požadavek web serveru OWASPJS. Tlačítko je třeba zmačknout několikrát, aby zmizely všechny informace o požadavku v Burp. Účet bude úspěšně vytvořen s administrátorským oprávněním. Toto lze ověřit zadáním adresy http://127.0.0.1:3000/#/administration Účet, který nemá administrátorské oprávnění dostane chybu 403, ale účet, který byl vytvořen v tomto návodu úspěšně získá dostup k této sekci OWASPJS.

Vyhodnocení úkolu: Funkce Intercept byla úspěšně použita pro modifikaci http požadavku na OWASPJS, což umožnilo naplnit cíl vytvoření účtu s administrátorskými oprávněními. Tento účet má dostup k nejkritičtějším částem webové stránky.

6.5 Úloha 3 – SQL Injekce Pomocí SQLMAP

Cílem této úlohy je zjistit, zda je OWASPJS zranitelný vůči SQL injekcí, a pokud ano, zkusit využit tuto zranitelnost a spustit útok SQL injekci. To se provede ručním testem a pak pomocí nástroje SQLMAP, který byl vytvořen přesně pro tyto účely.

SQL Injekce se obvykle objeví na stránce, která obsahuje formulář (obvykle přihlašovací formulář), nebo na adrese s parametrem. Parametry dávají požadavku specifickou informaci, jako například konkrétní ID uživatele, jehož stránku je třeba zobrazit.

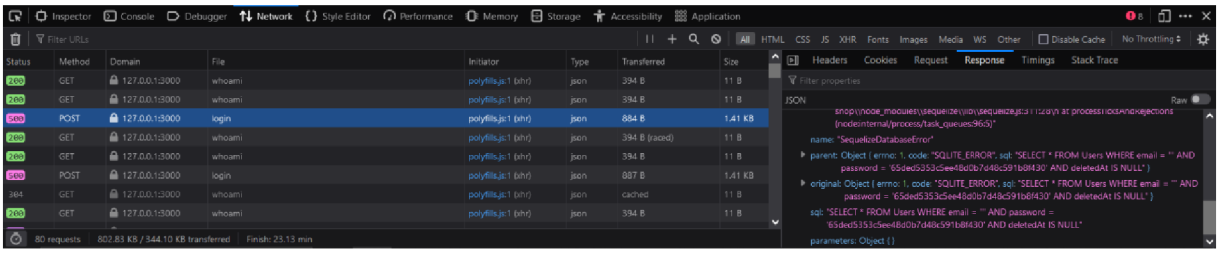
První parametr v URL adrese se zobrazí s pomocí otazníku, další parametry s pomocí ampersandu:

```
?parametr1=hodnota1&parametr2=hodnota2
```

Příklad adresy s parametrem:
`https://www.example.com/people?name=jan&surname=novak`

Pro nalezení adresy s parametrem na webové stránce, musíme najít část webové stránky, ve které se předává informace o specifickém elementu či skupině elementů. Když zkusíme vyhledat něco na OWASPJS, například slovo “juice”, je vidět, že URL adresa stránky se teď končí na `/search?q=juice`. Otazník říká, že tady se používá parametr **q** s hodnotou **juice**. Tato adresa může obsahovat zranitelnost SQL injekci.

SQL Injekci lze najít a využit ručně. Nejdříve zkusíme nalézt a využit injekci ve formulářích. Na stránce login v OWASPJS do políčka email napíšeme například uvozovku a do heslového políčka cokoliv. Než zmačkneme Log in, otevřeme konzoli prohlížeče a složku network. Zmačkneme Log in a podíváme se na odpověď post požadavku. V sekci JSON lze vidět chybu. Když otevřeme tuto chybu a posuneme se dolů, uvidíme SQL dotaz:



Obrázek 19 SQL dotaz v konzoli Firefox. Zdroj: Autor

Takovýto výsledek znamená, že uvedená data do políček se interpretovala jako SQL kód, a uvozovka způsobila chybu, protože je to část syntaxe SQL. Stránka login je proto zranitelná vůči SQL injekcím. Pro využití zranitelnosti lze napsat například:

```
' OR 1=1 --
```

Do emailového políčka a zase cokoliv do políčka hesla. Tento dotaz nás úspěšně přihlásí do prvního nalezeného účtu, který je naštěstí administrátorský účet. Tento dotaz je podmínka,

která vždycky vrací `TRUE`, a proto nás přihlásí do účtu. `--` znamená okomentování části dotazu, která následuje tento symbol, bez tohoto okomentování by dotaz nevrátil `TRUE`, protože by pak kontroloval, zda heslo v DB odpovídá heslu v políčku.

Teď je čas zkusit využit injekci v parametrech. V OWASPJS existuje funkce vyhledání produktů. Když se vyhledá produkt, zobrazí se hledaný text v URL adrese jako hodnota parametru q. Pro nalezení a využití zranitelností v tomto případě je třeba v OWASPJS použit trochu jinou URL adresu. Místo, například, http://127.0.0.1:3000/#/search?q=1 se má napsat http://127.0.0.1:3000/rest/products/search?q=1. U hodnoty parametru q je třeba místo 1 například uvozovku se středníkem. Výsledkem bude SQLITE (SQLITE je DB, která je využívaná OWASPJS) chyba, která říká, že tento parametr je zranitelný. Využití tohoto typu zranitelnosti ručně je trochu těžší než v případě login formuláře. Teď je čas využit nástroje SQLMAP.

Zkusíme provést skenování stejné stránky s parametrem q a zjisti pomocí SQLMAP, zda je q zranitelný. Nejjednodušší forma skenu se píše takto:

```
sqlmap -u "<adresa>"
```

Parametr `-u` označuje cílovou URL adresu. Adresa bude zase http://127.0.0.1:3000 Pokud se ale takhle provede sken, SQLMAP nám napíše, že parametr `q` není zranitelný. SQLMAP má k dispozici různé metody skenování SQL injekcí. Některé z nich jsou velmi náročné, a proto se nepoužívají při základním skenu. Existují 2 parametry `--risk` a `--level`. Oba mají různé stupně: Risk je od 1 do 3 a level je od 1 do 5. Při základním skenu oba tyto parametry budou nastaveny na první stupeň. Pro nalezení zranitelnosti je často potřeba zvýšit tyto stupně. Při zvýšení stupně se ale skeny budou trvat mnohem déle. Pro zjištění přesných stupňů těchto dvou parametrů je třeba experimentovat.

Pro úspěšné nalezení zranitelnosti v parametru `q` použijeme následující příkaz:

```
sqlmap -u "<adresa>" --risk=3 --level=3 --batch --dbms=sqlite
```

V tomto příkazu jsou další 2 parametry: `batch` a `dbms`. `batch` se používá pro usnadnění práce se SQLMAP. Při skenech se SQLMAP často zeptá různé otázky typu ano/ne, které ovlivní průběh skenu. Každá z těchto otázek má nastavenou defaultní hodnotu a obvykle je nejlépe ji zvolit. `batch` automaticky zvolí defaultní odpověď pro každou otázku.

Parametr `dbms` informuje SQLMAP o použitém databázovém serveru. Při ruční analýze jsme zjistili, že databáze OWASPJS je SQLITE. Uvedení typu DB parametrem `dbms` hodně sníží dobu skenu.

Obrázek 20 Výsledek skenu SQLMAP ukazuje zranitelnost u parametru q. Zdroj: Autor

SQLMAP úspěšně určil parametr `q` jako zranitelný. Teď je čas zranitelnost využit.

Jedna z možností je zkusit získat obsah určité tabulky v databázi.

Nejdříve je potřeba vědět, jaké tabulky vůbec existují. Je třeba získat jejich názvy. To se jednoduše udělá příkazem:

```
sqlmap -u "127.0.0.1:3000/rest/products/search?q=1" --tables
```

Toto vrací jména všech tabulek v DB. Nejzajímavější je určitě Users. Zkusíme získat dostup k jejímu obsahu:

```
sqlmap -u "127.0.0.1:3000/rest/products/search?q=1" --dump -T Users
```

Parametr `--dump` znamená získání všech záznamů určité tabulky. Lze použit i parametr `--dump-all`, který získá záznamy všech tabulek, ale to by trvalo příliš dlouho. Pomocí parametru `-T` se uvádí tabulka, jejíž záznamy je třeba získat.

Ve výsledku je vidět jména sloupců uvedené tabulky a pak záznamy – je vidět jméno každého uživatele, jeho email adresu, (zašifrované) heslo, roli a další atributy. SQLMAP také automaticky zkusil použit Dictionary útok (z defaultního slovníku, které se nachází ve složce obsahující SQLMAP) a podařilo se mu odhadnout některá hesla. Například, podle výsledků SQLMAP, heslo uživatele s emailem J12934@juice-sh.op je admin123. Když se ale zkusíme přihlásit s těmito údaji, bohužel se to nepodaří. To je proto, že hesla a emaily v těchto výsledcích se neodpovídají (je to problém s SQLMAP). admin123 je skutečné heslo uživatele OWASPJS, ale ve skutečnosti je to heslo uživatele s emailem admin@juice-sh.op.

Obrázky výsledků lze najít v příloze č.3.

Výhodnocení úkolu: S pomocí ručního testu a pak nástroje SQLMAP bylo zjištěno, že OWASPJS je opravdu zranitelný vůči SQL injekci. Zranitelnost byla využitá výpisem tabulek a jejích obsahu, což umožnilo provést útok na heslo a získat dostup k některým účtům.

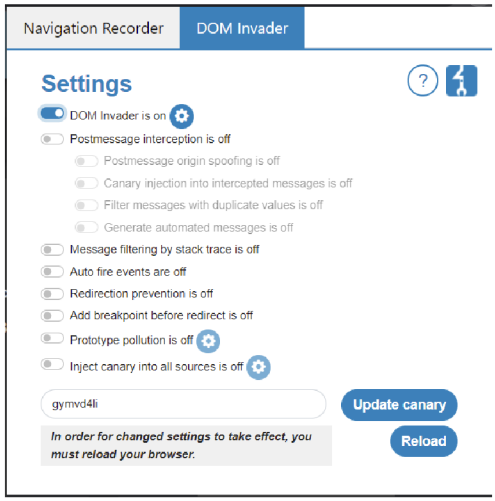
6.6 Úloha 4 – DOM XSS Pomocí Burp Suite

U této úlohy bude cílem ukázat použití nové funkcí Burp Suite – DOM Invader, která umožní najít a využit zranitelnost typu DOM XSS na OWASPJS.

DOM XSS je jeden z typů XSS (Cross-Site Scripting) útoků, o kterých se krátce mluvilo v části o běžných zranitelnostech.

„S použitím JS skriptu, XSS útok založený na DOM [Document Object Model] využívá zranitelností XSS v DOM, která se vyskytuje na straně uživatele v průběhu zpracování dat. Jak vyplývá z jeho názvu, tento typ XSS útoku je implementován přes DOM, což je platformově a jazykově nezaujaté rozhraní, které dává aplikacím a skriptům přístup k obsahu HTML a XML dokumentů a modifikuje jejich obsah, tvar a provedení. S nesprávným filtrováním může být DOM napadeného webu modifikován a zlomyslný JS kód může být v napadené stránce spuštěn.“ **[39]**

Zapnutí DOM Invader: DOM Invader je dostupný pouze v integrovaném prohlížeči Burp Suite a je ho taky potřeba zapnout. Pro zapnutí DOM Invader se nejdříve musí otevřít Burp Browser (Burp Suite -> Proxy -> Intercept -> Open Browser) pak v prohlížeči zmačknutím ikonky skládačky vpravo nahoře se otevře okno rozšíření. V otevřeném okně je třeba zmačknout Burp Suite. Po zmačknutí se otevře další okno, ve kterém se musí zvolit složka DOM Invader a zmačknout se tlačítko „DOM Invader is off“.



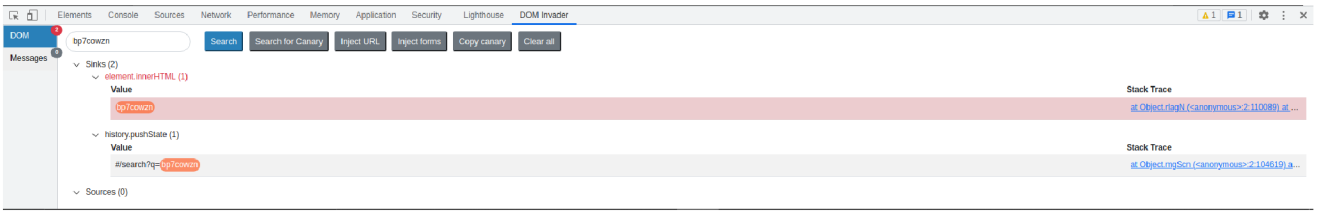
Obrázek 21 Okno DOM Invader v prohlížeči Burp Suite. Zdroj: Autor

Pak se musí restartovat prohlížeč. DOM Invader teď lze najít v konzoli prohlížeče, (ve složce DOM Invader), která se otevře zmačknutím F12.

V konzoli lze vidět určitý řetězec. Tento řetězec se jmenuje “canary“ (kanárek). Je to náhodně generovaný unikátní řetězec, který by neměl vyskytovat kdekoli na cílové webové stránce a budeme ho používat v některých částech stránky, což umožní DOM Invader zjistit, do které části kódu tento řetězec přechází. Toto nám pomůže najít zranitelnosti DOM XSS.

Zkusíme zase analyzovat parametr q, stejně jako v úloze o SQL injekci. Otevřeme OWASPJS v Burp prohlížeči a otevřeme konzoli.

Zkopírujeme kanárka z konzoli a zkusíme ho vyhledat pomocí funkci hledání nahoře. V konzoli se objeví něco podobného chybě. Není to chyba, ale informace o tom, kam přechází (v kódu) hledaný řetězec.



Obrázek 22 Výstup v konzoli DOM Invader. Zdroj: Autor

Z výstupu je vidět, že se řetězec používá ve funkci `element.innerHTML`. Pokud teď zmačkneme Inject URL, vedle tohoto výstupu by se mohlo objevit také zelené tlačítko **Exploit**, při jehož zmačknutí se stránka automaticky přesměruje na adresu, která demonstruje tuto zranitelnost v parametru. Například: http://127.0.0.1:3000/?x=<img src onerror=alert(1)>#/search?q=<img src onerror%3Dalert(1)>. Na této adrese se spustí JS funkce `alert(1)`, která zobrazí číslo 1 na obrazovce. Je to demonstrace toho, že parametr q (a také určitý nalezený parametr x) je zranitelný vůči DOM XSS a místo `alert(1)` může útočník napsat libovolný JS kód, a ten bude úspěšně spuštěn.

Bohužel tlačítko Exploit se nemusí vždycky objevit, údajně to záleží na verzi Burp Suite nebo operačním systému. V takových případech lze parametr testovat ručně. Často se používá JS kód v tagu `<script>`, například: `http://127.0.0.1:3000/#/search?q=<script>alert(1)</script>`. To ale nebude fungovat na OWASPJS a musí se použít jiný způsob, jako například URL adresa s `img` uvedená nahoře nebo html element **iframe**: `http://127.0.0.1:3000/#/search?q=<iframe src="javascript:alert(`xss`)">` (iframe způsob je převzat z **[40]**)

V těchto způsobech se zase objeví zpráva na obrazovce kvůli použití JS funkce `alert`, a tuto funkci zase lze nahradit nějakou škodlivější funkcí.

DOM XSS zranitelnosti se mohou někdy objevit i v HTML formulářích. Kromě toho, je možné zkusit automaticky najít zranitelnosti s použitím tlačítek Inject URL a Inject Forms. Nemusí to ale vždy fungovat a někdy může způsobit problémy s webovou stránkou. Proto je lepší zranitelnosti hledat ručně, kopírováním kanárky do hodnoty parametrů a do políček ve formulářích a analyzování výstupu v konzoli.

Vyhodnocení úkolu: V úloze bylo zjištěno, že OWASPJS je zranitelný vůči DOM XSS. Cíl úlohy byl naplněn exploatací této zranitelnosti ručně a automaticky pomocí nástroje DOM Invader, což umožňuje útočníkovi spustit na cílové stránce zlomyslný JS kód.

6.7 Úloha 5 – Základní zkoumání s Nmap

Cílem této úlohy je demonstrace základních zkoumání nástroje Nmap na příkladu základního zkoumání cílové adresy. Kromě základního skenu budou ještě demonstrovány nejdůležitější parametry Nmap a na konci ukázka volitelného GUI. Jako cílová adresa bude tentokrát použitá adresa http://scanme.nmap.org/, což je speciální stránka vytvořená speciálně pro testování funkcí Nmap.

Nejjednodušší příkaz neobsahuje žádné parametry a vypadá takto: `nmap <adresa>`. Adresa může být ve formě IP adresy nebo názvu domény, nesmí obsahovat předponu http/https či www a nesmí obsahovat číslo portu (port lze určit parametrem). Příklad: `nmap 127.0.0.1`, `nmap scanme.nmap.org`. Základní sken bez parametrů zkontroluje 1000 nejčastějších portů na cíli. Použité cílem porty budou vypsány v terminálu spolu s jejich stavem (otevřeno/zavřeno) a službou spuštěnou na uvedeném portu.

Obrázek 23 Výsledek základního skenu – 4 otevřené porty. Zdroj: Autor

S pomocí parametru `-sV` Nmap prozkoumá otevřené porty se snahou najít informace o verzích výše uvedených služeb a může určit i operační systém. Například při skenu stejné adresy urči Nmap, že operační systém, na kterém je hostována stránka scanme.nmap.org je Linux.

Parametr `-O` také umí určit operační systém, ale sken s tímto parametrem může trvat dlouho a potřebuje root privilegii. Při dlouhých zkoumáních lze zmačknout klávesu enter a Nmap vypíše v procentech pokrok skenu.

Mnoho Nmapových skenů mohou být jednoduše detekovány cílovým serverem. Proto má Nmap tajný režim skenu (parametr `-sS`), při kterém jsou menší šance, že skeny Nmap budou detekovány. Nmap má také parametr `-A`, který provede agresivní sken, což je opakem tajného skenu, protože má větší šanci být detekovaným. Agresivní sken provede zkoumání operačního systému, verzí služeb, skenování skriptů a provedení traceroute.

Nmap Scripting Engine (NSE): Kromě spuštění různých režimu zkoumání pomocí parametrů, lze také využit skriptovacího stroje Nmap. NSE umožňuje uživateli napsat vlastní skript v jazyce LUA pro splnění jiných úkolů než povoluje sám Nmap. Nmap má více než 600 oficiálních skriptů, které jsou defaultně instalované s Nmap. Seznam těchto skriptů lze najít na stránce https://nmap.org/nsedoc/scripts/ a na stránce https://nmap.org/book/nse-usage.html#nse-categories lze najít kategorie skriptů.

Skript lze spustit pomocí příkazu:

```
nmap --script "nazev-skriptu" adresa
```

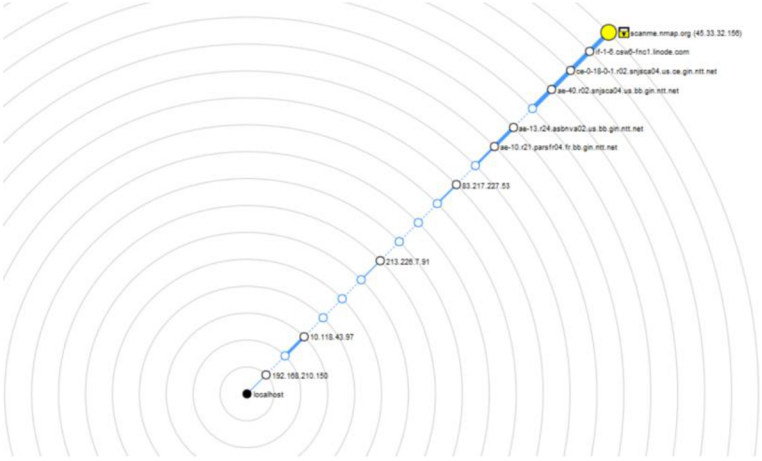
Přičemž lze spustit nejenom 1 skript, ale i celou kategorii (místo názvu skriptu napsat název kategorie, například `vuln` (vulnerability)). Lze také používat booleovské výrazy `and`, `or`, `not` pro pokročilý výběr skriptů.

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap --script vuln 127.0.0.1
Starting Nmap 7.92 ( https://nmap.org ) at 2022-08-02 08:16 EDT
Pre-scan script results:
| broadcast-avahi-dos:
|   Discovered hosts:
|     224.0.0.251
|   After NULL UDP avahi packet DoS (CVE-2011-1002).
|_  Hosts are all up (not vulnerable).
Nmap scan report for localhost (127.0.0.1)
Host is up (0.0000070s latency).
Not shown: 999 closed tcp ports (reset)
PORT     STATE SERVICE
3000/tcp open  ppp

Nmap done: 1 IP address (1 host up) scanned in 34.44 seconds
```

Obrázek 24 Skenování OWASPJS pomocí skriptů kategorie vuln. Zdroj: Autor

Všechny tyto funkce lze provést i s použitím GUI nástroje Zenmap. Pro zkoumání sítí stačí terminál a Zenmap nebyl vytvořen pro nahrazení příkazového řádku, ale může být jednodušší pro začátečníky a obsahuje některé užitečné funkce. Umí například vytvořit interaktivní topologii objevených sítí:



Obrázek 25 Interaktivní topologie v Zenmap. Zdroj: Autor

Vyhodnocení úkolu: Cíl úlohy byl splněn demonstrací základního skenu a základních parametrů Nmap. Ve výsledcích provedených skenů byly nalezené použité porty a služby spuštěné na nich, a také nalezené hosty.

6.8 Úloha 6 Útoky na hesla pomocí Burp Intruder

Cílem této úlohy je provést útok na hesla typu Dictionary (česky slovník) pomocí funkcí Intruder u Burp Suite. Dictionary útok znamená zadávání slov z existujícího slovníku jako login a heslo, s cílem získat dostup k odpovídajícímu uživatelskému účtu. Před začátkem útoků bude nejdříve demonstrováno "připojení" Intruderu k cílové stránce, aby věděl Intruder, v jakém místě se nachází uživatelské jméno a heslo. Pak se ukáže vytvoření slovníku a vysvětlí se režimy útoků v Intruderu.

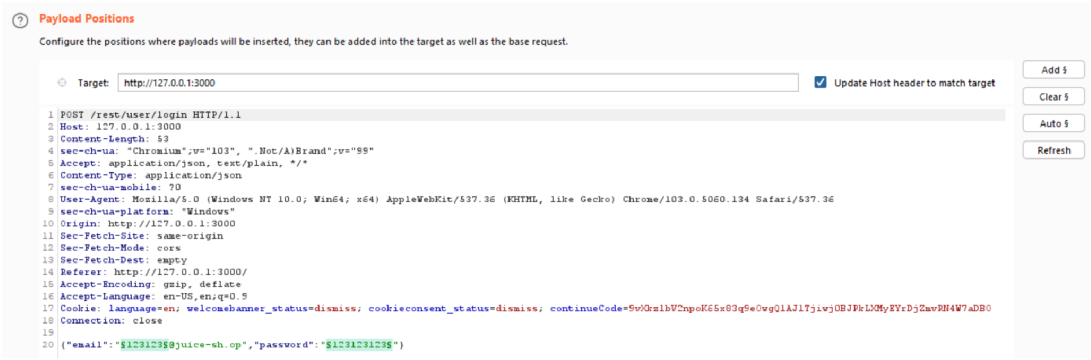
Cílovou stránkou bude zase OWASPJS. Ten obsahuje v databázi několik účtu a některé z nich mají slabá hesla.

Pro použití Intruder nejdříve se musí provést funkce Intercept ve složce Proxy stejně jako v 2. úloze. Budeme muset zachytit přihlášení se do OWASPJS, pokud ve výstupu není vidět přihlašovací údaje, zmačknout tlačítko Forward. Až bude vidět přihlašovací údaje, zmačknout pravém tlačítkem myši kdekoli na oblasti výstupu a zvolit možnost Send to Intruder.

Teď ve složce Intruder, v podsložce Positions je třeba nastavit pozice payloadu. Payload je seznam řetězců – potenciálních hesel/emailů. Pozice payloadů jsou místa, kam budeme ty řetězci vkládat, tedy v našem případě musíme ukázat Intruderu kde v kódu se nachází přihlašovací údaje. Pozicí payloadu může být například i parametr URL adresy, protože Burp Intruder se může používat i, například, pro SQL injekce. V této úloze ale budeme provádět pouze útok na heslo, proto je třeba nastavit jako pozice pouze email a heslo.

V pravé části okna jsou tlačítka pro přidání a smazání pozic payloadů. Pozice payloadů je označena dvěma symboly § a zelenou barvou v kódu požadavku. Defaultně jsou nastaveny některé pozice, a nejenom přihlašovací údaje. Musíme smazat všechny pozice tlačítkem Clear §. Pak přidáme hodnotu emailu a hesla jako pozice payloadu zmačknutím tlačítka Add §.

Pozor: V OWASPJS se používá pro přihlášení email místo uživatelského jména. Defaultní emaily účtu v databázi OWASPJS mají všechny formu xxxxx@juice-sh.op. To znamená že řetězci, které budeme chtít použít jako email, musí být v tomto formátu. Pokud si stáhneme seznam takovýchto řetězců z internetu, nebudou v tomto formátu a Intruder nebude moci prolomit žádný účet. To se řeší tím, že jako pozice payloadu nastavíme jenom tu část emailu, která předchází zavináči:

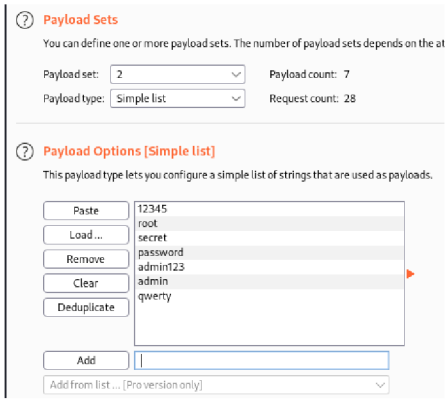


Obrázek 26 Správně nastavené pozice payloadu. Zdroj: Autor

Po nastavení pozic payloadů je třeba zvolit režim útoku. Existují 4 režimy:

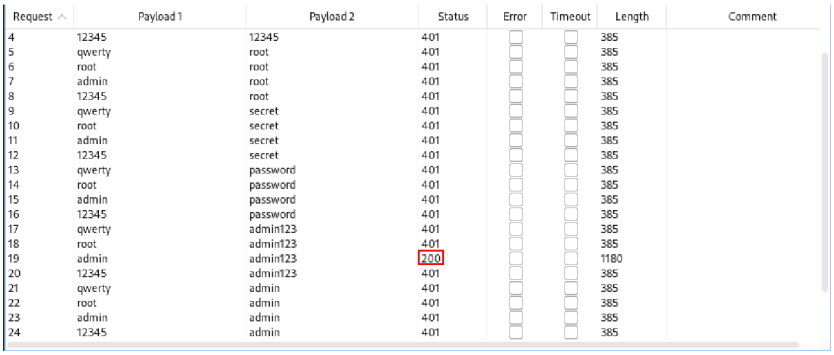
- **Sniper** – Tento režim používá jednu sadu payloadů a postupně umístí všechny payloady do každé pozice – tento režim je dobrý pouze když nepotřebujeme prolomit více než jednu pozici najednou. Nehodí se v případě odhadnutí přihlašovacích údajů.
- **Battering Ram** – Taky používá jednu sadu payloadů, ale dává každý payload do všech pozicí najednou, aby se ten režim hodil v našem případě, muselo by být uživatelské heslo stejné jako email, což by se nikdy nemělo stát.
- **Pitchfork** – Tento režim už používá několik sad payloadů a funguje tak, že první payload z prvního seznamu umístí do první pozice, první payload z druhé sady umístí do druhé pozice atd. To se hodí, když je hodnota v jedné pozici nějak spojená s hodnotou v druhé pozici.
- **Cluster Bomb** – Nejefektivnější, ale i nejpomalejší způsob. Používá několik sad payloadů. Do pozic umístí všechny kombinace payloadů, tedy do první pozice umístí první payload z první sady, do druhé pozice umístí první payload z druhé sady, pak druhý payload z druhé sady atd. až bude na konci druhé sady, umístí druhý payload první sady do první pozice, a zase začne procházet druhou sadu atd. Tedy když máme 2 pozici, 5 payloadů v první sadě a 10 payloadů v druhé sadě, bude počet možností celkem 5x10 = 50. Při dlouhých seznamech payloadů tento způsob bude velmi pomalý, ale je to nejlepší způsob pro účely odhadnutí přihlašovacích údajů.

Zvolíme režim Cluster Bomb a vytvoříme sady payloadů v podsložce Payloads. Sady payloadů lze vytvořit ručně nebo načíst ze souboru. V pro verzi Burp Suite existuje předem vytvořený seznam, který lze hned načíst. Lze ale stáhnout seznam nejčastěji používaných hesel a loginů z internetu. Pro účely demonstrace stačí vytvořit seznam ručně. První sada bude nasazená do pole emailu, druhá do pole hesla. Nahoře v sekci Payload Sets se zvolí sada payloadu, a v sekci Payload options lze přidat payloady do seznamu tlačítkem Add. Typ payloadu by měl zůstat defaultní Simple List.



Obrázek 27 Sada payloadů. Zdroj: Autor

Po vytvoření sad payloadů můžeme spustit útok tlačítkem Start Attack v pravé části okna. Burp Suite nás upozorní, že regulární verze obsahuje pouze demo verzi Intruderu a že Intruder bude v této verzi pomalý. Je to problém, ale jenom při větších sadách payloadů.



Request	Payload 1	Payload 2	Status	Error	Timeout	Length	Comment
4	12345	12345	401			385	
5	qwerty	root	401			385	
6	root	root	401			385	
7	admin	root	401			385	
8	12345	root	401			385	
9	qwerty	secret	401			385	
10	root	secret	401			385	
11	admin	secret	401			385	
12	12345	secret	401			385	
13	qwerty	password	401			385	
14	root	password	401			385	
15	admin	password	401			385	
16	12345	password	401			385	
17	qwerty	admin123	401			385	
18	root	admin123	401			385	
19	admin	admin123	200			1180	
20	12345	admin123	401			385	
21	qwerty	admin	401			385	
22	root	admin	401			385	
23	admin	admin	401			385	
24	12345	admin	401			385	

Obrázek 28 Výsledek útoku pomocí Burp Intruder. Zdroj: Autor

Úspěšně odhadnuté přihlašovací údaje jsou označeny stavem 200 (http odpověď OK). Z obrázku je vidět úspěšně odhadnuté přihlašovací údaje admin – admin123. Při přihlášení se s těmito údaji je třeba si pamatovat přidat "@juice-sh.op" do prvního payloadu (admin), protože je to emailová adresa.

Vyhodnocení úkolu: Nástroj Intruder byl úspěšně použit pro Dictionary útok na heslo. Ukázalo se vytvoření správného slovníku běžných loginů a hesel, což umožnilo najít uživatelský účet, který používal login a heslo uvedené ve slovníku. Kromě demonstrace funkcí Intruder byly také popsány všechny režimy Dictionary útoku v tomto nástroji.

6.9 Úloha 7 WPScan

V této úloze se provede zkoumání webové stránky, vytvořené s pomocí WordPress nástrojem WPScan. Cílem je najít zranitelnosti v starší verzi WordPress a starších verzích její pluginů.

Stejně jako v případě jiných nástrojů, nesmí se skenovat webová stránka bez souhlasu její majitele, a proto se musí vytvořit vlastní. Lokální instalace OWASP Juice Shop, která byla použitá v minulých úlohách, tentokrát fungovat nebude, protože není vytvořena s pomocí WordPress. Musíme proto nainstalovat WordPress a vytvořit stránku pomocí něj.

Návod na instalaci WordPress lze najít na oficiální webové stránce https://wordpress.org/support/article/how-to-install-wordpress/. Způsobů instalace ale existuje několik. Lze například nainstalovat a spustit webovou stránku WordPress s využitím software XAMPP.

Po správné instalaci a spuštění lokálního WordPress serveru můžeme spustit WPScan.

Stejně jako každý jiný nástroj, WPScan má spoustu různých parametrů. Pro účely demonstrace ale stačí spustit jenom základní zkoumání:

```
wpscan --url <adresa>
```

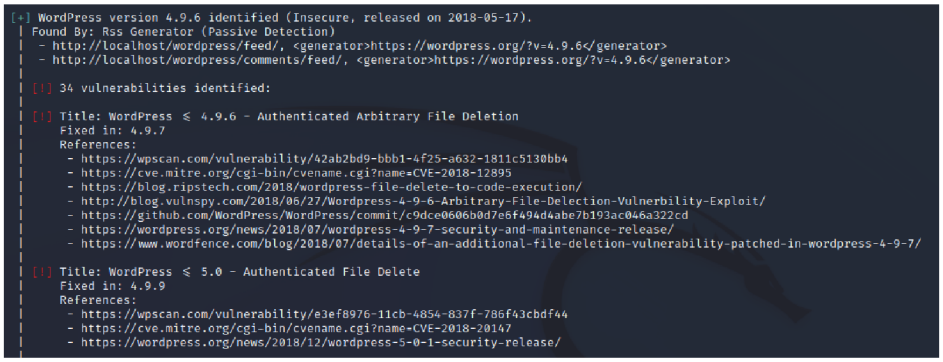
Takovýto sken naleze různé důležité informace, jako například verzi WordPress, informace o aktivních pluginech a jejich verze, použitý motiv, hlavičky a zálohy souborů konfigurace.

Neposkytne ale žádnou informaci o zranitelnostech. Pro zkoumání zranitelností vyžaduje WPScan API klíč. Tento API klíč lze získat po registraci na oficiální stránce WPScan https://wpscan.com/wordpress-security-scanner. Po registraci lze najít API klíč ve složce Profile https://wpscan.com/profile.

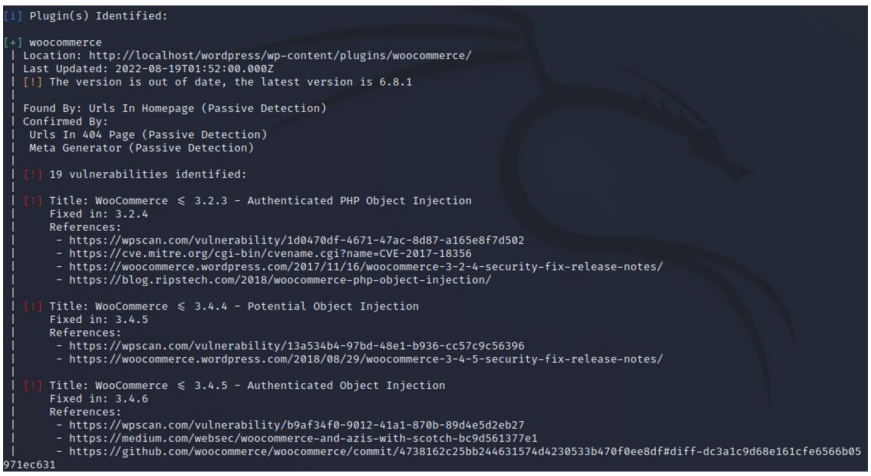
Po získání API klíče se musí klíč vždy přidávat jako další parametr k příkazům WPScan:

```
wpscan --url <adresa> --api-token <token>
```

Zkoumání s tímto parametrem už bude hledat zranitelnosti v samotném WordPress a v instalovaných a aktivních pluginech. V následujících obrázcích lze vidět malou část výstupu tohoto základního zkoumání:



Obrázek 29 Část nalezených zranitelností ve WordPress. Zdroj: Autor



Obrázek 30 Část nalezených zranitelností v pluginu Woocommerce. Zdroj: Autor

Z obrázků je vidět, že bylo nalezeno 34 zranitelností v samotném WordPress a dalších 19 v pluginu Woocommerce. Zranitelností je tak mnoho, protože pro účely demonstrace autor schválně nainstaloval starou verzi WordPress a také starou verzi Woocommerce. V posledních verzích WordPress a Woocommerce pravděpodobně nebude nalezena žádná zranitelnost.

Poslední důležitá poznámka je o tom, že existuje omezení na zkoumání zranitelností ve formě maximálního počtu požadavků – 75 za den. V předchozím zkoumání byly využité hned 3 požadavky. Pro vyšší počet požadavků za den a některé další možnosti je třeba si koupit placenou verzi WPScan.

Vyhodnocení úkolu: U této úlohy byl zkoumán lokální server WordPress s pomocí WPScan. Nejdříve se ukázalo správné nastavení API klíče u WPScan a pak se základním skenem byly nalezené zranitelnosti (které lze vidět na obrazcích) ve schválně nainstalované staré verzi WordPress a pluginu Woocommerce.

6.10 Úloha 8 Instalace a aktualizace Metasploit

Cílem této úlohy je poskytnout návod pro instalaci a aktualizaci Metasploit Framework pro všechny podporované operační systémy (návod je zpracován podle **[41]**)

Pokud uživatel používá Kali Linux verze 2.0 a více, pak není třeba Metasploit instalovat, protože už je v této verzi nainstalován.

Požadavky: Metasploit Framework podporuje operační systémy Windows, Linux a OSX. Pro instalaci je třeba vypnout anti-virus, nebo v anti-vírovém software vyloučit složku Metasploit. Je to proto, že antivirus může identifikovat Metasploit jako zlomyslný software, i když není. Ze stejných důvodů je třeba vypnout i firewall. Je třeba také mít administrátorská oprávnění na operačním systému.

Po instalaci je možné začít používat Metasploit z příkazového řádku pomocí vstupu do **msfconsole**.

Instalace na Windows

Je třeba si stáhnout instalátor pro Windows z

http://windows.metasploit.com/metasploitframework-latest.msi

pak spustit instalátor, přijat licenční smlouvu, zvolit složku, ve které bude instalován Metasploit a pokračovat v instalaci.

Po instalaci lze napsat v příkazovém řádku `msfconsole.bat` pro vstupu do msfconsole.

Instalace na Linux

V terminálu je třeba napsat následující příkaz:

```
curl https://raw.githubusercontent.com/rapid7/metasploit-
omnibus/master/config/templates/metasploit-framework-wrappers/msfupdate.erb
> msfinstall && chmod 755 msfinstall && ./msfinstall
```

Vstup do msfconsole:

```
./msfconsole
```

Terminál se zeptá, zda uživatel chce nastavit novou databázi. Pro nastavení DB je třeba napsat `y` nebo `yes`.
Pro zkontrolování, zda DB byla nastavena je možné napsat `db_status`. Pokud ve výstupu bude napsáno `postgresql connected to msf`, je DB nastavena úspěšně.

Instalace na OSX
Instalátor pro OSX je možné si stáhnout z http://osx.metasploit.com/metasploitframework-latest.pkg .

Proces instalace na OSX je pak úplně stejný jako na Windows.

Aktualizace Metasploit Framework: Nejjednodušší způsob, jak aktualizovat Metasploit je napsat v příkazovém řádku/terminálu příkaz `msfupdate`. Tento způsob ale nefunguje v případech, když je Metasploit částí operačního systému, což platí u **Kali Linux**. V případě Kali Linux je možné aktualizovat Metasploit příkazem `sudo apt update`.

Vyhodnocení úkolu: V této úloze byly poskytnuty požadavky pro instalaci Metasploit Framework, pak byly popsány návody na instalaci Metasploit Framework na operačních systémech Windows, Linux a OSX a na konci byl uveden způsob aktualizace tohoto nástroje.

6.11 Úloha 9 Instalace operačního systému Metasploitable

V této úloze se ukáže návod na instalaci operačního systému Metasploitable, o kterém se mluvilo v teoretické části, s pomocí software VMWare Workstation. Pak se ukáže získání IP adresy potřebné pro použití Metasploitable jako cíl funkcí Metasploit Framework.

Metasploitable lze najít na oficiální stránce Rapid7:

https://docs.rapid7.com/metasploit/metasploitable-2/

Tento operační systém je na tomto odkazu dostupný ke stažení zadarmo, ve formě virtuálního stroje. Pro instalaci je třeba si nejdříve nainstalovat software VMWare Workstation nebo VirtualBox či jiný software umožňující použití virtuálních strojů.

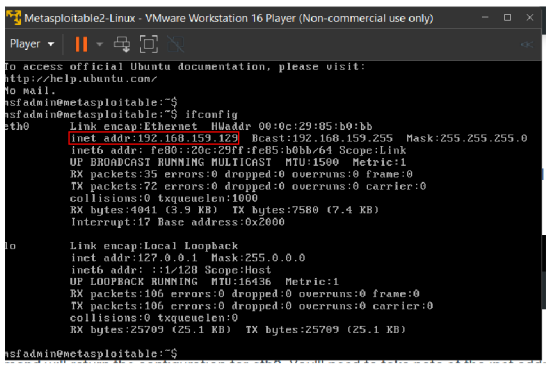
Instalace s pomocí VMWare Workstation:

Z odkazu uvedeného výše je třeba si stáhnout zip soubor Metasploitable. Zip soubor je třeba rozbalit někde na počítači.

Pak je třeba spustit VMWare Workstation. V jeho rozhraní zmačknout „Open a Virtual Machine“ a zvolit `.vmx` soubor uvnitř rozbalené složky Metasploitable.

Metasploitable se teď objeví na levé straně rozhraní VMWare Workstation. Dvojitým zmačknutím myši na Metasploitable lze ho spustit. Objeví se příkazový řádek a zeptá se uživatele na přihlašovací údaje. Jméno a heslo jsou stejné – **msfadmin**.

Posledním krokem je získání IP adresy virtuálního stroje, aby Metasploitable bylo možné použít jako cíl při testování Metasploit Framework. IP adresu lze získat pomocí příkazu `ifconfig`. Správná adresa je první “`inet addr`” ve výstupu:



Obrázek 31 Výstup z `ifconfig` a správná adresa. Zdroj: Autor

Vyhodnocení úkolu: Popsáním návodu na instalaci Metasploitable s pomocí VMWare Workstation, demonstrací příkazu ifconfig a uvedením správné IP adresy byly splněny cíle tohoto úkolu.

6.12 Úloha 10 nalezení a využití zranitelnosti v Metasploit

Cílem této úlohy je na příkladu ukázat nalezení zranitelnosti, která umožní získat dostup do cílového systému s pomocí vzdáleného ovládání. Zranitelnost se naleze pomocí skenerů v tzv. auxiliary modulu Metasploit a následně se využije. Cílovým systémem bude virtuální stroj Metasploitable 2, což je nejlepší způsob testování funkcí Metasploit Framework.

Po spuštění Kali Linux a otevření terminálu je třeba vstoupit do konzoli Metasploit s pomocí příkazu `msfconsole`. Teď lze pracovat s příkazy Metasploit.

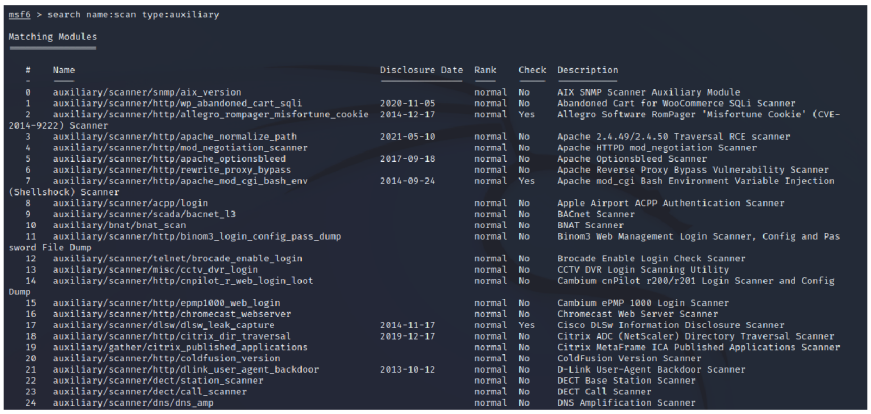
Nejdříve je třeba najít zranitelnost. V Metasploit existuje spousta skenerů zranitelností v auxiliary modulu. Seznam těchto skenerů lze najít s pomocí příkazu `search`:

```
search name:scan type:auxiliary
```

Tento příkaz naleze všechny příkazy, které jsou v modulu auxiliary a obsahují klíčové slovo **scan**. Na výstupu bude seznam těchto příkazu a bude obsahovat jejich jméno, datum, kdy byla tato zranitelnost zveřejněna, jeho rank, "Check" a krátký popis.

Rank exploitu záleží na jeho vlivu na cílový systém a na šanci jeho úspěchu. Nejlepší rank je excellent.

Check říká, zda je u tohoto exploitu možný příkaz `check`, který kontroluje, jestli je cílový systém zranitelný vůči tomuto exploitu.



```
msf6 > search name:scan type:auxiliary

Matching Modules
================

   #    Name                                                     Disclosure Date  Rank    Check  Description
   -    ----                                                     ---------------  ----    -----  -----------
   0    auxiliary/scanner/snmp/aix_version                                        normal  No     AIX SNMP Scanner Auxiliary Module
   1    auxiliary/scanner/http/wp_abandoned_cart_sqli            2020-11-05       normal  No     Abandoned Cart for WooCommerce SQLi Scanner
   2    auxiliary/scanner/http/allegro_rompager_misfortune_cookie 2014-12-17      normal  Yes    Allegro Software RomPager 'Misfortune Cookie' (CVE-
2014-9222) Scanner
   3    auxiliary/scanner/http/apache_normalize_path             2021-05-10       normal  No     Apache 2.4.49/2.4.50 Traversal RCE scanner
   4    auxiliary/scanner/http/mod_negotiation_scanner                            normal  No     Apache HTTPD mod_negotiation Scanner
   5    auxiliary/scanner/http/apache_optionsbleed               2017-09-18       normal  No     Apache Optionsbleed Scanner
   6    auxiliary/scanner/http/rewrite_proxy_bypass                               normal  No     Apache Reverse Proxy Bypass Vulnerability Scanner
   7    auxiliary/scanner/http/apache_mod_cgi_bash_env           2014-09-24       normal  Yes    Apache mod_cgi Bash Environment Variable Injection
(Shellshock) Scanner
   8    auxiliary/scanner/acpp/login                                              normal  No     Apple Airport ACPP Authentication Scanner
   9    auxiliary/scanner/scada/bacnet_l3                                         normal  No     BACnet Scanner
   10   auxiliary/bnat/bnat_scan                                                  normal  No     BNAT Scanner
   11   auxiliary/scanner/http/binom3_login_config_pass_dump                      normal  No     Binom3 Web Management Login Scanner, Config and Pas
sword File Dump
   12   auxiliary/scanner/telnet/brocade_enable_login                             normal  No     Brocade Enable Login Check Scanner
   13   auxiliary/scanner/misc/cctv_dvr_login                                     normal  No     CCTV DVR Login Scanning Utility
   14   auxiliary/scanner/http/cnpilot_r_web_login_loot                           normal  No     Cambium cnPilot r200/r201 Login Scanner and Config
Dump
   15   auxiliary/scanner/http/epmp1000_web_login                                 normal  No     Cambium ePMP 1000 Login Scanner
   16   auxiliary/scanner/http/chromecast_webserver                               normal  No     Chromecast Web Server Scanner
   17   auxiliary/scanner/dlsw/dlsw_leak_capture                 2014-11-17       normal  Yes    Cisco DLSw Information Disclosure Scanner
   18   auxiliary/scanner/http/citrix_dir_traversal              2019-12-17       normal  No     Citrix ADC (NetScaler) Directory Traversal Scanner
   19   auxiliary/gather/citrix_published_applications                            normal  No     Citrix MetaFrame ICA Published Applications Scanner
   20   auxiliary/scanner/http/coldfusion_version                                 normal  No     ColdFusion Version Scanner
   21   auxiliary/scanner/http/dlink_user_agent_backdoor         2013-10-12       normal  No     D-Link User-Agent Backdoor Scanner
   22   auxiliary/scanner/dect/station_scanner                                    normal  No     DECT Base Station Scanner
   23   auxiliary/scanner/dect/call_scanner                                       normal  No     DECT Call Scanner
   24   auxiliary/scanner/dns/dns_amp                                             normal  No     DNS Amplification Scanner
```

Obrázek 32 Část výstupu příkazu search. Zdroj: Autor

Podrobnější popis každého příkazu lze najít na https://www.offensive-security.com/metasploit-unleashed/.

Na Metasploitable běží server programu VNC. VNC (Virtual Network Computing) je program umožňující vzdálené ovládání jiného počítače. Jeden z auxiliary skenerů je skener `auxiliary/scanner/vnc/vnc_login`, který může odhadnout heslo k serveru VNC na cílovém systému.

Pro použití tohoto skenerů je třeba napsat příkaz:

```
use auxiliary/scanner/vnc/vnc_login
```

Příkaz `use` vlastně "zvolí" určený modul (v tomto případě vnc skener) aby ho bylo možné spustit. Dále je třeba nastavit IP adresu s pomocí příkazu:

```
set RHOST <IP_Address>
```

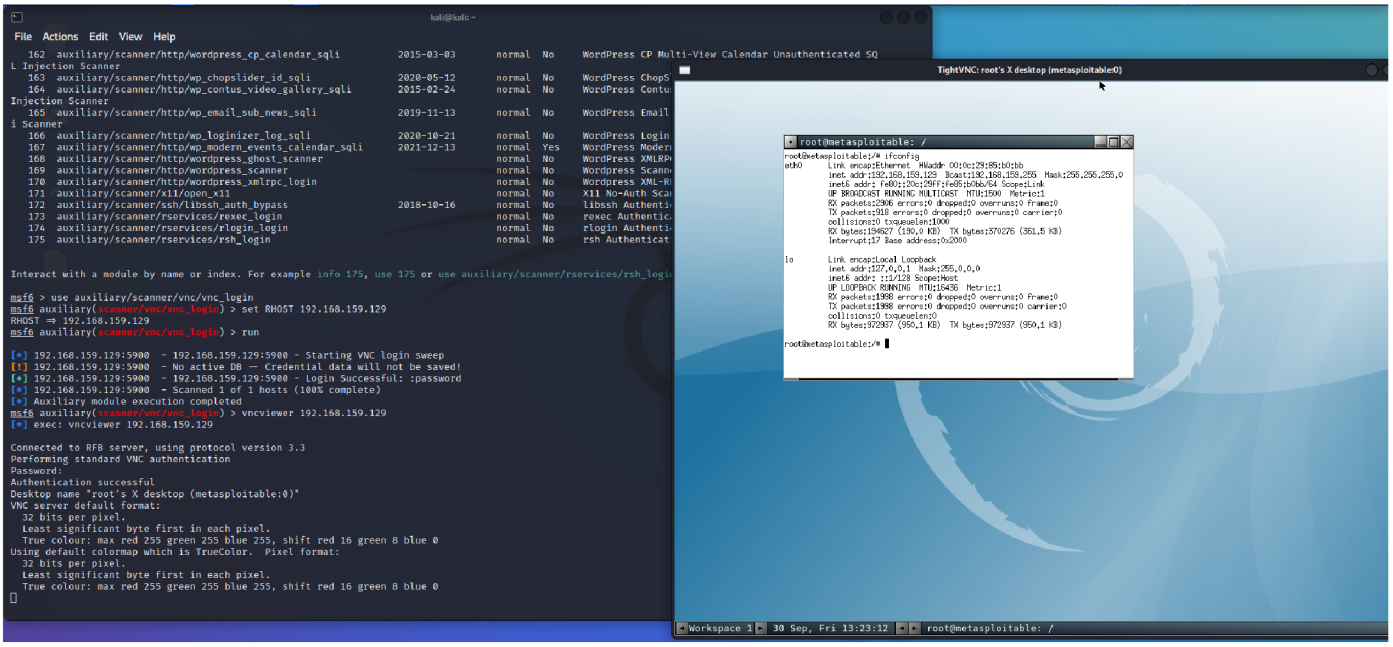
Je třeba nastavit IP adresu Metasploitable, návod na jejíž nalezení je popsán v úloze 9.

Poté je možné spustit skener s pomocí příkazu `run`. Ve výstupu je napsáno, že přihlášení do VNC bylo úspěšné s heslem `password`.

Poslední krok je připojení k VNC serveru pro vzdálené ovládání Metasploitable. Je třeba napsat příkaz:

```
vncviewer <IP_Address>
```

Kde IP adresa je zase IP adresa Metasploitable. Ve výstupu bude potřeba uvést heslo, což je dříve nalezené heslo `password`. Výsledek:



Obrázek 33 Výsledek VNC skeneru a úspěšné připojení do VNC Metasploitable. Zdroj: Autor

Vyhodnocení úkolu: S pomocí funkcí `vnc_login`, která byla nalezena příkazem `search` v Metasploit Framework, byla na Metasploitable úspěšně odhalena zranitelnost u VNC serveru, což umožnilo vzdálené ovládání tohoto operačního systému.

7 Závěr

Práce představila několik hlavních nástrojů v operačním systému Kali Linux pro analýzu (a využití) zranitelností. Na začátku byl představen samotný operační systém Kali Linux, jeho hlavní možnosti a návody na instalaci. Poté byl vymezen pojem zranitelnost, byly uvedené nejkritičtější zranitelnosti roku 2021 a způsoby ochrany proti nim. V teoretické části byly představeny vybrané nástroje analýzy zranitelností, jejich hlavní možnosti, k čemu slouží a jaké může odhalit či využit zranitelnosti. U některých nástrojů byly také uvedeny základní parametry.

V praktické části se demonstrovalo použití výše uvedených nástrojů. Praktická část byla rozdělena na 10 úloh, přičemž u každé úlohy byl stanoven určitý cíl, po jehož splnění by uživatel nástroje mohl odhalit či využit určitou zranitelnost na cílovém serveru. Krok po kroku byl představen návod na použití nástroje s uvedením potřebných příkazů, parametrů či prvků uživatelského rozhraní, které bylo potřeba využit pro splnění stanoveného cíle. V této části byl kladen důraz na demonstraci co nejjednodušších způsobů použití uvedených nástrojů k dosažení stanoveného cíle. Toto ukazuje velkou moc a výkon těchto nástrojů – díky nim by přečtením krátkých návodů dokázal téměř každý člověk odhalit kritické zranitelnosti na cílových systémech.

Na druhou stranu je kybernetická bezpečnost velmi složitá věda a není divné, že odborníci v této oblasti dostávají vysoké odměny za jejich práci. Existuje mnohem více typů zranitelností a útoků, než popsáno v této bakalářské práci a nové zranitelnosti se pořád vyskytují. Žádný počítačový systém není úplně v bezpečnosti a nikdy nebude – je to nekonečný závod ve zbrojení mezi zlomyslnými útočníky a analytiky kybernetické bezpečnosti.

Tyto skutečnosti ale neomlouvají nedodržování základních zásad kybernetické bezpečnosti nekvalifikovanými lidmi a běžnými uživateli. V dnešní době tyto zásady musí být dodržovány úplně všemi, protože dnes je každý uživatel počítače vystaven útoku.

Realita ale není tak chmurná, protože dodržování těchto základních zásad není příliš těžké pro zkušeného uživatele počítače a v případě jednotlivého uživatele to většinou stačí.

V případě společností, zejména těch větších, je doporučeno si najmout odborníka na kybernetickou bezpečnost, protože velké společnosti jsou ve mnohem větším riziku útoků.

8 Seznam použitých zdrojů

[1]: SAJINDRA, Hirushan, Case Study of Colonial Pipeline Ransomware Attack [online]. Researchgate, 2022. [cit. 09.10.2022] Dostupné z: https://www.researchgate.net/publication/361910184_Case_Study_of_Colonial_Pipeline_Ransomware_Attack

[2]: Anon. Glossary | CVE [online] [cit. 17.10.2022]. Dostupné z: https://www.cve.org/ResourcesSupport/Glossary

[3]: SHAH, Sugandh a B. M. MEHTRE, 2015. An overview of vulnerability assessment and penetration testing techniques. Journal of Computer Virology and Hacking Techniques [online] [cit. 19.10.2022]. 11(1), 27–49. ISSN 2263-8733. Dostupné z: doi:10.1007/s11416-014-0231-x

[4]: EDITOR, CSRC Content, [b.r.]. penetration testing - Glossary | CSRC [online] [cit. 24.10.2022]. Dostupné z: https://csrc.nist.gov/glossary/term/penetration_testing

[5]: Anon. Kali Linux | Penetration Testing and Ethical Hacking Linux Distribution. Kali Linux [online] [cit. 09.10.2022]. Dostupné z: https://www.kali.org/

[6]: WEIDMAN, Georgia. Penetration testing: a hands-on introduction to hacking. San Francisco, CA: No Starch Press, 2014. ISBN 978-1-59327-564-8.

[7]: Anon. Features. Kali Linux [online] [cit. 09.10.2022]. Dostupné z: https://www.kali.org/features/

[8]: Anon. What is Kali Linux? | Kali Linux Documentation. Kali Linux [online] [cit. 09.10.2022]. Dostupné z: https://www.kali.org/docs/introduction/what-is-kali-linux/

[9]: Anon. Kali Docs | Kali Linux Documentation. Kali Linux [online] [cit. 09.10.2022]. Dostupné z: https://www.kali.org/docs/

[10]: Anon. Kali Linux Forums [online] [cit. 09.10.2022]. Dostupné z: https://forums.kali.org/

[11]: Kali Linux, Penetration Testing and Ethical Hacking Linux Distribution [online]. OffSec Services Limited 2022. [cit. 02.07.2022]. Dostupné z: https://www.kali.org/

[12]: Rahalkar, S. A complete guide to Burp suite learn to detect application vulnerabilities. Apress, 2021. ISBN 978-1-4842-6402-7

[13]: Anon. Updating Kali | Kali Linux Documentation. Kali Linux [online] [cit. 02.10.2022]. Dostupné z: https://www.kali.org/docs/general-use/updating-kali/

[14]: KRSUL, Ivan, 1997. Computer Vulnerability Analysis: Thesis Proposal. Department of Computer Science Technical Reports [online] [cit. 17.10.2022]. Dostupné z: https://docs.lib.purdue.edu/cstech/1363

[15]: Anon. What is a Vulnerability? Definition + Examples | UpGuard [online] [cit. 17.10.2022]. Dostupné z: https://www.upguard.com/blog/vulnerability

[16]: KIZZA, Joseph Migga, 2009. Computer Network Vulnerabilities. In: Joseph Migga KIZZA, ed. A Guide to Computer Network Security [online]. London: Springer, The Computer Communications and Networks, s. 89–106 [online] [cit. 17.10.2022]. ISBN 978-1-84800-917-2. Dostupné z: doi:10.1007/978-1-84800-917-2_4

[17]: Anon. CVE – CVE [online] [cit. 17.10.2022]. Dostupné z: https://cve.mitre.org/

[18]: Anon. NVD – Home [online] [cit. 17.10.2022]. Dostupné z: https://nvd.nist.gov/

[19]: Anon. NVD – Categories [online] [cit. 17.10.2022]. Dostupné z: https://nvd.nist.gov/vuln/categories

[20]: Anon. CWE – Common Weakness Enumeration [online] [cit. 12.10.2022]. Dostupné z: https://cwe.mitre.org/

[21]: GORBENKO, Anatoliy, Alexander ROMANOVSKY, Olga TARASYUK a Oleksandr BILOBORODOV, 2017. Experience Report: Study of Vulnerabilities of Enterprise Operating Systems. In: 2017 IEEE 28th International Symposium on Software Reliability Engineering (ISSRE) [online] [cit. 17.10.2022]. s. 205–215. ISSN 2332-6549. Dostupné z: doi:10.1109/ISSRE.2017.20

[22]: ALGARNI, Abdullah, 2022. The Historical Relationship between the Software Vulnerability Lifecycle and Vulnerability Markets: Security and Economic Risks. Computers [online] [cit. 17.10.2022]. 11, 137. Dostupné z: doi:10.3390/computers11090137

[23]: JIANG, Yuning, 2022. PhD Thesis – Vulnerability Analysis for Critical Infrastructures. B.m. b.n. [online] [cit. 17.10.2022] Dostupné z: https://www.diva-portal.org/smash/record.jsf?pid=diva2%3A1680358&dswid=7826

[24]: VENTER, H. S., J. H. P. ELOFF a Y. L. LI, 2008. Standardising vulnerability categories. Computers & Security [online] [cit. 17.10.2022]. 27(3), 71–83. ISSN 0167-4048. Dostupné z: doi:10.1016/j.cose.2008.04.002

[25]: Anon. OWASP Top 10:2021 [online] [cit. 12.10.2022]. Dostupné z: https://owasp.org/Top10/

[26]: MANTRA, IGN, Muhammad Syarif HARTAWAN, Hoga SARAGIH a Aedah Abd RAHMAN, 2019. Web Vulnerability Assessment and Maturity Model Analysis on Indonesia Higher Education. Procedia Computer Science [online] [cit. 18.10.2022]. 161, The Fifth Information Systems International Conference, 23-24 July 2019, Surabaya, Indonesia, 1165–1172. ISSN 1877-0509. Dostupné z: doi:10.1016/j.procs.2019.11.229

[27]: JEON, Sanghoon a Huy Kang KIM, 2021. AutoVAS: An automated vulnerability analysis system with a deep learning approach. Computers & Security [online] [cit. 19.10.2022]. 106, 102308. ISSN 0167-4048. Dostupné z: doi:10.1016/j.cose.2021.102308

[28]: Anon. Guide to Vulnerability Analysis for Computer Networks and Systems [online] [cit. 19.10.2022]. Dostupné z: https://link.springer.com/book/10.1007/978-3-319-92624-7

[29]: Anon. Burp Suite – Application Security Testing Software [online] [cit. 11.10.2022]. Dostupné z: https://portswigger.net/burp

[30]: Anon. Burp Suite documentation [online] [cit. 02.10.2022]. Dostupné z: https://portswigger.net/burp/documentation

[31]: Anon. Nmap: the Network Mapper – Free Security Scanner [online] [cit. 11.10.2022]. Dostupné z: https://nmap.org/

[32]: Anon. sqlmap: automatic SQL injection and database takeover tool [online] [cit. 11.10.2022]. Dostupné z: https://sqlmap.org/

[33]: Anon. WPScan: WordPress Security [online] [cit. 12.10.2022]. Dostupné z: https://wpscan.com/

[34]: Anon. Metasploit | Penetration Testing Software, Pen Testing Security. Metasploit [online] [cit. 12.10.2022]. Dostupné z: https://www.metasploit.com/

[35]: Rahalkar, S. Metasploit 5.0 for beginners: Perform penetration testing to secure your it environment against threats and vulnerabilities. Packt Publishing, 2020. ISBN 978-1-83898-266-9

[36]: Anon. Cloud Application Platform | Heroku [online] [cit. 09.10.2022]. Dostupné z: https://www.heroku.com/

[37]: Bareño-Gutierrez, Raúl & López-Sevillano, Alexandra & Piraquive, Flor & Gonzalez Crespo, Ruben. Analysis of WEB browsers of HSTS security under a man attack in the MITM environment. Researchgate. 2020. [online] [cit. 26.07.2022] DOI: 10.1007/978-3-030-81635-3_27

[38]: R. Palacios, A. F. Fernández-Portillo, E. F. Sánchez-Úbeda and P. García-De-Zúñiga, "HTB: A Very Effective Method to Protect Web Servers Against BREACH Attack to HTTPS," in IEEE Access, vol. 10, pp. 40381-40390, 2022. [online] [cit. 26.07.2022] DOI: 10.1109/ACCESS.2022.3166175

[39]: Alsaffar, Mohammad, Saud Aljaloud, Badiea Al-Shaibani, Zeyad Al-Mekhlafi, Tariq Almurayziq, Gharbi Alshammari, and Abdullah Alshammari. "Detection of Web Cross-Site Scripting (XSS) Attacks." Electronics 11 (July 15, 2022): 2212. [online] [cit. 29.07.2022]. Dostupné z: DOI: 10.3390/electronics11142212

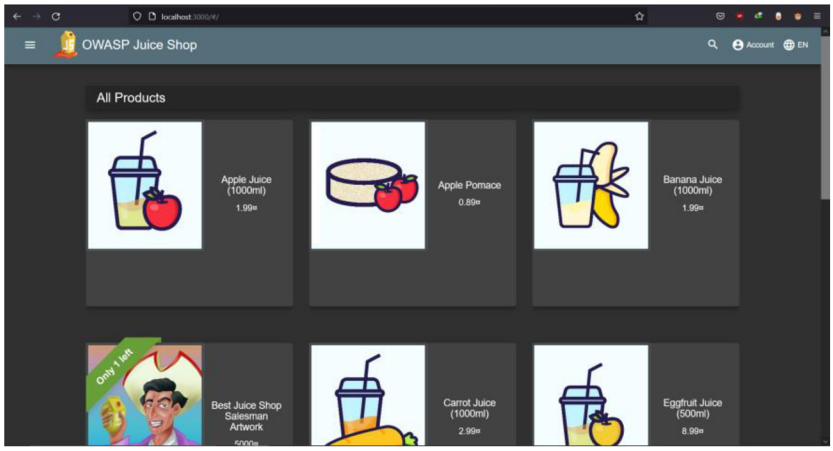
[40]: Kimminich, B. (n.d.). Cross site scripting (XSS). Cross Site Scripting (XSS) · Pwning OWASP Juice Shop. [online] [cit. 30.07.2022]. Dostupné z https://pwning.owasp-juice.shop/part2/xss.html

[41]: Installing the Metasploit framework. Installing the Metasploit Framework | Metasploit Documentation. (n.d.). [online] [cit. 24.09.2022]. Dostupné z: https://docs.rapid7.com/metasploit/installing-the-metasploit-framework/

9 Seznam použitých obrázků

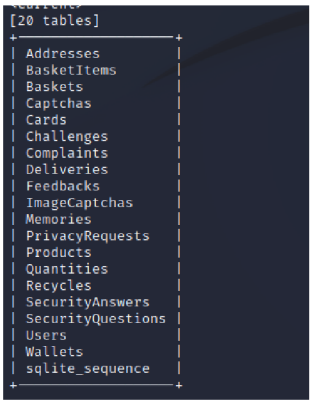
10 Přílohy

Příloha č.1 – webová stránka OWASP Juice Shop na vlastním serveru:

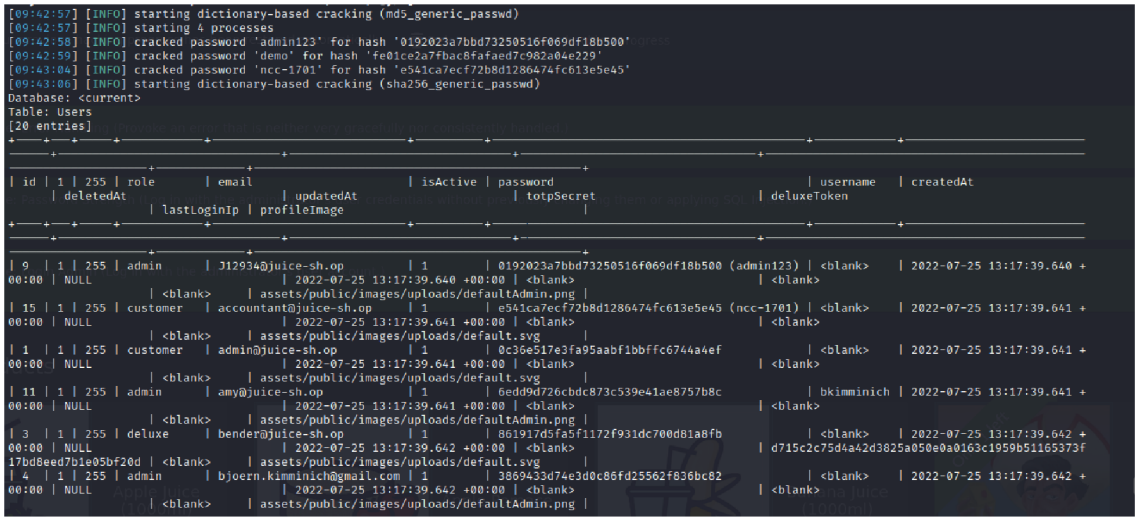


Obrázek 34 OWASP Juice Shop. Zdroj: Autor

Příloha č.2 – Enumerace DB pomocí SQLMAP



Obrázek 35 Získané názvy tabulek. Zdroj: Autor



Obrázek 36 Dump tabulky Users. Zdroj: Autor

Univerzita Hradec Králové
Fakulta informatiky a managementu

Zadání bakalářské práce

Autor:	**Vyacheslav Novak**
Studium:	I1900233
Studijní program:	B1802 Aplikovaná informatika
Studijní obor:	Aplikovaná informatika
Název bakalářské práce:	**Využití Kali Linux pro analýzu zranitelností**
Název bakalářské práce AJ:	Using Kali Linux for analyzing vulnerabilities

Cíl, metody, literatura, předpoklady:

Cílem práce je provést analýzu nástrojů v Kali Linux pro analýzu zranitelností a jejich praktické využití. V teoretické části autor práce analyzuje dostupné nástroje v Kali Linux využitelné pro analýzu zranitelností. V praktické části autor navrhne a realizuje praktické využití vybraných nástrojů v podobě deseti řešených úloh.

Zadávající pracoviště:	Katedra informačních technologií, Fakulta informatiky a managementu
Vedoucí práce:	Mgr. Josef Horálek, Ph.D.
Datum zadání závěrečné práce:	15.10.2021