UNIVERZITA PARDUBICE

FAKULTA ELEKTROTECHNIKY
A INFORMATIKY

BAKALÁŘSKÁ PRÁCE

2018 Tomáš Kodym

Univerzita Pardubice

Fakulta elektrotechniky a informatiky

Penetrační testování mobilů

Tomáš Kodym

Bakalářská práce

2018

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: **2016/2017**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Tomáš Kodym**
Osobní číslo:	**I14322**
Studijní program:	**B2646 Informační technologie**
Studijní obor:	**Informační technologie**
Název tématu:	**Penetrační testování mobilů**
Zadávající katedra:	**Katedra informačních technologií**

Zásady pro vypracování:

Cílem práce je představit a analyzovat problematiku bezpečnosti moderních OS v mobilních telefonech platformy Android. Teoretická část představí nástroje Metasploit a MSFvenom, přičemž bude zanalyzováno, jak lze tyto nástroje využít pro napadání mobilních telefonů. Na případové studii bude útok na mobilní telefon předveden a rovněž bude provedena analýza, jak se lze proti útokům tohoto typu bránit.

Rozsah grafických prací:

Rozsah pracovní zprávy: **35**

Forma zpracování bakalářské práce: **tištěná**

Seznam odborné literatury:

ENGEBRETSON, Pat a James BROAD. The basics of hacking and penetration testing: ethical hacking and penetration testing made easy. Waltham, MA: Syngress, c2011. ISBN 1597496553.

MURPHY, Mark L. - LINDA, Bohdan. Android 2: průvodce programováním mobilních aplikací. Vyd. 1. Brno: Computer Press, 2011, 375 s. Technology in action series. ISBN 978-80-251-3194-7.

Vedoucí bakalářské práce: **Ing. Soňa Neradová, Ph.D.**
Katedra informačních technologií

Datum zadání bakalářské práce: **31. října 2016**

Termín odevzdání bakalářské práce: **12. května 2017**

Univerzita Pardubice Česká republika

L.S.

Ing. Zdeněk Němec, Ph.D.
děkan

Mgr. Josef Horálek, Ph.D.
vedoucí katedry

V Pardubicích dne 31. března 2017

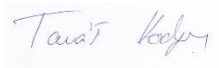
Prohlášení autora

Prohlašuji, že jsem tuto práci vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Souhlasím s prezenčním zpřístupněním své práce v Univerzitní knihovně.

V Pardubicích dne 29. 09. 2017



Tomáš Kodym

PODĚKOVÁNÍ

Zde bych chtěl poděkovat paní Ing. Soně Neradové Ph.D. za strávený čas a cenné rady při zpracování bakalářské práce. Dále bych chtěl poděkovat celé své rodině a přítelkyni za poskytnutou oporu během celého studia.

ANOTACE

Tato práce se zabývá problematikou bezpečnosti moderních operačních systémů v mobilních telefonech platformy Android pomoc nástrojů Metasploit a MSFvenom, které lze využít pro napadání mobilních telefonů. Záměrem práce je předvedení útoku na mobilní telefon s platformou Android a získání citlivých dat v různých případech s využitím a bez využití antivirového software a přímým vložením škodlivého kódu do již existující aplikace.

KLÍČOVÁ SLOVA

Penetrační testování, Metasploit, MSFvenom, Android

TITLE

Penetration testing of mobile phones

ANNOTATION

This bachelor thesis deals with the security issues of modern operating systems in Android mobile phones by using the Metasploit and MSFvenom tools that can be used for attacking mobile phones. The aim of the thesis is to demonstrate the attack on an Android-based mobile phone and to obtain sensitive data in various cases by using and without the usage of antivirus software and direct insertion of malicious code into an existing application.

KEYWORDS

Penetration testing, Metasploit, MSFVenom, Android

OBSAH

SEZNAM ILUSTRACÍ A TABULEK

SEZNAM ZKRATEK A ZNAČEK

AES	Advanced Encryption Standard
APT	The Advanced Package Tool
DLL	Dynamic Link Library
DSA	Digital Signature Algorithm
GPS	Global Positioning Systém
HTML	Hypertext Markup Language
LTS	Long Term Support
MSF	Metasploit Framework
NFC	Near Field Communication
OHA	Open Handset Alliance
OS	Operační systém
RAM	Random Access Memory
RSA	Rivest Shamir Adleman
SHA	Secure Hash Algorithm
SIP	Session Initiation Protocol
SQL	Structured Query Language
SU	Super User
USB	Universal Serial Bus
VPN	Virtual Private Network

ÚVOD

Vzhledem k enormnímu rozvoji moderní technologie, kdy mobilní zařízení jsou součástí každodenního života většiny z nás, je důležité se zamyslet nad tím, jaké jsou nástrahy a rizika při jejich používání.

Tato práce představuje operační systém Android, včetně jeho verzování, kdy u každé následující verze jsou vylíčeny výhody oproti verzi předchozí. Popisuje architekturu OS Android, která je složena z 6 částí, jež mezi sebou komunikují a spolupracují, tudíž nejsou úplně odděleny, ačkoliv každá vrstva má svůj úkol a vystupuje samostatně.

V další části jsou rozebrány možnosti zabezpečení, které jsou poskytovány od společnosti Google nebo dalšími eventualitami zabezpečení včetně možností, které jádro Linuxu poskytuje.

Následně je rozebrán framework Metasploit, včetně jeho architektury, vysvětleny základní skupiny příkazů k ovládání a základní pojmy či vysvětleno navázání spojení mezi útočníkem a obětí prostřednictvím Meterpreteru. Práce se také zaměřuje na představení a rozebrání základních příkazů programu MSFvenom, který slouží k vytváření škodlivých aplikací. Dále je předvedena instalace frameworku Metasploit na Linuxovou distribuci Ubuntu.

Následuje zobrazení útoku na mobilní telefon a získání citlivých dat ve třech odlišných scénářích. V prvním scénáři je předvedeno vytvoření škodlivé aplikace a její distribuce k uživateli do nezabezpečeného zařízení, možnosti útoku útočníka a získání citlivých dat oběti. Druhý scénář je zaměřen na útok na zařízení, které je zabezpečeno antivirovým programem. Ve třetím scénáři je provedeno nahrání škodlivého kódu do již existující aplikace.

Záměrem práce je zjištění, jaké jsou možnosti odhalení škodlivého kódu a jak se lze proti útokům na mobilní telefon bránit.

1 TEORETICKÁ ČÁST

1.1 Operační systém Android

Operační systém Android, vytvořený společností Google, Inc., patří mezi nejrozšířenější operační systémy pro mobilní zařízení na světě. Jedná se o Open source platformu, která zajišťuje dostupnost, rozšiřitelnost a svobodu šíření kódu a při splnění určitých podmínek i šíření licence. Android je založen na jádře operačního systému Linux a je určen primárně pro chytré mobilní telefony, tablety, chytré televize a další elektronické doplňky, např.: chytré hodinky, automobily a herní konzole. (Ujbányai, 2012)

Historie OS Android začíná v roce 2003 v Kalifornii ve městě Palo Alto, kde byla založena firma Android Inc., která se specializovala na vývoj aplikací pro mobilní zařízení. Zakladateli byli: Andy Rubin, Rich Miner, Nick Sears a Chris White. V roce 2005 byla společnost odkoupena společností Google, Inc., a firma Android se stala její dceřinou společností. Hlavním důvodem tohoto kroku, byla snaha proniknout na trh mobilních telefonů. V roce 2007 bylo založeno sdružení firem Open Headset Alliance, jehož úkol spočívá ve vývoji operačního systému Android. V současné době skupina obsahuje 84 společností. Společnosti, které do uskupení patří, jsou například: T-Mobile, Vodafone, Dell, Samsung Electronics a Intel Inc. (Ujbányai, 2012)

Od vydání první verze v září 2008, prošel Android mnoha aktualizacemi a opravami předešlých verzí. (Ujbányai, 2012)Mimo prvních dvou verzí je každá další verze označena speciálním označením, které symbolizuje název pochutiny například: Cupcake, Donut, Oreo, Lollipop. Poslední vydaná verze je Android 8.0 s označením Oreo.

1.2 Verze OS Android

Cílem této podkapitoly je seznámit čtenáře s nejdůležitějšími verzemi OS Android a stručně popsat změny, výhody a nevýhody oproti verzi předchozí.

Android 1.0

První verze byla vydána 23. září 2008, byla založena na jádru OS Linux 2.6.25 a obsahovala pouze základní programy. Jediné zařízení, na kterém OS běžel, bylo HTC Dream.

Android 1.0 obsahoval:

- Android Market – repositář, který umožňoval stahovat a instalovat další aplikace,
- plnohodnotný webový prohlížeč,
- obyčejný fotoaparát bez jakýkoliv možností,
- podpora Wi-Fi a Bluetooth,
- aplikace od společnosti Google – Gmail, Google kontakty, Google kalendář a Google mapy, které měly uživateli usnadnit využívání zařízení. (Ujbányai, 2012)

Android 1.1

Druhá verze byla oficiálně vydána 9. února 2009, pouze několik měsíců po vydání první verze. Podstatou jejího vypuštění bylo řešení chyb a nedostatků verze 1.0. Změny se týkaly pouze zařízení HTC Dream. (Kilián, 2015; Ujbányai, 2012)

Nově zde bylo možné:

- ukládat přílohy SMS a MMS zpráv,
- zobrazení a skrytí číselníku.

Opravenými chybami předchozí verze bylo:

- odstranění položek nastavení, které nebyly otestovány nebo nefungovaly správně,
- zlepšení hlášení o chybách připojení, která umožňují snadnější hledání a vyřešení chyby. (Kilián, 2015; Ujbányai, 2012)

Android 1.5 Cupcake

V pořadí třetí verze byla vydána 27. dubna 2009 a je první verzí, která dostala název dle pochutiny. Změněna byla nejen verze jádra, která se změnila na verzi 2.6.27, ale byly přidány nové funkce:

- rychlejší získání polohy pomocí GPS,
- softwarová klávesnice na obrazovce,
- zrychlení spouštění fotoaparátu a možnost nahrávání videa,
- podpora kopírování a vkládání textu,

- zpřesnění a výrazné zrychlení uživatelského rozhraní. (Kilián, 2015; Ujbányai, 2012)

Vzhledem k přechodu na novou verzi jádra se operační systém jevil mnohem robustněji a stabilněji než předchozí verze. Největší přínos byl zaznamenán v případě softwarové klávesnice, která vytlačila hardwarovou klávesnici. (Kilián, 2015; Ujbányai, 2012)

Android 1.6 Donut

Tato verze byla vydána 15. září 2009 z důvodu nové verze jádra 2.6.29. Systém byl doplněn o další funkčnosti:

- podpora pro rozdílné rozlišení obrazovky,
- výkonnější repositáře Android Market,
- vylepšení vyhledávání hlasem,
- přidáno API pro práci s gesty,
- přidán nový ovládací panel pro práci s VPN,
- indikátor využití baterie,
- automatické zálohy. (Kilián, 2015; Ujbányai, 2012)

Mezi největší přínos této verze patří podpora rozdílného rozlišení obrazovky, které umožňuje běh OS na zařízeních, která podporují různé rozlišení obrazovek. (Kilián, 2015)

Android 2.0/2.1 Eclair

Nastupující verze je vydána 26. října 2009, postavena na stejném jádře jako verze 1.6 a přináší s sebou mnoho nových výhod:

- podpora Bluetooth verze 2.1,
- podpora více účtů v kontaktech,
- vylepšený webový prohlížeč,
- podpora HTML5,
- podpora MS Exchange,
- vylepšení softwarové klávesnice,
- synchronizace kontaktů.

Tato verze představovala jednu z nejzákladnějších změn od první verze. Změna byla podstatná, jak po vizuální stránce, tak po stránce funkcionální. Mnoho malých výhod a vylepšení hrálo podstatnou roli v oblíbenosti zařízení. (Kilián, 2015; Ujbányai, 2012)

Android 2.2 Froyo

Následující verze, která byla založená na Linuxovém jádru 2.6.32, byla představena 20. 5. 2010, spolu s ní bylo představeno mnoho nových funkcí a vylepšení například:

- možnost instalace aplikací na paměťovou kartu,
- zrychlení výkonu pomocí Just In Time kompilaci, která vede k rychlejšímu spouštění kódu,
- podpora USB a Wi-Fi tethering,
- vylepšení správy paměti RAM,
- možnost přidání hesla k zamčení zařízení,
- zvýšení výkonu prohlížeče.

Úkolem nové verze byla snaha o výrazné zvýšení rychlosti a o zjednodušení práce s OS. (Kilián, 2015; Ujbányai, 2012)

Android 2.3/2.4 Gingerbread

Nová verze založená na Linuxovém jádru verze 2.6.35 byla vydána 6. 12. 2010 a přinesla s sebou několik zajímavých změn:

- vylepšení funkce kopírovat a vložit,
- vylepšená správa energie,
- podpora NFC,
- změna podporovaného souborového systému,
- aktualizace uživatelského prostředí,
- podpora internetové telefonie pomocí protokolu SIP.

Většina všech provedených změn se týkala proměny uživatelského prostředí, kde převládala snaha o zvýšení výkonu OS a snížení spotřeby baterie. (Kilián, 2015; Ujbányai, 2012)

Android 3.0/3.1/3.2 Honeycomb

Android 3.0 byl navržen pro zařízení s větším rozlišením obrazovky, převážně pro tablety. I přes nízkou oblíbenost přináší několik nových funkcí:

- nejsou využívána fyzická tlačítka, ale pouze softwarová tlačítka na displeji,
- přepracování klávesnice pro pohodlnější využívání,
- možnost šifrování uživatelských dat,
- podpora příslušenství připojeného přes USB. (Kilián, 2015; Ujbányai, 2012)

Android 4.0/4.0.1/4.0.2 Ice Cream Sandwich

Verze Android 4.0 se na trhu objevila 19. října 2011, byla založena na Linuxovém jádře verze 3.0.1. Po pokusu předchozí verze, která byla dostupná pouze pro tablety, dochází ke změně a je předvedena platforma, která bude vhodná pro všechny typy zařízení. Taková změna s sebou přináší mnoho nových funkcí a vymožeností:

- funkce Android Beam – slouží pro rychlejší výměnu dat mezi dvěma zařízeními pomocí NFC,
- vylepšené rozpoznávání hlasu,
- rozpoznávání tváře,
- kontrola využívání mobilních dat,
- podpora pro Wi-Fi direct – slouží k propojení dvou zařízení přes Wi-Fi,
- podpora pro nahrávání videa v rozlišení 1080 p.

Podstatnou novinkou u verze 4.0 byla markantnější snaha o zabezpečení, bylo přidáno šifrování dat, které zamezilo přístupu, dokud se do zařízení nepřihlásil právoplatný uživatel. (Kilián, 2015; Ujbányai, 2012)

Android 4.1/4.2/4.3 Jelly Bean

Nastupující platformy 4.1 až 4.3 byly výrazně zaměřeny na vylepšení designu a vylepšení uživatelského rozhraní. Podstatnou součástí platformy byl projekt Butter, jehož cílem bylo minimalizovat prodlevy systému a zrychlit jeho odezvu. I když se většina vylepšení týkala převážně designu i v této verzi bylo přidáno několik nových funkcí:

- šifrování placených aplikací stažených z Google Play,
- podpora vícekanálového zvuku,

- podpora technologie Miracast – bezdrátový přenos obrazu přes Wi-Fi do jiného zařízení např.: televizor, projektor,
- podpora pro Bluetooth Smart technology – umožňuje snížit spotřebu energie při přenosu. („Jelly Bean", 2015; Kilián, 2015)

Android 4.4 Kitkat

Android 4.4 je navrhnut pro rychlý běh na mnohem větší škále zařízení než všechny předchozí platformy. Předností byla optimalizace pro mnohem méně výkonná zařízená, kterým postačila pro spuštění paměť RAM pouze 512 MB. Založen byla na Linuxovém jádře verze 3.4.0. Nejzákladnější změny se týkaly opět vizuální stránky systému:

- vylepšené uživatelské rozhraní,
- kompatibilita se systémem Android Wear – tento systém se objevuje v nositelné elektronice a umožňuje snadné připojení,
- nově byla přidána podpora bezdrátového tisku,
- senzor počítání kroků,
- možnost nahrávání videozáznamu z obrazovky,
- vylepšení kryptografických algoritmů a přidání několika dalších,
- podpora pro digitální podpisy. („Android KitKat", 2015; Kilián, 2015)

Android 5.0/5.1 Lollipop

Následují platforma byla založena na Linuxovém jádře 3.4.0 a byla vydána 12. listopadu 2014. Podstatnou součástí jsou opět velké vizuální změny uživatelského rozhraní, navigační lišty a ikon. Mimo to proběhlo několik zajímavých změn:

- lepší výdrž baterie,
- snížení odezvy a zvýšení výkonu systému,
- nové šifrování uživatelských dat, které umožní chránit data při ztracených nebo odcizených zařízení – nepomůže ani resetování telefonu do továrního nastavení,
- vymáhání funkce SE Linux pro všechny aplikace,
- nově podpora pro Duální sim karty. („Android Lollipop", 2015; Kilián, 2015)

Android 6.0 Marshmallow

Verze 6.0 byla vydána 5. října 2015 a přináší s sebou mnoho nových možností a funkcí. V této verzi se nekonaly velké vizuální změny. Pravděpodobně největší změnou prošla oprávnění aplikací a úspora baterie.

- nový systém oprávnění – Aplikace nemusí využívat všechna dostupná práva, ale pouze ta, která jsou uživatelem povolena,
- podpora sensoru otisku prstů,
- propracovaná správa energie – aplikace Doze a APP Standby,
- podpora USB-C,
- přesnější a úspornější využívání paměti,
- mobilní platby. („Android 6.0 Marshmallow", 2015; Kovaczicz, 2015)

Android 7.0/7.1 Nougat

Předposlední verze byla vydána 22. srpna 2016. Tato verze s sebou přinesla mnoho příjemných aktualizací a uživateli nových funkcí:

- možnost spustit dvě okna vedle sebe,
- rychlé přepínání posledních dvou aplikací,
- změny dlaždic v rychlém nastavení,
- změna velikosti položek na obrazovce,
- vylepšená správa energie,
- podpora virtuální reality,
- nový Android JIT Compiler – až o 75 % rychlejší instalace aplikací,
- spořič mobilních dat,
- zlepšení dotykové odezvy displeje. („Android 7.0 for Developers", 2015; „Android 7.0 Nougat", 2015; Macho, 2016)

Android 8.0 Oreo

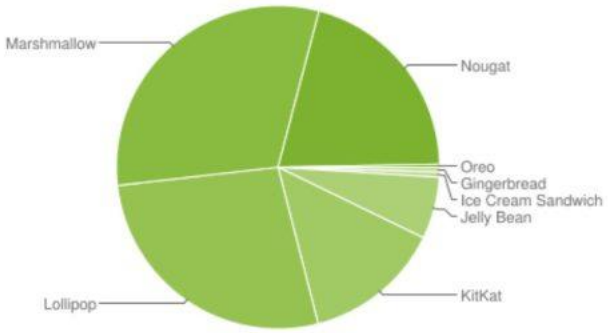
Nejnovější verze se zaměřuje převážně výdrž baterie, bezpečnost a optimalizaci. Byla vypuštěna 21. srpna 2017 a prozatím se řadí mezi nejméně rozšířené verze. Poslední verze s sebou přináší pro uživatele i pro vývojáře mnoho zajímavostí:

- vylepšená správa energie pro aplikace v pozadí,

- přepracování notifikací,
- přidání funkce obraz v obraze,
- neviditelné aktualizace – stahování a instalace aktualizací se provádí na pozadí. („Android Oreo”, 2015; Srb, 2017; Trlica, 2017a)

Na obrázku č. 1 je zobrazeno aktuální rozdělení nejoblíbenějších verzí.

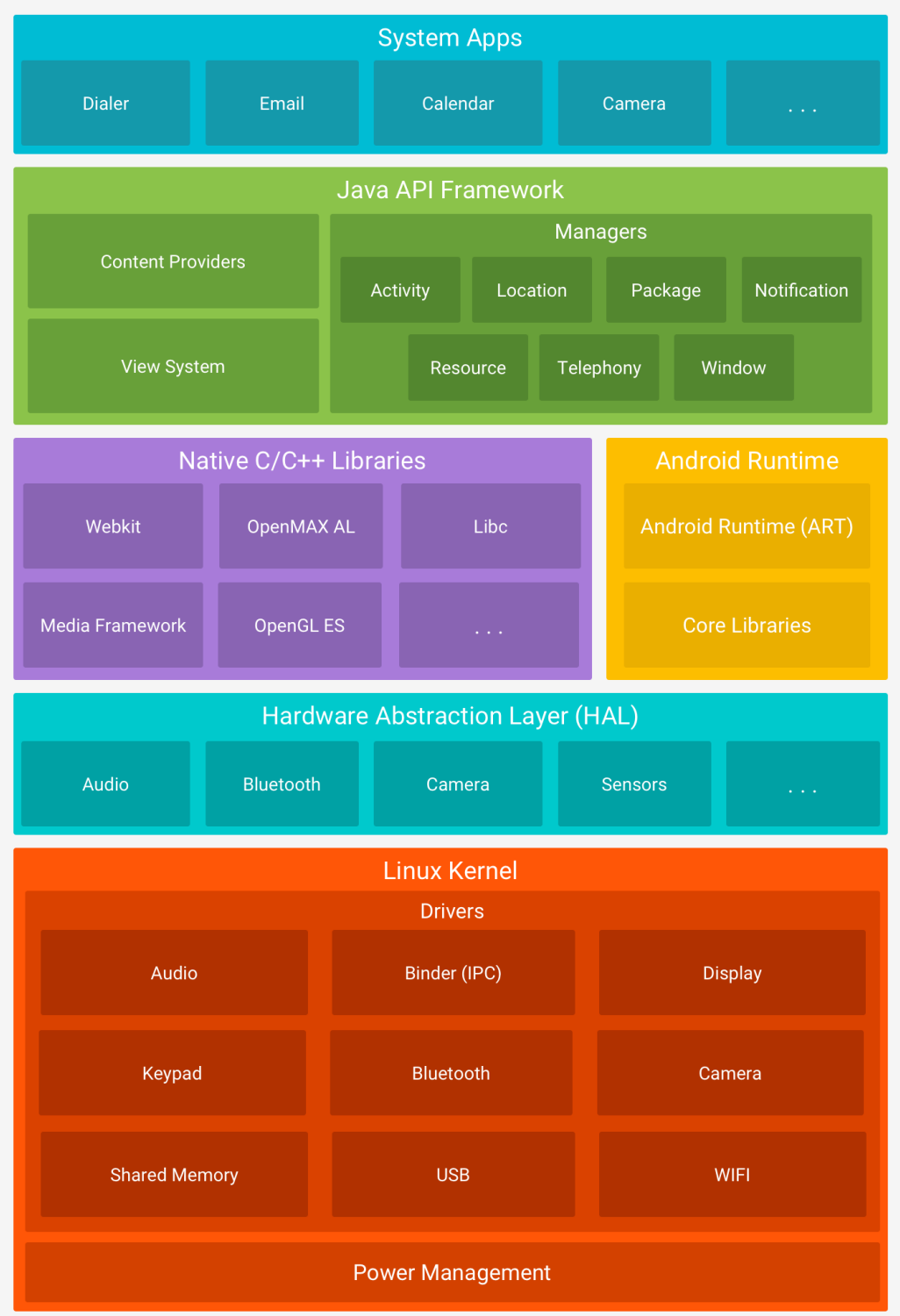
Version	Codename	API	Distribution
2.3.3 - 2.3.7	Gingerbread	10	0.5%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	0.5%
4.1.x	Jelly Bean	16	2.2%
4.2.x		17	3.1%
4.3		18	0.9%
4.4	KitKat	19	13.8%
5.0	Lollipop	21	6.4%
5.1		22	20.8%
6.0	Marshmallow	23	30.9%
7.0	Nougat	24	17.6%
7.1		25	3.0%
8.0	Oreo	26	0.3%



Obrázek 1 Podíl verzí na trhu (Trlica, 2017a)

1.3 Architektura OS Android

Architektura OS Android je složena z 6 částí, jak je zřejmé z obrázku 2. Z obrázku vyplývá, že každá vrstva má svůj vlastní úkol a vystupuje samostatně, ale při praxi dochází ke komunikaci a spolupráci mezi jednotlivými vrstvami, tudíž mezi sebou nejsou vrstvy úplně odděleny. (Ujbányai, 2012)



Obrázek 2 Vrstvy OS Android (Platform Architecture, 2015)

1.3.1 Linux Kernel

Kernel je základem operačního systému. Jeho hlavním úkolem je zajišťování komunikace mezi hardwarovými a softwarovými komponentami. Pro tuto komunikaci jsou využívány ovladače. Při startu zařízení je jádro zavedeno do operační paměti a postupně je mu předáno

řízení a kontrola nad OS například: Správa procesů, podpora správy paměti, správa sítí a správa napájení a využívání klíčových stupňů zabezpečení. Jedním z důvodů použití Linuxového jádra je možnost využití přenositelnosti mezi mnoha zařízeními. (Ujbányai, 2012)

1.3.2 Hardware Abstraction Layer (HAL)

Android dává možnost implementovat vlastní specifikace pro zařízení a vlastní ovladače. Modul HAL definuje standartní rozhraní pro hardwarové komponenty. Vrstva HAL se skládá z knihovních modulů, z nichž každý modul je určen pro konkrétní hardwarovou komponentu. V momentě, kdy dojde k přijmutí požadavku k přístupu pomocí API Framework, systém Android načte požadovanou knihovnu pro hardwarovou komponentu. Tímto způsobem je možné vytvořit modul bez ovlivnění a úpravy systému vyšší úrovně. (Platform Architecture, 2015)

1.3.3 Native Libraries

Základní knihovny OS Android jsou napsány v programovacím jazyce C/C++ a jedná se o základní funkce systému. Funkce jsou vývojářům poskytnuty pomocí Android Aplication Framework. Tato kategorie obsahuje například:

- Audio Manager – knihovna sloužící pro práci se zvuky a vyzváněním
- Free Type – knihovna pro bitmapové a vektorové vykreslování písma
- Libc – standardní C knihovna uzpůsobená pro práci na vestavěném zařízení
- Media Framework – knihovna pro práci s mediálními soubory
- OpenGL – knihovna sloužící pro práci s 3D grafikou
- SQLite – knihovna obsahující relační databázi, která je dostupná všem aplikacím
- Surface Manager – knihovna spravující přístup k displeji
- Web Kit – knihovna poskytující nástroje pro prohlížení internetu
- SSL – knihovna umožňující využívání šifrované komunikace („Platform Architecture”, 2015; Ujbányai, 2012)

1.3.4 Android Runtime

Tato vrstva slouží primárně pro běh aplikací. Aplikace nejsou naprogramovány v nativním kódu, ale v programovacím jazyce Java. Do verze Androidu 5.0 je využíván Dalvik Virtual Machine, jejímž úkolem je převod kódu aplikací do Java byte kódu. DVM pro svůj běh využívá

základní vlastnosti Linuxového jádra, například: správa procesů, správa paměti nebo podpora pro práci s vlákny. Hlavním důvodem využití DVM je snaha optimalizace virtuálního stroje pro potřeby mobilního zařízení. Podstatnou roli zde hraje pokus o největší výkon se zachováním nejmenší spotřeby energie. Důvod, proč byla zvolena DVM místo Java Virtual Machine je především licenční. Java VM a jeho součásti nejsou Open source. Každá aplikace běží ve svém vlastním procesu s vlastní instancí virtuálního stroje. DVM je navržen tak, že jej možné spouštět několik virtuálních strojů současně a přesto efektivně. Od verze 5.0 je využíván ART, který funguje na stejném principu jako DVM, ale využívá mnoho nových vlastností například: lepší podpora pro ladění, specializovaný odběr vzorků, předběžná kompilace a kompilace v reálném čase a hlášení o selhání. („Platform Architecture", 2015; Ujbányai, 2012)

Android Core obsahuje sadu základních knihoven, které poskytují většinu funkcí programovacího jazyka Java.

1.3.5 Java API Framework

Mnoho funkcí je v OS Android dostupných prostřednictvím API napsaném v programovacím jazyku Java. Lze si je představit jako stavební bloky (viz níže), pomocí kterých je možné vytvářet aplikace.

- View manager umožňuje využívat prvky grafického rozhraní, používat hardware zařízení, pracovat s obsahem jiných aplikací, využívat fotoaparát, spouštět aplikace na pozadí.
- Resource manager poskytuje přístup k nekódovým částím jako řetězce, grafika, přidané soubory.
- Notification manager umožňuje všem aplikacím zobrazovat upozornění ve stavovém řádku.
- Activity manager slouží ke správě životního cyklu aplikace například start, průběh a konec.
- Content provider umožňuje aplikacím přistupovat a pracovat s daty z jiných aplikací. (Platform Architecture, 2015)

1.3.6 System Apps

Nejvyšší vrstva obsahuje samotné aplikace, které jsou využívány uživateli. Může se jednat o předem nainstalované aplikace nebo aplikace, které si uživatel sám nainstaloval. (Platform Architecture, 2015)

1.4 Zabezpečení OS Android

Open Source architektura OS Android vyžaduje silnou bezpečnostní architekturu a přísné bezpečnostní programy. Využívá se zde vícestránkového zabezpečení, které svou flexibilitou dokáže ochránit všechny uživatele dané platformy. V každé části vývojového životního cyklu je podroben přísnému bezpečnostnímu testování. (Security, 2015)

1.4.1 Bezpečnost distribuovaná společností Google

Google nabízí uživatelům možnost využívání cloudových bezpečnostních služeb, které jsou k dispozici pro zařízení s OS Android. Mezi nejdůležitější se řadí například:

- Google Play – Jedná se o službu, která umožňuje uživatelům získávat, nakupovat a instalovat aplikace pro jejich zařízení. Sem nahrané aplikace projdou kontrolou licence a skenováním, zda neobsahují škodlivý software. Nespornou výhodou této služby, kterou poskytuje pro vývojáře je možnost oslovit potencionální zákazníky svou aplikací.
- Android Aktualizace – Aktualizace slouží pro získání nových dostupných upgradů OS a vylepšení zabezpečení.
- Ověřování aplikací – Slouží pro zneškodnění škodlivých aplikací. Automaticky skenuje stahovanou aplikaci z Google Play a pokud je závadná upozorní uživatele nebo aplikaci ze zařízení odstraní.
- Služby aplikací – Umožňuje aplikacím zálohovat svá data pomocí cloud technologií.
- SafetyNet – Jedná se o systém, který pomáhá sledovat a likvidovat známé bezpečnostní hrozby.
- Android Device Manager – Webová nebo mobilní aplikace, která slouží k detekci a zjištění umístění zcizeného nebo ztraceného zařízení. (Security, 2015)

1.4.2 Zabezpečení systému a jádra

Využití Linuxového jádra umožňuje používání pokročilých funkcí zabezpečení. Dovoluje vývojářům vytvořit ovladače pro známé jádro. Linuxové jádro poskytuje platformě Android bezpečnou komunikaci mezi procesy, která umožňuje komunikaci mezi aplikacemi běžící v různých procesech. (System and kernel security, 2015)

Výhody poskytovány Linuxovým jádrem jsou:

- Model založený na oprávnění uživatelů
- Izolaci procesů
- Rozšiřitelný mechanismus pro bezpečnou komunikaci mezi procesy
- Schopnost odstranit nepotřebné nebo bezpečnostně nestabilní části jádra (System and kernel security, 2015)

Vzhledem k víceuživatelskému využití Linuxu je důležité, aby jeden uživatel nemohl ovlivňovat data druhého uživatele a vzájemně si nezasahovali do svých procesů. OS musí zajistit aby:

- Uživatel A nemohl číst soubory uživatele B
- Uživatel A nemohl vyčerpat paměť uživatele B
- Uživatel A nevyčerpal zdroje procesoru
- Uživatel A nevyčerpal zařízení uživatele B (System and kernel security, 2015)

1.4.3 Systémový oddíl a nouzový režim

Systémový oddíl, který je v základu nastaven na režim čtení a není možné do něj zapisovat, obsahuje jádro systému Android, knihovny OS, aplikační framework a aplikace. Při spuštění OS v nouzovém režimu jsou k dispozici pouze předinstalované aplikace výrobcem. Aplikace třetích stran nejsou v základu spuštěny, tímto stylem je možné vyloučit problém, který způsobuje jedna z aplikací. Následně je možné vyloučit i nebezpečnou aplikaci neboli malware. Nouzový režim je využívám při neočekáváné havárii systému, a může být mocným pomocníkem. (System and kernel security, 2015)

1.4.4 Oprávnění souborového systému

Podstatou OS založených na Unixové distribuci je zajištění integrity souborů mezi uživateli. Tzn. uživatel A nemá oprávnění měnit soubory uživatele B. Toto omezení nesmí být absolutní, nýbrž selektivní. Uživatel nebo vývojář musí mít možnost své soubory sdílet. (System and kernel security, 2015)

1.4.5 Ověřené zavádění

Verze 6.0 a vyšší OS Android, podporují transparentní integritu blokových zařízení a testují přítomnost rootkitů[1]. K tomuto účelu je využívána funkce dm-verity, která prochází vrstvy úložiště systému souborů a ověřuje, zda odpovídá očekávané konfiguraci. Tato funkce zajišťuje uživateli stejný stav jako při posledním spuštěním OS. (System and kernel security, 2015)

1.4.6 Kryptografie

OS Android poskytuje sadu kryptografických rozhraní, které je možné aplikačně využívat. Nabízí se možnost použití standartních protokolů jako například: AES, RSA, DSA a SHA, nebo protokolů vyšší úrovně jako: SSL, HTTPS. (System and kernel security, 2015)

1.4.7 Šifrování souborového systému

Od verze 3.0 OS Android poskytuje šifrování celého souborového systému, které zajišťuje šifrování všech uživatelských dat. (System and kernel security, 2015)

Od verze 5.0 Android podporuje šifrování celého oddílu disku. Ověření probíhá pomocí uživatelského hesla. Po zavedení je nutné, aby se uživatel přihlásil, než bude moci využívat svou část disku. Takto nastavené zabezpečení však skýtá i nevýhody. Při restartování zařízení dojde k znepřístupnění dat do okamžiku, než dojde k opětovné autorizaci uživatele, to vede ke znepřístupnění některých základních služeb například: nefunkčnost budíku, nemožnost přijímat hovory. (System and kernel security, 2015)

Od verze 7.0 Android podporuje šifrování na úrovni souborů. Pomocí různých hesel může být zašifrováno několik souborů. (System and kernel security, 2015)

[1] Jedná se o sadu programů, pomocí kterých je možné maskovat přítomnost škodlivého kódu.

2 PRAKTICKÁ ČÁST

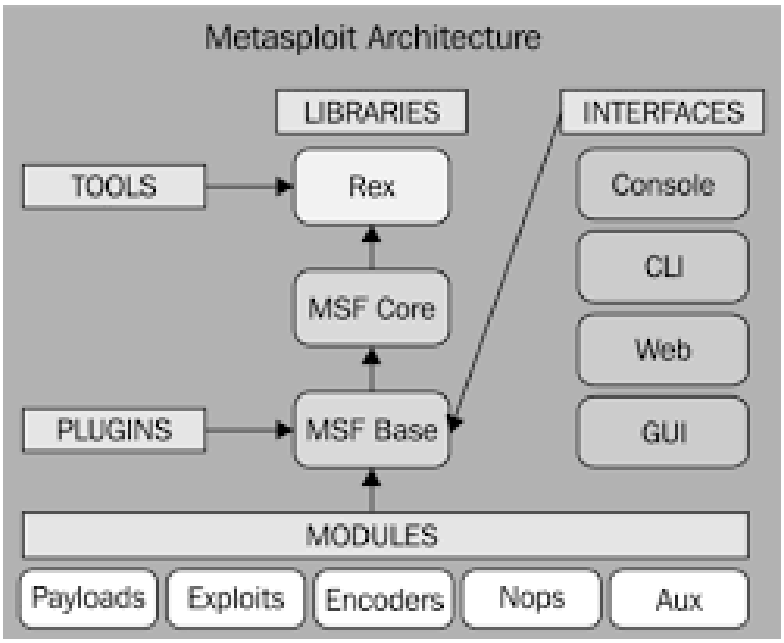
V první části práce jsou podrobně popsány použité nástroje, postupně vysvětleny základní pojmy, požadavky a jednotlivé kroky instalace představených nástrojů. Náplní druhé části práce je předvedení útoku na mobilní telefon s platformou Android a získání citlivých dat.

2.1 Metasploit Framework

Metasploit Framework je vytvářen jako OpenSource projekt od společnosti Rapid7. Slouží k vytváření a provádění exploitů proti síťovým zařízením. V současné době je využíván ve verzi 4.8. Původně byl Metasploit naprogramován v programovacím jazyce Perl v roce 2003 programátorem H. D. Moorem. Od roku 2007 je ale kompletně přepsán do programovacího jazyku Ruby. Obsahuje největší databázi exploitů a ročně si jej stáhnou miliony uživatelů. Původně byl určen pouze pro Unixové OS, ale nyní je možné jej používat i v OS Windows, kde běží ve speciálním prostředí Cygwin. (Agarwal, 2013)

2.2 Struktura Metasploit Frameworku

V současnosti patří Metasploit Framework k nejpoužívanějším nástrojům penetračního testování. Jeho největší výhodou je snadná rozšiřitelnost a možnost implementace vlastních pluginů a nástrojů, které se dále dají bez problému zavést a ihned využívat. Architektura je rozdělena do třech hlavních kategorií, jak je vidět na obrázku 3. (Allen, 2014)Postupně si popíšeme všechny prvky architektury.



Obrázek 3 Architektura Frameworku Metasploit (Agarwal, 2013)

2.2.1 Architektura Frameworku Metasploit

Modules (moduly): Jsou nejzákladnějšími prvky celého systému. Každý modul má svůj přidělený úkol a musí ho plnit správně a efektivně. (Agarwal, 2013)

Payloads: Jedná se o kód, který řídí další akce po zdárné exploitaci. Může se jednat například o přidání uživatele do napadnutého systému nebo mohou vytvořit spojení mezi útočníkem a obětí. Metasploit obsahuje více než 250 payloadů. Přičemž můžeme dále bez problému rozšiřovat o další moduly, ať už stažené z internetu nebo vlastnoručně naprogramované. (Agarwal, 2013)

Exploits: Je krátký program, který má za úkol využít chyby v systému, programu nebo služby. Jde o nalezení a využití chyby, kterou neošetřil programátor. Může se jednat o přetečení zásobníku nebo SQL injection. (Agarwal, 2013)

Encoders: Hlavní funkce kodérů je zašifrovat část kódu tak, aby nedošlo k odhalení kódu antivirovými programy nebo firewalem. Ve své podstatě se jedná o malware, kterému by v případě odhalení byl zamítnut přístup k zařízení. (Allen, 2014)

Nops (No Operation Performed**):** Jedná se o zajištění konzistence payloadů pomocí instrukcí jazyka. (Allen, 2014)

Aux: Náplní těchto modulů je skenování, odposlouchávání, sniffing a mnoho dalšího. Ačkoliv neposkytují shell, jsou velice cenné při penetračním testování. (Allen, 2014)

Interfaces (rozhraní): Vytváří nadstavbu pro práci s moduly a interakci s uživateli. MSF využívá více rozhraní, pomocí kterých lze s Frameworkem pracovat. (Allen, 2014)

Msfconsole: patří mezi nejúčinnější a nejsilnější interface pro ovládání. Je velmi intuitivní a poskytuje přístup prakticky ke všem možnostem dostupným v MSF. Podporuje automatizaci. Není příliš vhodná pro začátečníky. (Agarwal, 2013)

Msfcli: využívá příkazový řádek dané distribuce. Je vhodný pro začátečníky. Od roku 2015 již není součástí Frameworku. Představuje zjednodušenou verzi Msfconsole, ale neobsahuje všechny možné funkce a nepodporuje automatizaci. (Agarwal, 2013)

Armitage: je grafický interface naprogramovaný v jazyce Java. Jeho účelem je zjednodušit a zrychlit práci s MSF. Speciální funkcí je možnost spolupráce více testerů. (Singh, 2012)

Msfweb: MSF může být ovladatelný i přes webový prohlížeč. Mezi jeho výhody patří snadná ovladatelnost a uživatelsky přívětivý design. (Singh, 2012)

Libraries (knihovny): Knihovnám vděčíme za to, že odpadají další problémy, které by se museli řešit samostatně. Dále knihovny zajišťují správné fungování celého Frameworku. Patří sem například HTTP požadavky nebo práce se síťovými protokoly. (Singh, 2012)

Rex: Základní knihovna pro většinu úkolů, která slouží pro práci se síťovými sockety a protokoly. (Singh, 2012)

MSF Core: Je knihovna, jejíž hlavní úkol spočívá v rozšiřování knihovny Rex. Dále je zodpovědná za komunikaci mezi rozhraními, relacemi a moduly. Jedna z jejích částí se věnuje zašifrování payloadů, aby nedošlo k prozrazení například antivirovým softwarem. (Agarwal, 2013)

MSF Base: Tato základní knihovna je navržena pro podporu knihovny MSF Core. Poskytuje základní nástroje pro práci s jádrem, například serializaci modulu na různé výstupní formáty. Navíc je rozšířena o knihovnu Interfaces, kde implementuje podporu pro různé rozhraní. (Singh, 2012)

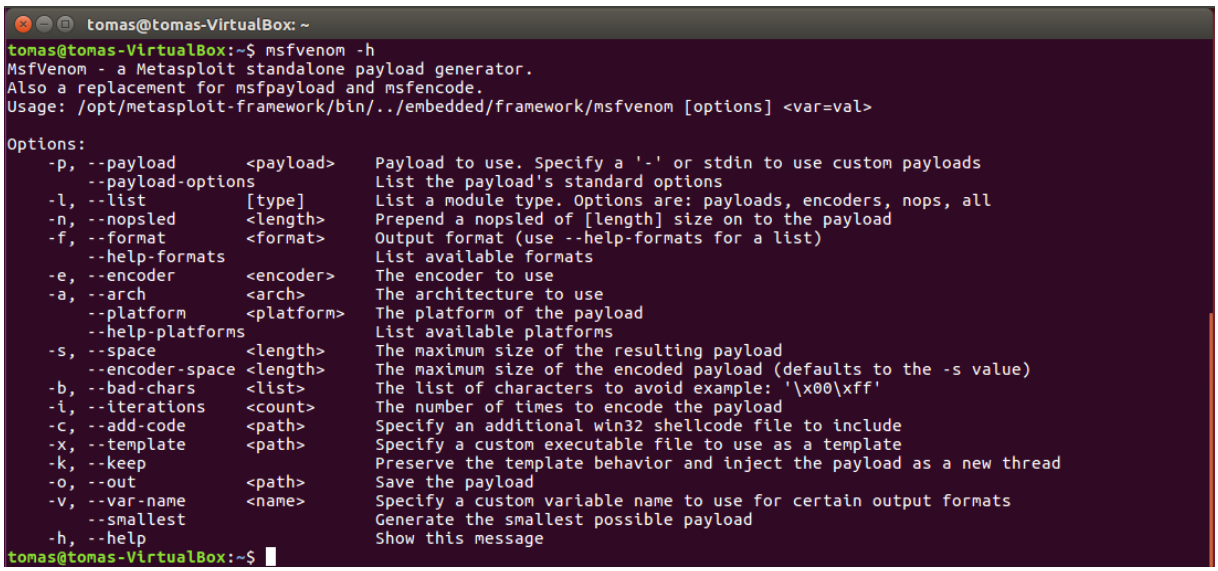
Plugins (zásuvné moduly): Jelikož se MSF snaží o co největší možnost rozšíření, tak zde jsou zavedeny zásuvné moduly, které umožňují rozšiřovat MFS o nové funkce či payloady. (Singh, 2012)

2.3 MSFvenom

MSFvenom vznikl v roce 2015 kombinací MSFpayload a MSFencode do jednoho Frameworku. MSFpayload slouží k vytvoření payloadu, který v současné době není v MSF dostupný, zatímco MSFencode slouží k zašifrování části kódu, různými šifrovacími technikami. Tímto sloučením, vznikla řada výhod, například:

- Zvýšení rychlosti
- Jeden nástroj místo původních dvou
- Standardizované rozhraní příkazové řádky

MSFvenom je program, který umožňuje vytvářet vlastní payloady. Tyto payloady, je možné dále zašifrovat tak, aby nebyly rozpoznatelné pomocí antivirového programu. Po zadání příkazu MSFvenom – h se zobrazí základní příkazy a jejich popis. (MSFvenom, 2017)



Obrázek 4 MSFvenom příkazy (zdroj – Vlastní)

V tabulce č.1 jsou popsány základní příkazy programu MSFvenom.

Tabulka 1 Popis přepínačů MSFVenom

--payload	určuje, který payload bude využit
--list	vypíše seznam modulů
--nopsled	přidá NOP do payloadu
--encoder	určuje, které šifrování bude využito, například: shikata_ga_nai

--arch	určuje typ architektury, na kterou bude payload použit
--bad-chars	vypíše seznam znaků, kterým se má program při šifrování vyhnout
--iterations	určuje, kolikrát se má provést šifrování
--template	určí existující soubor jako šablonu pro vytváření nového souboru
--out	uloží payload
--help	vypíše nápovědu jako na obrázku 4

2.4 Instalace

V této podkapitole bude ukázáno, jak nainstalovat Metasploit Framework na Unixovém OS Ubuntu ve verzi 16.04. V zájmu ulehčení instalace je možné využívat distribuci Kali Linux, která je přímo navržená pro penetrační testování a sociální inženýrství a všechny nástroje pro to vhodné již obsahuje, a to i Metasploit Framework.

2.4.1 Ubuntu

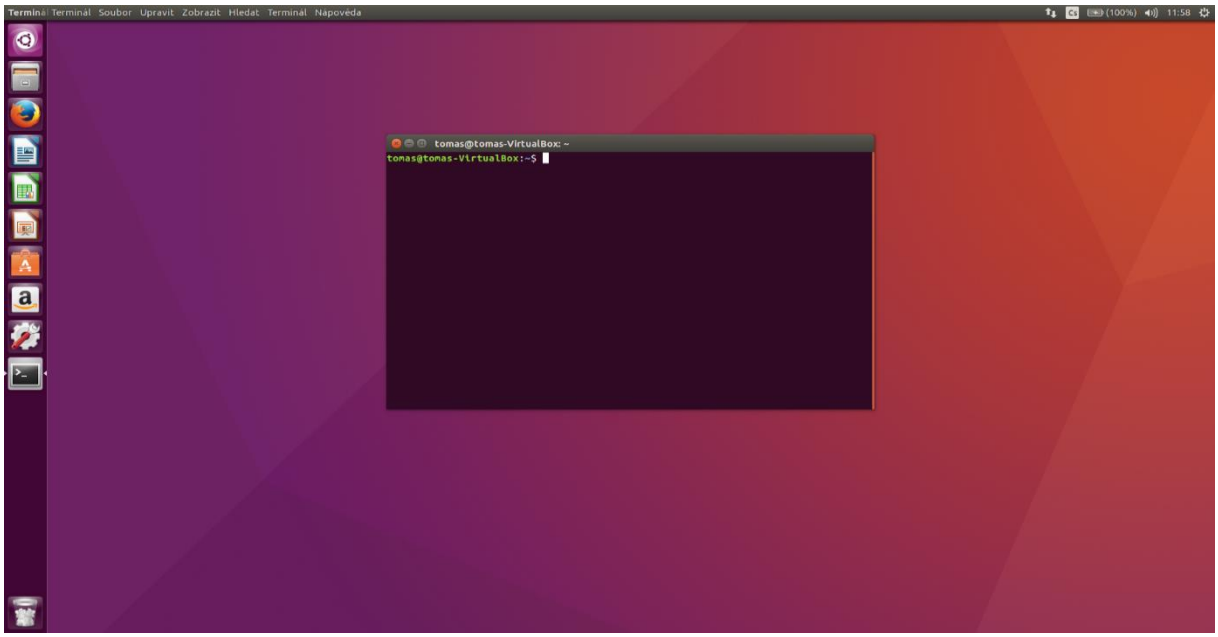
Ubuntu je unixový operační systém odvozený od distribuce Debian. Je možné jej využít pro laptopy, stolní počítače i servery. Spolu s OS Debian se řadí mezi nejoblíbenější OS na světě. Je stále vyvíjený komunitně. Nová verze vychází přibližně každý půl rok s podporou na 9 měsíců. Verze s dlouhodobou podporou LTS jsou podporovány až 5 let. Aktuální verze je 17.04. Poslední verze s LTS je verze 16.04. Jelikož se na Ubuntu vyskytuje pouze balík základních programů, bude potřeba všechny ostatní potřebné programy doinstalovat. Ubuntu využívá balíčkovací systém APT. Balíčkovací systém slouží k získání programů ze vzdálených adresářů uložených na serveru. Podstatnou výhodou balíčkovacího systému APT je automatické vyhledávání a instalace závislostí, které nám podstatně zjednoduší práci. (Co je Ubuntu, 2017)

2.4.2 Požadavky na Ubuntu

Základní systémové požadavky na instalaci OS Ubuntu jsou:

- 1 GB RAM paměti
- 5 GB místa na disku
- 1GHz frekvence procesoru (Intel Pentium nebo novější) (Stáhnout Ubuntu, 2017)

Na obrázku 5 je předvedeno pracovní prostředí Ubuntu.



Obrázek 5 Ubuntu (zdroj – Vlastní)

2.4.3 Instalace MSF na OS Ubuntu

Pro instalaci MSF na Ubuntu se předpokládá alespoň základní znalost Unixových OS. Nejdříve je důležité ověřit, zda máme nejnovější balíčky, aktualizované repositáře a vše plně aktualizované. OracleJDK by měl vše bez problému zajistit, ale někdy je vyžadováno mít nainstalovaný Oracle Java. Je předpokládáno, že uživatel pracuje v super uživatelském režimu neboli *su*.

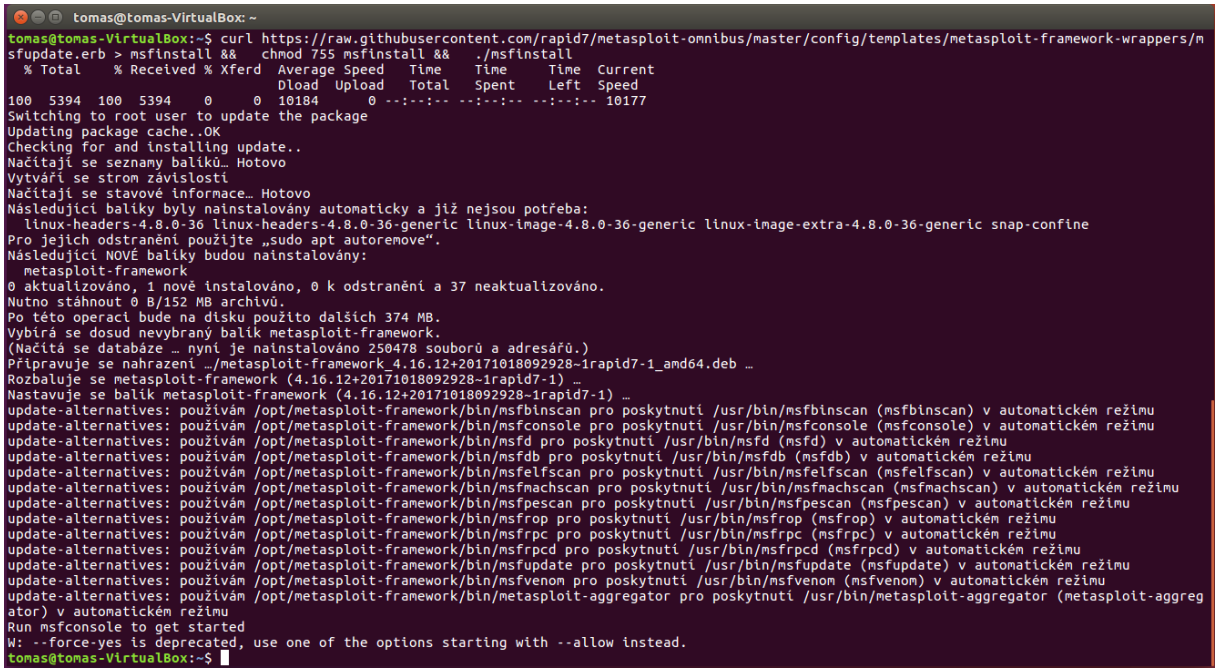
Instalaci budeme provádět pomocí webové služby GitHub.

a) Pomocí příkazu Curl se stáhne požadovaný archiv z internetové adresy a uloží jej do scriptu s názvem msfinstall.
b) Pomocí příkazu chmod 755 budou souboru nastavena přístupová práva. Nyní už stačí soubor pouze zkompilovat, a to jednoduše pomocí příkazu ./msfinstal.

```
Curl

 https://raw.githubusercontent.com/rapid7/metasploit-
omnibus/master/config/templates/metasploit-framework-
wrappers/msfupdate.erb >

msfinstall&&\

chmod 755 msfinstall && \
```

```
./msfinstall
```

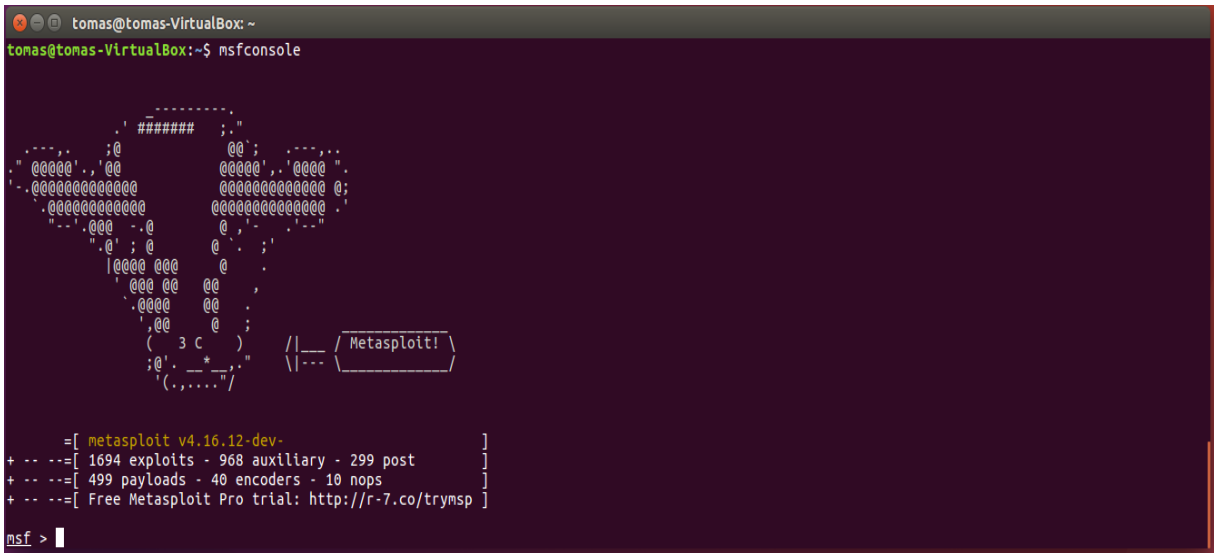
c) Pokud předcházející operace proběhla v pořádku, tak následný výpis informací v konzoli by měl vypadat obdobně jako na obrázku 6. Výpis se bude lišit podle potřebných závislostí vybraných balíčků. Některé balíčky vyžadují pro správnou funkčnost podporu dalších balíčků.



Obrázek 6 Instalace MFS (zdroj – Vlastní)

d) Pokud vše proběhlo v pořádku, stačí MSF pomocí příkazu msfconsole spustit. Na úvodní stránce jsou zobrazeny základní informace o MSF (obrázek 7). Je zde uvedena například aktuální verze, počet payloadů a exploitů.

e) Poslední věc, která je potřeba nastavit je propojení s Postgre databází. Nyní už je možné MSF plně využívat.



Obrázek 7 Úvodní rozhraní MSF (zdroj – Vlastní)

2.5 Ovládání MSF

V této části budou popsány základní skupiny příkazů k ovládání MSF. Po napsání příkazu help do msfconsole budou vypsány základní příkazy, které jsou rozděleny do pěti kategorií.

- Core Commands
- Module Commands
- Job Commands
- Resource Script Commands
- Database Backend Commands

Core Commands: Zde se nacházejí základní příkazy pro práci s MSF, je možné zde zjistit verzi, historii příkazů, je možné zde načítat, mazat a upravovat pluginy nebo nastavit jednoduché routování. (MSFconsole Commands, 2017)

Module Commands: V této části se nachází základní příkazy pro práci s moduly, je možné zde zjistit bližší informace o používaných modulech, vyhledávat moduly podle jména nebo popisu a v neposlední řadě moduly upravovat. (MSFconsole Commands, 2017)

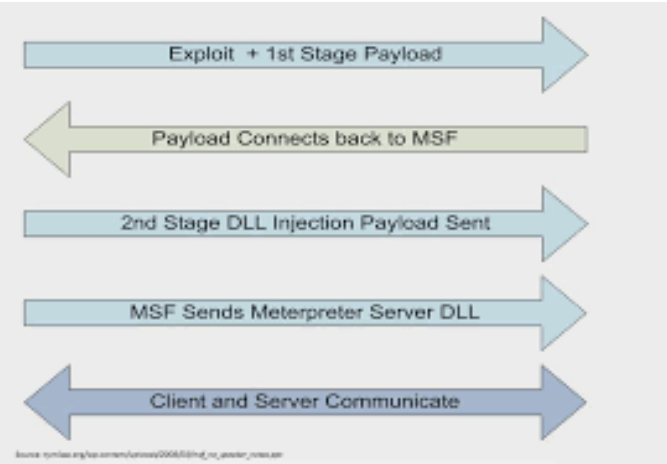
Job Commands: Tato část se zabývá prací s moduly, které běží na pozadí, je zde možné spouštět, zobrazovat a zastavovat tyto moduly. (MSFconsole Commands, 2017)

Resource Script Commands: Tato sekce se zabývá automatizováním úloh. Nabízí možnost ukládat provedené příkazy do skriptu a znovu je spouštět. (MSFconsole Commands, 2017)

Database Backend Commands: Tato část příkazů se zabývá prací s PostgreSQL databází. Je možné zde připojovat a odpojovat databázi, zjišťovat stav a nahrávat data do databáze pomocí skenování sítě nebo ručně. Spolupráce s databází rozšiřuje funkcionalitu MSF. (MSFconsole Commands, 2017)

2.6 Meterpreter

Meterpreter, zkratka The Meta-Interpreter je speciální payload, který je součástí MSF. Jedná se o mnohostranného, rozšiřitelného, nesnadno odhalitelného pomocníka, jehož úkolem je poskytovat komplexní řešení a pokročilé funkce. Mezi jeho největší výhody patří nesnadná odhalitelnost pomocí antivirového programu. Způsob, jakým je tato výhoda využita, je velmi jednoduchý. Meterpreter pracuje s DLL soubory, které jsou nahrány přímo do paměti a žádným způsobem nejsou zapisovány na pevný disk, kde by byly vystaveny možnému odhalení antivirovým programem. Další výhodou, je dynamické načítání skriptů a zásuvných modulů za běhu programu za účelem jeho rozšíření. Tím odpadá nutnost navazování nového spojení nahrávání nových rozšíření. Důležitou součástí meterpreteru je fakt, že celá komunikace probíhá zašifrovaná. Nakonec meterpreter poskytuje také snadný multitasking tím, že dává k dispozici možnost vytvořit více relací, které může uživatel bez problému zpracovávat. Meterpreter je možné si představit jako jakýkoliv příkazový terminál, kde je možné pomocí příkazů provádět jednoduché úkoly. Funguje na základním principu klient – server, jakmile bude navázáno spojení, je možné odesílat příkazy na meterpreter server, který odesílá odpovědi zpět na útočící zařízení. (Agarwal, 2013; Allen, 2014)



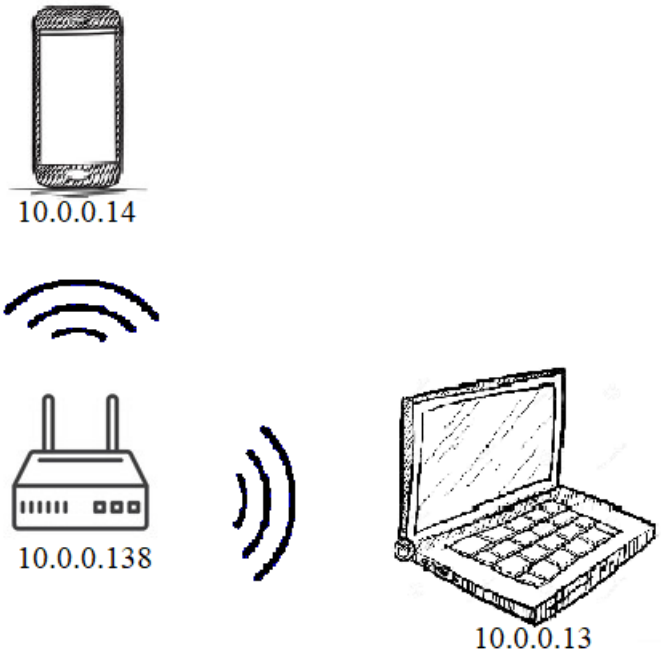
Obrázek 8 Meterpreter nahrání do paměti zdroj – (Agarwal, 2013)

Na obrázku 8 je předvedeno nahrávání meterpreteru do paměti. V první části je odeslán exploit a první payload na napadené zařízení. Po provedení exploitace se odeslaná část připojí

k cíli a pokusí se připojit zpět na využité rozhraní přes předem nastavený komunikační kanál. Ve druhé fázi dochází k načtení DLL zvoleného payloadu. Současně s tím MSF odesílá své DLL aby mohlo dojít k navázání spojení. V poslední fázi dochází k načtení rozšíření. Všechna rozšíření jsou nahrávána pomocí protokolu TLS. (Agarwal, 2013)

2.7 Útok na mobilní telefon

Útok na mobilní telefon probíhal na domácí síti, odpojené od internetu. Rozvržení sítě, včetně IP adres je ukázáno na obrázku 9.



Obrázek 9 Ukázka zapojení sítě (zdroj – Vlastní)

2.7.1 Informace o mobilním telefonu

- Výrobce: Samsung electronics
- Verze Android: 7.0.1 Nougat
- Verze jádra: 3.18.14
- Přidělená IP adresa: 10.0.0.14

2.7.2 Úvod o útoku

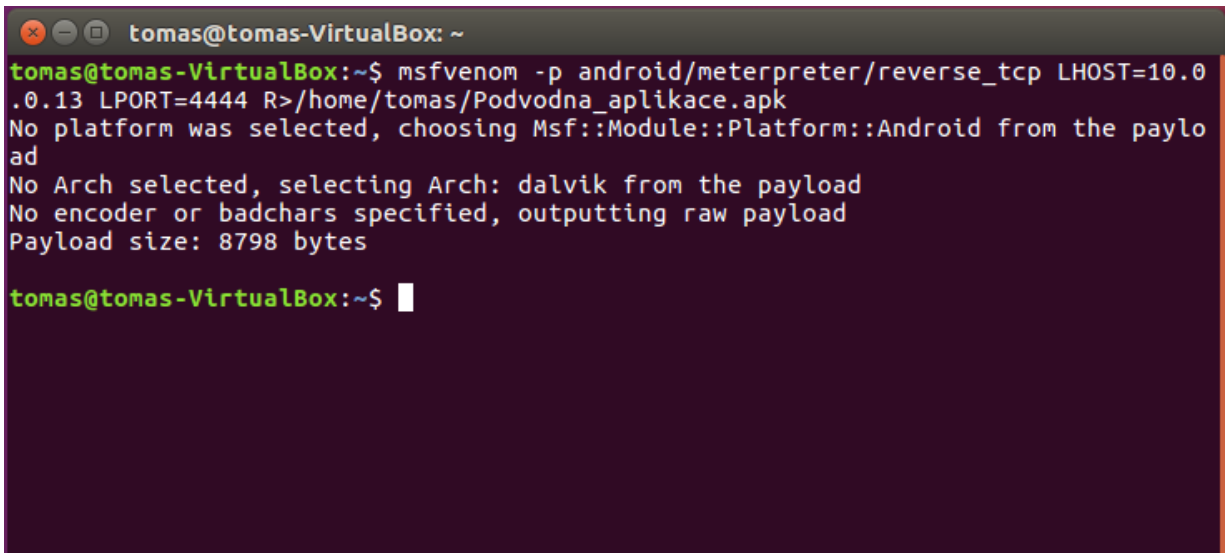
K útoku na mobilní telefon je využit Metasploit Framework a jedno z jeho rozšíření MSFvenom, obě tyto aplikace a jejich funkcionality jsou popsány výše. První část spočívá ve vytvoření podvodného programu, který umožňuje vytvořit zadní vrátka do zařízení. Dále je

potřeba tento program nainstalovat do mobilního telefonu, po instalaci a spuštění aplikace, dojde k nastavení spojení. Jakmile dojde k nastavení relace mezi zařízeními, není problém získávat, nahrávat a odposlouchávat data z mobilního telefonu.

V prvním scénáři dojde k vytvoření podvodné aplikace bez zašifrování, a je nahrána do telefonu bez jakéhokoliv antivirového programu. Ve druhém scénáři bude vytvořená aplikace též nezašifrovaná, ale dojde k instalaci antivirového programu na napadené zařízení. Ve třetím scénáři dojde k infikování existující aplikace škodlivým kódem pomocí MSFVenom.

2.7.3 Scénář č. 1.

První krok je vytvoření podvodné aplikace pomocí aplikace MSFvenom. Po provedení by měl výstup vypadat podobně jako na obrázku 10.

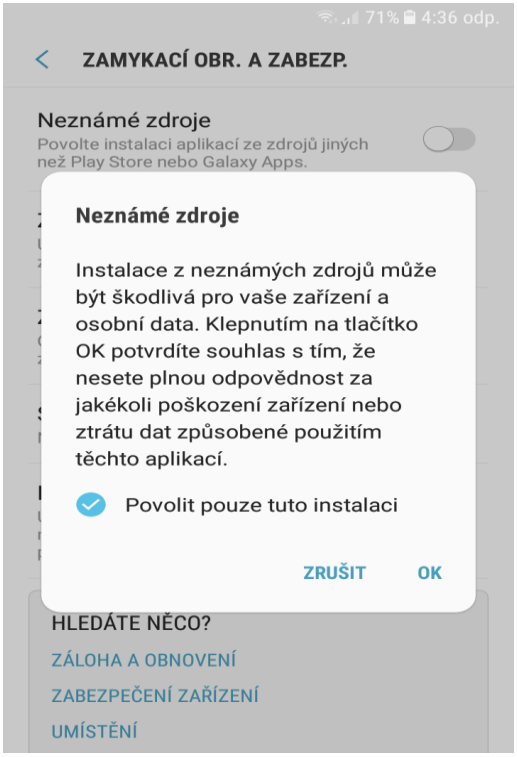


Obrázek 10 Vytvoření podvodné aplikace (zdroj – Vlastní)

Pomocí zobrazeného příkazu vytvoříme payload, který vytvoří meterpreter server na zařízení. Pomocí parametru LHOST nastavíme IP adresu zařízení, ze kterého bude meterpreter ovládán. Dále pomocí LPORT nastavíme port, na kterém bude zařízení naslouchat. A celou tuto aplikaci uložíme pod názvem: Podvodna_aplikace.apk. Ve výpisu je vidět, že architektura programu pro napadený systém byla zvolena automaticky, nebylo zvolené žádné šifrování a je zde zobrazena velikost celého vytvořeného souboru.

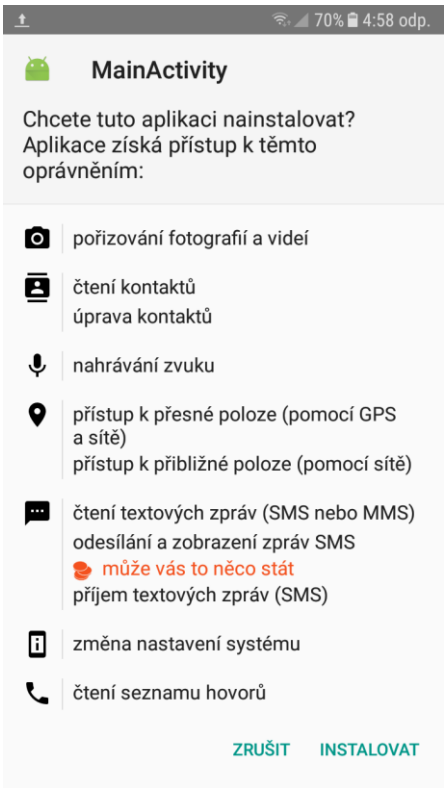
Dalším bodem útoku je dopravit vytvořenou aplikaci k cílovému zařízení. Většinou se takto vytvořené aplikace sdílejí pomocí data sdílejících serverů, kde jsou vydávány za jiné aplikace, než doopravdy jsou. Takto nasdílené aplikace přilákají mnoho potencionálních obětí.

Po dopravení takto vytvořené aplikace do telefonu, je nutné ji nainstalovat. Jelikož se jedná o instalaci z neznámého zdroje je nutné v telefonu povolit tuto možnost. Aplikace nebyla zašifrovaná, ale protože telefon nedisponuje žádným antivirovým programem, tak aplikace zůstala neobjevena.



Obrázek 11 Povolení instalace z neznámých zdrojů (zdroj – Vlastní)

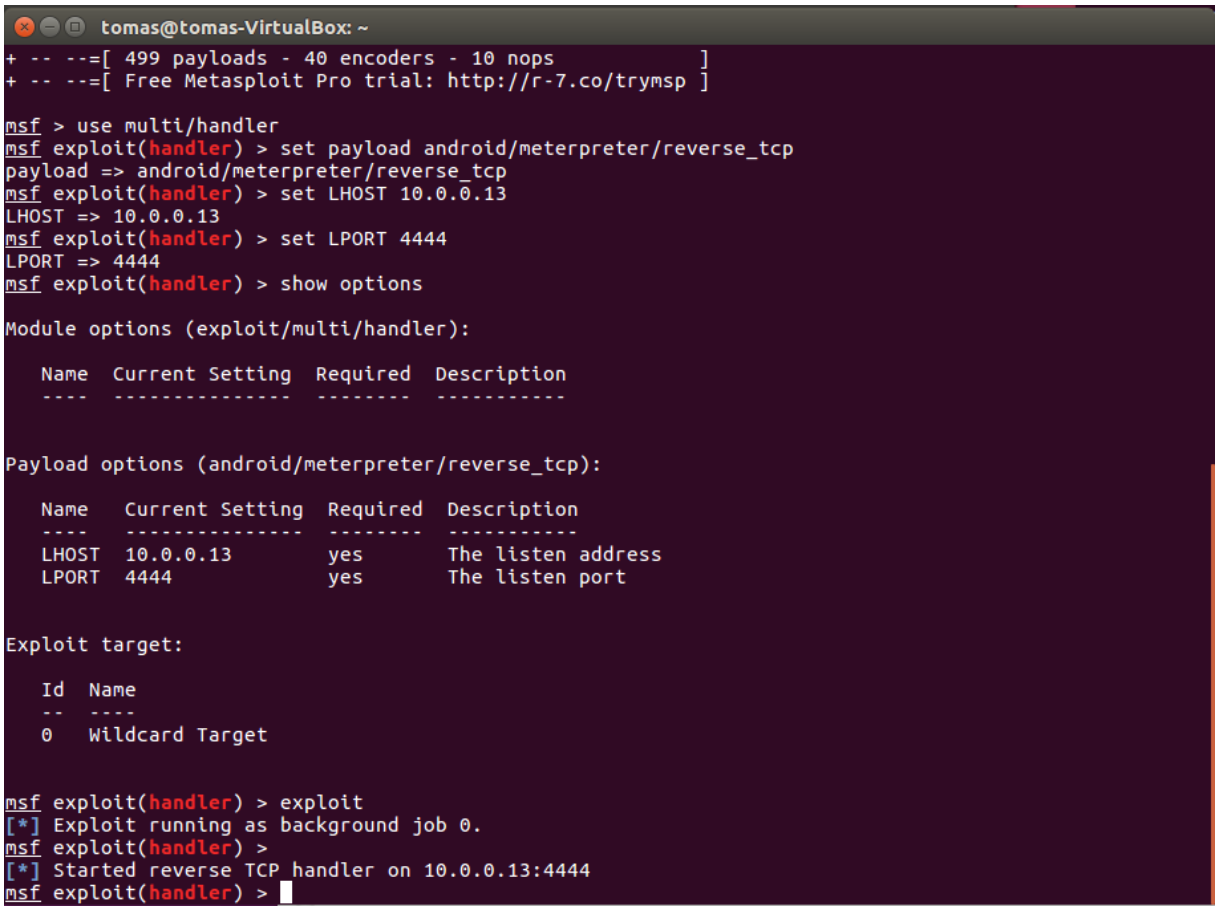
Dále dochází k určení oprávnění, ke kterým získá aplikace přístup a co může využívat. Naše podvodná aplikace má k dispozici velké množství oprávnění a některá nesouvisí s jejím využitím. Jak je předvedeno na obrázku 12.



Obrázek 12 Nastavení práv (zdroj – Vlastní)

Zkušenější uživatel by jistě zjistil, že se nejedná o aplikaci, jakou by vyžadoval. Ať už podle názvu aplikace nebo podle požadovaných práv, která aplikace vyžaduje. Pokud vše proběhlo v pořádku, tak po spuštění aplikace uživatelem dojde k vytvoření spojení mezi obětí a útočníkem.

Dalším úkolem je nastavit naslouchání na útočícím zařízení. K tomuto kroku využijeme MFS a budeme využívat rozhraní msfconsole. Po provedení následujících příkazů, by měl výstup z konzole vypadat jako na obrázku 13.



Obrázek 13 Výběr a nastavení typu payloadu (zdroj – Vlastní)

Na obrázku 13 je nejdříve nastaven multi/handler, což je prostředník, který obstarává spuštěné exploity. Díky handleru můžeme spolehlivě a jednoduše zpracovávat spuštěné exploity. Dalším krokem, který je nezbytné provést je vybrání odpovídajícího payloadu, který chceme využívat. Nyní už jen zbývá nastavit IP adresu útočícího zařízení a port, na kterém bude zařízení naslouchat. Po tomto základním nastavení je důležité si nastavení zkontrolovat, to se může provést pomocí příkazu show options. Pokud je všechno správně a v pořádku, tak můžeme spustit naslouchání pomocí příkazu exploit. V této fázi bude aplikace čekat na spuštění aplikace na mobilním telefonu, kterou může spustit jen a pouze uživatel. Pokud bude spuštěna, dojde k navázání spojení. Následně bude telefon plně ovladatelný.

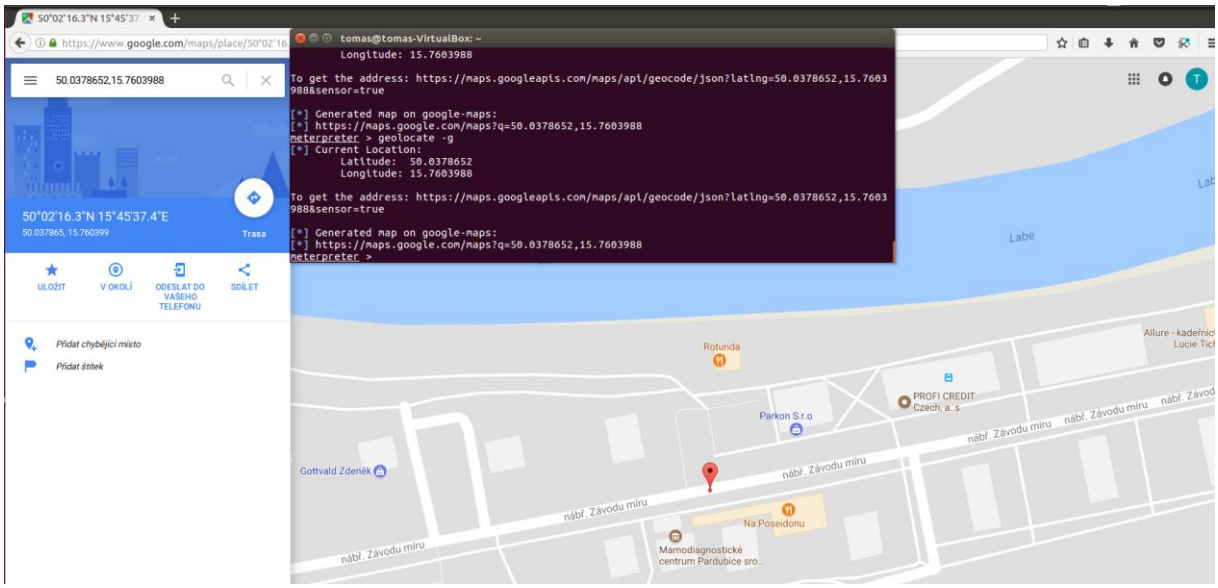
Souhrn všech spojení je možné zobrazit pomocí příkazu sessions.

```
msf exploit(handler) > sessions

Active sessions
===============

  Id  Name  Type                     Information          Connection
  --  ----  ----                     -----------          ----------
  1         meterpreter dalvik/android  u0_a192 @ localhost  10.0.0.13:4444 -> 10.0.0.10:35552 (10.0
.0.10)

msf exploit(handler) > [*] 10.0.0.10 - Meterpreter session 1 closed.  Reason: Died
```

Obrázek 14 Otevřené relace (zdroj – Vlastní)

Pro práci se spojeními můžeme využít přepínač sessions –i „id session“, pomocí kterého můžeme pohodlně mezi spojeními přepínat. Pokud ovšem uživatel odpojí telefon od sítě, nebo odinstaluje aplikaci, nebude již možné telefon znovu napadnout.

V tomto okamžiku má útočník plně pod kontrolou mobilní telefon. V tuto chvíli stačí pouze malý okamžik k získání všech potřebných informací. Kdyby útočník využil například skript, který by hned po připojení začal stahovat žádané informace, stačilo by mu k jejich zisku pouze několik desítek vteřin v závislosti na rychlosti připojení. Pokud by útočník nechtěl pouze získávat informace, je velmi jednoduché nahrát do napadeného telefonu další škodlivé aplikace. Na obrázku 15 je pomocí příkazu geolocate – g zjištěná aktuální pozice telefonu.



Obrázek 15 Zjištění pozice (zdroj – Vlastní)

Pro útočníka není problém získat podrobnější informace o telefonu pomocí jednoduchého příkazu sysinfo. Pomocí, kterého můžeme získat informace o jádře OS a o typu meterpreteru.

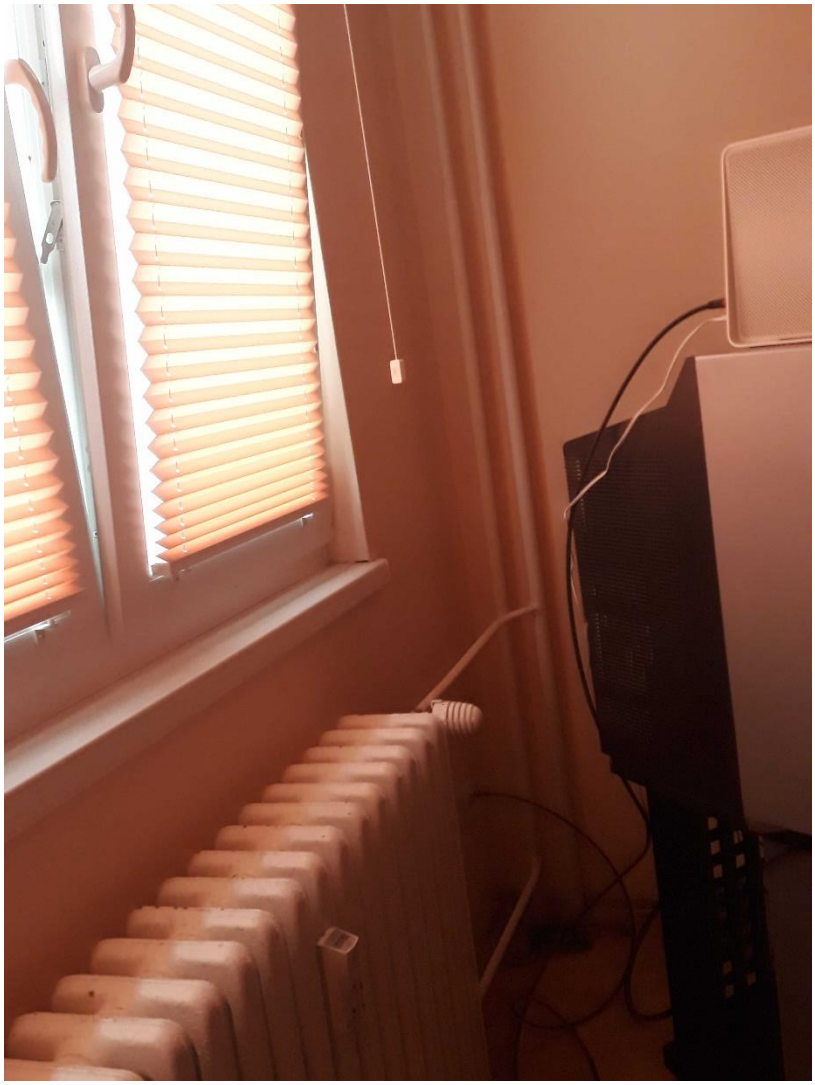
```
meterpreter > sysinfo -h
Computer    : localhost
OS          : Android 7.0 - Linux 3.18.14-12274792 (armv8l)
Meterpreter : dalvik/android
```

Obrázek 16 Zobrazení informací (zdroj – Vlastní)

V tuto chvíli bylo zjištěno pouze několik málo informací o telefonu a o poloze. Pokud by útočník chtěl získat citlivější data např.: výpis hovorů stačí mu k tomu jediný příkaz dump_calllog. Nyní má útočník přístup ke všem hovorům a telefonním číslům, jež byly ze zařízení uskutečněny.

Obrázek 17 Zobrazení všech hovorů (zdroj – Vlastní)

V neposlední řadě je možné telefon i aktivně ovládat, je zde možnost využívat fotoaparát, kameru a mikrofon mobilního telefonu nebo dokonce odesílat SMS zprávy. Takovéto ovládání už představuje problém, jelikož na tom uživatel může mít velkou peněžní ztrátu.

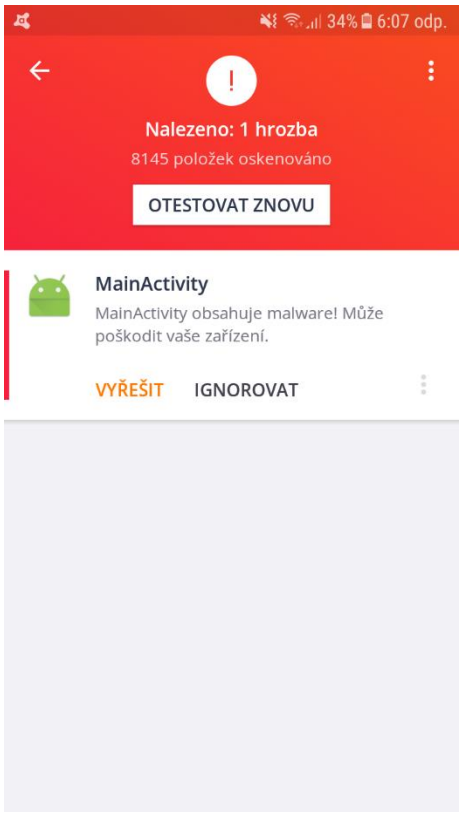


Obrázek 18 Fotografie vyfotografovaná pomocí MFS (zdroj – Vlastní)

2.7.4 Scénář č. 2.

Ve scénáři č. 2 dojde k vytvoření stejné podvodné aplikace pomocí MSFvenom jako ve scénáři č. 1. Změnou v tomto scénáři, bude nainstalovaný antivirový program v mobilním telefonu. Zvolený antivirový program bude velmi populární Avast antivirus, který nabízí bezplatnou 30denní Premier verzi, která poskytuje velké množství možných ochran např.: Ochrana před falešnými webovými stránkami, pokročilý firewall, blokování spamu a phishingových e-mailů.

Takto zabezpečený telefon odhalí nezašifrovanou aplikaci ještě před tím, než bude dovoleno jí nainstalovat a označí ji jako nebezpečnou.

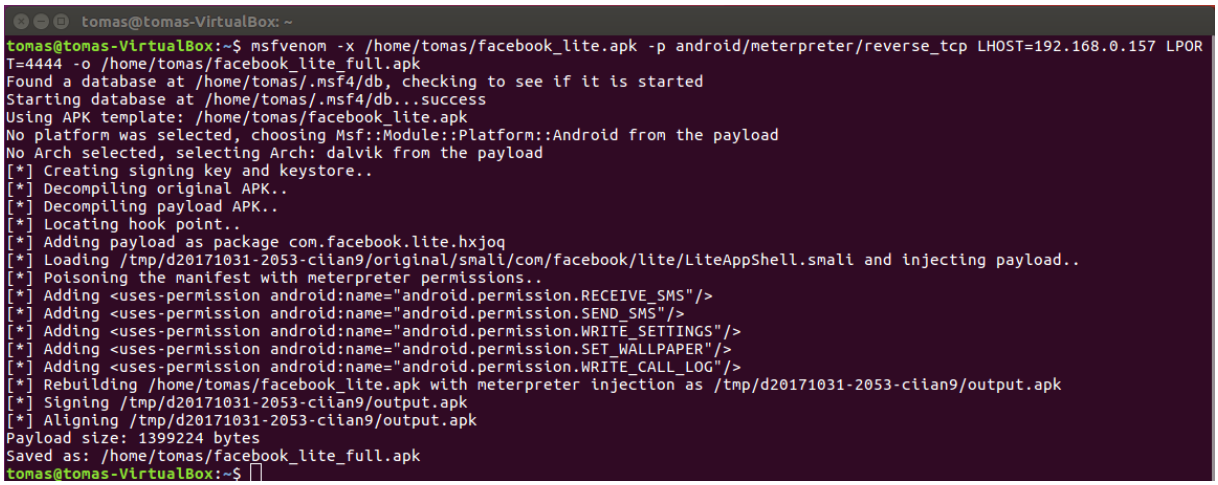


Obrázek 19 Odhalená hrozba (zdroj – Vlastní)

Nyní by již uživatel věděl, že se jedná o podvodnou aplikaci a pravděpodobně by ji nenainstaloval.

2.7.5 Scénář č. 3.

V posledním scénáři je provedeno nahrání škodlivého kódu do již existující aplikace. Takto vytvořená aplikace nevzbuzuje žádné podezření a jeví se naprosto věrohodně. V tomto scénáři využijeme aplikaci Facebook Lite. V zařízení bude nainstalován antivirový program, tak jako byl nainstalován ve scénáři č. 2.



Obrázek 20 MSFVenom nahrání škodlivého kódu do existujícího souboru (zdroj – Vlastní)

Pomocí přepínače –x zde zvolíme soubor, v tomto případě je využita aplikace facebook_lite.apk, který bude zvolen jako šablona pro vytvoření nového souboru facebook_lite_full.apk, který již je infikovaný škodlivým kódem. Takto vytvořenou aplikaci už jen stačí nainstalovat na napadené zařízení. Na napadeném zařízení je nezbytné povolit v nastavení možnost instalace z neznámých zdrojů. Při distribuci aplikace na zařízení ani při instalaci nedošlo k objevení programu antivirovým programem a aplikace mohla být bez problému spuštěna bez jakéhokoliv podezření ze strany uživatele. Po spuštění aplikace dojde ke vytvoření relace mezi napadeným zařízením a útočníkem. V tento okamžik má útočník již plnou kontrolu nad zařízením a je možné jej ovládat jako ve scénáři č. 1.

ZÁVĚR

OS Android je založen na jádře Linuxu a dnes je součástí základního vybavení téměř každého mobilního zařízení, tabletu, chytrých televizí a dalších zařízení. Jedná se o jeden z nejpoužívanějších mobilních operačních systémů.

Úvod práce se zabývá rozborem OS Android, vyvinutým společností Google Inc., obecně, jeho historií, kdy od prvního vydání v roce 2008 prošel systém mnoha změnami, jeho verzemi, z čehož vyplývá, že verze 6.0 Marshmallow je nejrozšířenější verzí (o 10,1 % před verzí 5.1 Lollipop). Dále je v této kapitole rozebrána struktura OS Android, kde jsou vysvětleny jednotlivé vrstvy tohoto systému. Následně je analyzováno zabezpečení, které je možné využít v zařízení s OS Android. Podporované zabezpečení dokáže výrazně ovlivnit celkovou ochranu zařízení a uživatele. Nicméně platí, že největší hrozbou pro uživatele je uživatel sám.

Myšlenku uvedenou výše potvrzuje i praktická část této práce, kde je vytvořen malware pomocí aplikace MSFvenom, který je následně distribuován na napadené zařízení. Takto vytvořený malware a další škodlivé aplikace se zkušenému uživateli mohou jevit velice podezřele, jelikož vyžadují výrazné povolení uživatelských práv, které nesouvisí s jejich obsahem, jak je předvedeno ve scénáři č. 1. Z důvodu neustálého snižování věku uživatelů chytrých telefonů a jisté nerozvážnosti se takto vytvořený malware může celkem snadno šířit, získávat citlivá data a napáchat podstatné škody. Připojení k zařízení po instalaci škodlivé aplikace a následná krádež citlivých dat je předvedena ve scénáři č. 1. Vzhledem k rychlosti internetových připojení stačí útočníkovi k zisku dat okamžik v řádu několika sekund.

Celkovou bezpečnost zařízení může uživatel zvýšit pomocí dodatečného softwaru. Může se jednat například o firewall nebo o antivirový program, kterých je k dispozici velké množství. Tyto aplikace jsou schopné odhalit škodlivé aplikace v případě uživatelské nerozvážnosti, jak je předvedeno ve scénáři č. 2. Takto zabezpečené zařízení dává uživateli reálnou šanci odhalit potencionální hrozbu.

Scénář č. 3 potvrdil, že je možné napadnout i zařízení, které využívá určitou úroveň zabezpečení. Antivirus AVG nebyl schopen rozpoznat aplikaci Facebook Lite, která byla infikovaná pomocí programu MSFvenom. Takto vytvořená aplikace nejevila žádné známky podezření, pouze požadovala povolení využívání více druhů oprávnění, než je obvyklé.

Společnost Google poskytuje úložiště aplikací Google Play, které nabízí uživateli otestované a bezpečné aplikace, jež napomáhají ke zvýšení bezpečnosti. Některé aplikace zpřístupněné

přes Google Play mají svou cenu, která může motivovat uživatele ke stahování aplikací z neznámých zdrojů a tím podporovat šíření malwaru. I přes veškerou snahu o maximální zabezpečení zůstává nejslabším článkem pomyslného řetězu uživatel.

3 BIBLIOGRAFIE

AGARWAL, Monika a Abhinav SINGH, 2013. *Metasploit penetration testing cookbook*. 2nd ed. Birmingham, UK: Packt Pub. ISBN 978-178-2166-788.

ALLEN, Lee, Tedi HERIYANTO a Shakeel ALI, 2014. *Kali linux: assuring security by penetration testing*. Sec. Edt. Birmingham: Packt Publishing Limited. ISBN 978-184-9519-489.

Android 6.0 Marshmallow, 2015. *Android Developers* [online]. Armonk: International Business Machines Corporation and others [cit. 2017-11-18]. Dostupné z: https://developer.android.com/about/versions/marshmallow/index.html

Android 7.0 for Developers, 2015. *Android Developers* [online]. Armonk: International Business Machines Corporation and others [cit. 2017-11-18]. Dostupné z: https://developer.android.com/about/versions/nougat/android-7.0.html

Android 7.0 Nougat, 2015. *Android Developers* [online]. Armonk: International Business Machines Corporation and others [cit. 2017-11-18]. Dostupné z: https://developer.android.com/about/versions/nougat/index.html

Android KitKat, 2015. *Android Developers* [online]. Armonk: International Business Machines Corporation and others [cit. 2017-11-18]. Dostupné z: https://developer.android.com/about/versions/kitkat.html

Android Lollipop, 2015. *Android Developers* [online]. Armonk: International Business Machines Corporation and others [cit. 2017-11-18]. Dostupné z: https://developer.android.com/about/versions/lollipop.html

Android Oreo, 2015. *Android Developers* [online]. Armonk: International Business Machines Corporation and others [cit. 2017-11-18]. Dostupné z: https://developer.android.com/about/versions/oreo/index.html

Co je Ubuntu, 2017. *Ubuntu.cz* [online]. Londýn: Canonical Ltd. [cit. 2017-10-27]. Dostupné z: https://www.ubuntu.cz

Jelly Bean, 2015. *Android Developers* [online]. Armonk: International Business Machines Corporation and others [cit. 2017-11-18]. Dostupné z: https://developer.android.com/about/versions/jelly-bean.html

KILIÁN, Karel, 2015. Historie Androidu v kostce aneb Od verze 1.0 až po Android M. *Svět Androida* [online]. SvetAndroida.cz [cit. 2017-11-17]. Dostupné z: https://www.svetandroida.cz/historie-androidu-201506/

KOVACZICZ, Johanna, 2015. Android 6 je oficiálně na světě: Bude nám Marshmallow chutnat? (shrnutí). *Svět Androida* [online]. SvetAndroida.cz [cit. 2017-11-18]. Dostupné z: https://www.svetandroida.cz/android-6-shrnuti-201509/

MACHO, Daniel, 2016. 5 důležitých změn, které do smartphonů přinese Android N. *Svět Androida* [online]. SvetAndroida.cz [cit. 2017-11-18]. Dostupné z: https://www.svetandroida.cz/5-dulezitych-zmen-android-n-201605/

MSFconsole Commands, 2017. *OFFENSIVE security* [online]. Offensive Security [cit. 2017-11-9]. Dostupné z: https://www.offensive-security.com/metasploit-unleashed/msfconsole-commands/

MSFvenom, 2017. *OFFENSIVE security* [online]. Offensive Security [cit. 2017-11-9]. Dostupné z: https://www.offensive-security.com/metasploit-unleashed/msfvenom/

Platform Architecture, 2015. *Android Developers* [online]. Armonk: International Business Machines Corporation and others [cit. 2017-11-18]. Dostupné z: https://developer.android.com/guide/platform/index.html

Security, 2015. *Android Source* [online]. Armonk: International Business Machines Corporation and others [cit. 2017-11-18]. Dostupné z: https://source.android.com/security/

SINGH, Abhinav, 2012. *Metasploit Penetration Testing Cookbook*. New Edition. Birmingham: Packt Publishing, Limited. ISBN 978-184-9517-423.

SRB, Luboš, 2017. 3+1 jednoznačně nejlepší funkce, které přináší Android 8.0 Oreo. *Mobilizujeme* [online]. Praha: Mobilizujeme.cz [cit. 2017-11-18]. Dostupné z: https://mobilizujeme.cz/clanky/31-jednoznacne-nejlepsi-funkce-ktere-prinese-android-o

Stáhnout Ubuntu, 2017. *Ubuntu.cz* [online]. Londýn: Canonical Ltd. [cit. 2017-11-18]. Dostupné z: https://www.ubuntu.cz/stahnout

System and kernel security, 2015. *Android Source* [online]. Armonk: International Business Machines Corporation and others [cit. 2017-11-18]. Dostupné z: https://source.android.com/security/overview/kernel-security

TRLICA, David, 2017a. Každý pátý Android telefon už má systém na Nougatu. Jak je na tom ten váš?. *Svět Androida* [online]. SvetAndroida.cz [cit. 2017-11-16]. Dostupné z: https://www.svetandroida.cz/android-telefon-nougat-aktualizace-201711/

TRLICA, David, 2017b. Android 8 novinky: Shrnutí hlavních funkcí. *Svět Androida* [online]. SvetAndroida.cz [cit. 2017-11-18]. Dostupné z: https://www.svetandroida.cz/android-8-shrnuti-novinek-201708/

UJBÁNYAI, Miroslav, 2012. *Programujeme pro Android*. Vyd. 1. Praha: Grada. Průvodce (Grada). ISBN 978-80-247-3995-3.

4 PŘÍLOHY

Příloha A – Seznam *použitého softwarového a hardwarového vybavení*

Hardwarové vybavení:

- Dell Inspiron 7566
- Zyxel P660HW-T3 v2
- Samsung Galaxy J5 2016

Softwarové vybavení:

- Linuxová distribuce Ubuntu verze 16.04
- Oracle VM VirtualBox verze 5.1.14
- Metasploit Framework verze 4.16.14
- OS Android verze 7.0.1 Nougat

Příloha B – Seznam *souborů na CD*

- soubor Penetrační testování mobilů – TomášKodym.pdf