VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

BAKALÁŘSKÁ PRÁCE

Brno, 2019 Jan Procházka

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

PROSTŘEDÍ PRO ANALÝZU PODEZŘELÉHO ZAŘÍZENÍ
ENVIRONMENT FOR ANALYZING SUSPICIOUS DEVICE

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE Jan Procházka
AUTHOR

VEDOUCÍ PRÁCE Ing. Lukáš Malina, Ph.D.
SUPERVISOR

BRNO 2019

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní obor **Informační bezpečnost**
Ústav telekomunikací

Student: Jan Procházka ***ID:*** 186169
Ročník: 3 ***Akademický rok:*** 2018/19

NÁZEV TÉMATU:

Prostředí pro analýzu podezřelého zařízení

POKYNY PRO VYPRACOVÁNÍ:

Cílem práce je návrh hardwarového a softwarového prostředí, které umožní prozkoumat elektronické důkazy na externím médiu, kterým může být škodlivým kódem kontaminovaný hw např. USB flash disk, externí disk, případně image kontaminovaného disku. Toto prostředí musí reflektovat trend "sandbox evasion" a "sandbox escape" technik a umožnit detekci evazivních a intrusivních technik. Analýza obsahu médií/zařízení musí probíhat v bezpečném kontrolovaném prostředí a znemožnit šíření škodlivého kódu i za předpokladu, že jde o dosud neprozkoumaný škodlivý kód, tedy umožnit detekci a rozbor anomálního chování. Zanalyzujte možnosti a popište požadavky na vytvoření tohoto prostředí. Dalším dílčím cílem je návrh prostředí a porovnání open source alternativ.
V rámci bakalářské práce prostředí vytvořte. Vytvořte i externí médium kontaminované škodlivým kódem. Vytvořené prostředí použijte pro simulovanou analýzu kontaminovaného externího média. Průběh a závěry simulace popište a prezentujte.

DOPORUČENÁ LITERATURA:

[1] PFLEEGER, Charles P.; PFLEEGER, Shari Lawrence. Analyzing computer security: a threat/vulnerability/countermeasure approach. Prentice Hall Professional, 2012.

[2] GARCIA-TEODORO, Pedro, et al. Anomaly-based network intrusion detection: Techniques, systems and challenges. computers & security, 2009, 28.1-2: 18-28.

Termín zadání: 1.2.2019 ***Termín odevzdání:*** 27.5.2019

Vedoucí práce: Ing. Lukáš Malina, Ph.D.
Konzultant:

prof. Ing. Jiří Mišurec, CSc.
předseda oborové rady

ABSTRAKT

Bakalářská práce se zabývá návrhem prostředí pro analýzu podezřelého zařízení. Tímto zařízením může být například škodlivým kódem kontaminovaný disk nebo mobilní zařízení. Cílem práce je navrhnout efektivní a jednoduché prostředí s použitím open source produktů. Finální prostředí by mělo být schopné provádět jak povrchovou, tak i hloubkovou analýzu dat. Teoretická část poskytuje informace spojené s problematikou práce a obsahuje pojmy jako jsou Sandbox, Malware, Android. Ty jsou popsány z pohledu důležitého pro pochopení analýzy malwaru vyskytujícího se převážně na mobilním zařízení. Praktická část popisuje použitý hardware a software pro návrh prostředí a obsahuje názorné ukázky analýz škodlivým kódem kontaminovaných externích zařízení. Jedná se převážně o mobilních zařízení se systémem Android.

KLÍČOVÁ SLOVA

malware, sandbox, android, analýza, open source, raspberry pi, USB flash disk, mobilní telefon, extrakce dat, škodlivá aplikace

ABSTRACT

This bachelor thesis focuses on a design of enviroment for analysis of a suspicious device. Such device may be for example a disc contaminated by malicious code or a mobile device. The aim of this work is to design an efficient and simple solution using open source products. The final designed environment should be capable of performing both surface and in-depth data analysis. The theoretical part offers an information related to the scope of addressed problem and includes terms such as Sandbox, Malware, Android. These are described from the point of view of understanding the analysis of malware occurring predominantly on mobile devices. The practical part describes the used hardware and software for the design of the environment and it contains examples of analyzes of the external devices contaminated by a malcode. These examples are mainly for Android mobile devices.

KEYWORDS

malware, sandbox, android, analysis, open source, raspberry pi, USB flash disk, mobile phone, data extraction, malicious application

PROCHÁZKA, Jan. *Prostředí pro analýzu podezřelého zařízení*. Brno, Rok, 65 s. Bakalářská práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Vedoucí práce: Ing. Lukáš Malina, Ph.D.

Vysázeno pomocí balíčku `thesis` verze 3.03; `http://latex.feec.vutbr.cz`

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „Prostředí pro analýzu podezřelého zařízení“ jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

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

Brno ....

podpis autora

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu bakalářské práce panu Ing. Lukáši Malinovi, Ph.D. a panu Jaroslavu Rusovi z firmy Anect za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Brno ....

podpis autora

Obsah

Seznam obrázků

Seznam výpisů

1 Úvod

Bezpečnost je jedna z nejdůležitějších faktorů v informační bezpečnosti. Pod bezpečností si lze představit cokoliv. Jednou z věcí je stav, kdy jsou do jisté míry chráněna aktiva před hrozbou. Může se jednat o citlivá data, osobní informace, firemní tajemství, bankovní informace a jiné. Tyto aktiva jsou pro určitou osobu, osoby nebo firmy velmi důležitá. Neustále jsou však vystaveny hrozbě. Hrozbě, že dotčená data budou odcizena, smazána nebo poškozena. Pro ochranu je potřeba vytvářet mnoho opatření. Jednou z nejdůležitějších opatření je prevence. Předcházet hrozbě dříve, než nastane. Dnes je 99 % informací dostupných na internetu. Jsou zde miliony terabajtů dat a mezi nimi i velmi nebezpečné. Pravděpodobnost stažení škodlivých dat je dnes velmi vysoká. A přesto největší hrozbu představují lidé. Neopatrnost a nedodržovaní bezpečnostních opatření může způsobit velké škody.

Každá osoba, každý zaměstnanec vlastní mobilní telefon, externí úložiště pro svá data. Tato zařízení mohou představovat potencionální možnost pro útočníka jako prostředek pro přenos škodlivých dat. Aby se předešlo nežádoucímu vniknutí do chráněného systému, od osobního počítače po firemní síť, je třeba takové zařízení kontrolovat.

V této bakalářské práci je realizován návrh prostředí pro analýzu podezřelého zařízení. Cílem práce je navrhnout takové zařízení které bude schopné jak povrchové, tak i hlubší analýzy. Toto zařízení musí představovat levné a efektivní hardwarové řešení, které s využitím open source programů analyzuje obsah připojeného zařízení, a to bezpečně v kontrolovaném prostředí. Zařízení obsahuje jednoduché i profesionální programy, které jsou dále využity pro ukázku analýzy flash disku a mobilního zařízení. V této práci jsou provedeny celkem tři analýzy. První je provedena na flash disku nakaženého škodlivým kódem. Další dvě jsou provedeny na mobilním zařízení, kdy v jednom případě mobilní telefon je nakažený škodlivým kódem a ve druhém není. Analýza je zaměřena na mobilní zařízení se systémem Android, a to z toho důvodu, že je v dnešní době nejrozšířenějším mobilním systémem a stále jsou používány starší verze tohoto systému, které již neposkytují dostatečné zabezpečení.

2 Definice pojmů

- **Aktivum** je cokoliv, co je potřeba chránit a co má pro majitele nějakou hodnotu. Například hardware, software, osobní data, služby atd.
- **Útočník** je osoba, která se úmyslně snaží zmocnit, poškodit nebo zneužít aktiva někoho jiného.
- **Hrozbou** se rozumí příčina jakékoliv události, ať už žádoucí či nežádoucí, při které by mohlo dojít k poškození nebo ztrátě aktiv, nedostupnosti systému nebo služeb nebo nežádoucímu vniknutí neoprávněné osoby.
- **Ochrana** představuje jakékoliv opatření, které slouží ke snížení rizika ztráty aktiv nebo jiným nežádoucím akcím.
- **Bezpečností** rozumíme stav, kdy jsou do jisté míry aktiva chráněna před hrozbou. Do jisté míry, protože bezpečnost nemůže být nikdy absolutní.
- **Zabezpečení**, jde o systém ochran, který je určen k efektivní ochraně aktiv.
- **Slabinou** nazýváme zranitelné místo, které by mohlo být útočníkem využito pro útok. Toto místo však nyní nelze zabezpečit ať už z důvodu technické náročnosti nebo vysokých nákladů.
- **Riziko** znamená, s jakou pravděpodobností muže dojít k využití zranitelného místa.
- **Incident** může nastat, pokud bude uskutečněna jakákoliv nežádoucí hrozba.
- **Průnik/dopad** znamená, že z důsledku útoku došlo ke ztrátě aktiv nebo jiným škodám.
- **Malware** je jakýkoliv nebezpečný nebo nežádoucí program, který může ohrozit počítač.
- **Sandbox** označuje uzavřené, oddělené a kontrolované hardwarové a softwarové prostředí tak aby žádný nebezpečný malware nebo jiný software nemohl způsobit škodu mimo toto prostředí. [1]
- **Penetrační testování** je dovolené a úmyslné napadání testovaného systému za účelem nalezení slabého místa a předejití tak úspěšnému skutečnému útoku.
- **Open source** je software s otevřeným zdrojovým kódem. Software, který je volně dostupný pod určitým typem licence. Tato licence určuje práva, jakým způsobem může uživatel s daným softwarem nakládat.
- **Virtualizace** je vytvoření virtuálního, zdánlivého počítače, uvnitř skutečného. Pomocí vhodného softwaru lze virtualizovat jednotlivé aplikace i celý počítač.
- **Hypervisor** je řídící prvek, který umožňuje řídit přístup virtualizovaným počítačům k hardwaru. Řídí jejich běh a zároveň je od sebe odděluje. Umožňuje na jednom počítači spustit zároveňn více operačních systémů.

3 Sandbox prostředí

V následující kapitole je vysvětlen pojmem sandbox, k čemu se používá a jakých využívá technik.

V informatice se pojem sandbox používá ve spojení s izolovaným prostředím, ve kterém může být program nebo soubor proveden bez ovlivnění prostředí ve kterém běží. Je tedy velmi využíván vývojáři nebo testery softwaru. Nový a neotestovaný software by mohl způsobit nevratné škody, stejně tak podezřelý soubor nebo program, který je potřeba analyzovat. Veškeré instrukce a práce s programy nebo soubory jsou prováděny v karanténě, ze které nesmí nic uniknout ven. Software nacházející se v sandboxu, má přesně vymezený prostor pro jeho hladký běh, ale zároveň omezené prostředky, ke kterým má přístup. Toto kontrolované prostředí umožňuje malwaru provést všechny jeho škodlivé operace. Program pak zaznamenává jeho chovaní a vyhodnocuje, zda je nebo není soubor škodlivý. Vzhledem k vyhodnocovaní programu jako originálního celku je možné analyzovat a ohodnotit program jako škodlivý, i když se s ním doposud ještě nikdo nesetkal.

Sandbox pro analýzu škodlivého softwaru záměrně spouští škodlivý kód, aby jej mohli analyzovat. Tyto softwary nám umožnují bezpečnou kontrolu externích zařízení (USB flash disk, externí hard disk aj.). Zjistit, zda se na nich nenachází škodlivý software a předejít tak možným škodám. Sandbox se také velmi často používá jako bezpečnostní řešení koncového bodu sítě jako poslední ochranná linie. Informace v této kapitole byly čerpány z [2] a [3].

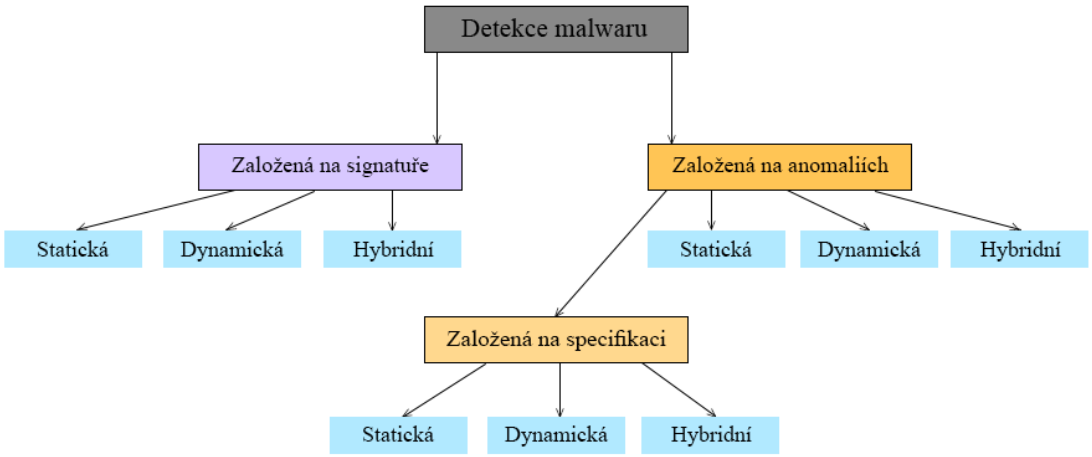
3.1 Detektor malwaru

Někdy je systém pro detekci škodlivého softwaru nahrazován synonymem detektor malwaru, obvykle se však jedná o součást tohoto systému. Detektor malwaru může nebo nemusí mít stejný systém, který se snaží chránit. Detektor malwaru implementuje detekční techniky, které zjišťují škodlivé chování a tím se nás snaží chránit. Tyto prokázané techniky slouží také jako měřítko schopnosti detekovat škodlivý software. Malware detektory mají dva vstupy. První je jeho znalost škodlivého chování. Detektor se musí naučit rozpoznávat také anomální chování, to vychází ze znalostí o tom, co je chování „normální“. Rozpoznávání normálního chování škodlivého softwaru, tedy škodlivého softwaru, který je znám a ví se o jeho škodlivé činnosti, vychází z tzv. repozitářů nebo knihoven, které jsou obvykle aktualizovány lidmi, kteří jsou schopni škodlivé chování malwaru vyjádřit v podobě, která je strojově čitelná. Detektor má tedy k dispozici tyto vědomosti jako šablonu normálního chování na základě kterých může vyhodnocovat chování anomální. Druhým vstupem jsou data, která mají být analyzována. Jakmile má detektor znalosti a data která jsou předmětem

kontroly, může využít svých detekčních technik a rozhodnout, zda jsou data škodlivá či nikoliv.

3.2 Techniky detekce malwaru

Tyto techniky lze rozdělit do dvou kategorií. První kategorií je detekce založená na anomáliích a druhá detekce je založená na signatuře. Detekce založená na anomáliích vychází z již víše zmíněné znalosti o tom, co je to „normální“ chování. Detektor na základě svých znalostí, tedy pravidlech, vyhodnocuje a rozhoduje o škodlivosti dat. Pokud zkoumaná data pravidla porušují, jsou považována za anomální a obvykle škodlivé. Detekce založená na signatuře rozhoduje o škodlivosti dat také na základě toho, co je známo. Obr. 3.1 nám znázorňuje vztah mezi různými typy technik detekce malwaru. Každá ze tří technik může využít tří různých přístupů: statický, dynamický a hybridní. Analýza anomálií nebo specifikační přístup nebo přístup na základě signatury. Tyto techniky jsou určeny tím, jak získávají informace o malwaru. Statický přístup využívá informace k určení škodlivosti ze syntaxe nebo ze struktu-



Obr. 3.1: Klasifikace detekčních technik malwaru.

rálních vlastností analyzovaného programu. Tím může být například sekvence bajtů. Obecně platí, že se statický přístup pokouší detekovat malware ještě před jeho spuštěním. Dynamický přístup využívá naopak vlastnosti procesu, jako je informace o runtime neboli běhové knihovně. Z toho vyplívá, že se dynamický přístup pokouší detekovat malware po jeho spuštění nebo při jeho provádění. Hybridní přístup je kombinací statického a dynamického přístupu. Informace v kapitole 3.1 a 3.2 byly čerpány z [4].

4 Moderní malware a databázové uložiště

Malware neboli škodlivý software, který při svém spuštění zahájí škodlivou činnost, která může způsobit poškození systému nebo ztrátu aktiv ze systému ve kterém se právě nachází. Spuštění škodlivého softwaru může být načasované nebo může nastat až ve chvíli kdy uživatel uskuteční určitou událost v systému na kterou je malware naprogramován a čeká na ni (např. spuštění souboru spojeného s malwarem) [5].

4.1 Typy malwaru

- **Infoware** je aplikace, která se používá ve válečných konfliktech pro získání výhody nad protivníkem. Jedná se o metody, které slouží k ochraně, poškození a zničení informací nebo informačních zdrojů.
- **Spyware** neboli špionážní software, se kterým se mohou setkat nejen běžní uživatelé, ale i firmy na které může například útočit konkurenční firma za účelem získání důvěrných informací o obchodování firmy. Jedná se o software, který průběžně monitoruje určitého uživatele nebo skupinu uživatelů a zjištěné informace zasílá útočníkovi nebo osobě která chce informace získat. Takovéto softwary jsou většinou na cílovém systému nainstalovány nevědomky, díky neopatrnosti uživatelů nebo zaměstnanců. Mohou se například nacházet na flash disku, který budoucí oběť nalezla na veřejném prostranství.
- **Adware** je software používající se pro předání reklamního sdělení uživateli, i proti jeho vůli. Nejznámější typem jsou vyskakovací reklamy na webových stránkách. Ty jsou většinou dovoleny vlastníky webových stránek za účelem generování příjmu. Samotný adware je sám o sobě ve většině případů neškodný, mohou však nastat situace kdy pracuje společně se spywarem. Informace pro infoware, spyware a adware byly čerpány z [5].
- **Bots** jsou softwarové programy, které jsou vytvořeny za účelem provádění určitých operací. Můžeme se s nimi běžně setkat v bezpečné formě jako například v počítačových hrách, internetových aukcích a podobně. Mohou však být i zneužity pro vytváření škodlivé činnosti. Boti se dají použít pro tzv. botnety což jsou sbírky počítačů ovládané třetími stranami a určené k DDoS útokům nebo pro vytváření reklam na webových stránkách. Ty se mohou bránit například pomocí testů CAPTCHA, které ověřují, zda se jedná o lidského uživatele a ne o robota.
- **Ransomware** je typ malwaru, který slouží k tzv. zajetí počítače. Uživateli je omezen přístup k počítači například zašifrováním veškerých souborů na pevném disku nebo uzamčením systému. Uživateli je dále zobrazena zpráva, která ho má donutit zaplatit za znovu získání přístupu do svého počítače.

Ramsoware se většinou šíří jako obvyklý virus. Může se jednat o zranitelnost v síti nebo stahováním nedůvěryhodných souborů.

- **Rootkit** je software, který je určen k vzdálenému přístupu nebo řízení počítače, aniž by byl uživatelem nebo bezpečnostním softwarem detekován. Rootkit se neustále snaží upravovat programy nebo registry, zejména prvky, které by pomohli uživateli detekovat škodlivý software. Může se jednat o skrývání adresářů, registrů nebo procesů. Díky tomu může být velmi obtížné se jej zbavit. Detekce rootkitu pak spíše závisí na ručních metodách jako je sledování chování počítače, jeho pomalý chod nebo využití procesoru. Rootkit může také spouštět soubory, získávat informace nebo instalovat skrytý malware.
- **Trojský kůň** je typ malwaru, který se maskuje jako normální soubor. Uživatel si tak myslí, že stahuje soubor, který chce. Pokud není trojský kůň detekován bezpečnostním softwarem, muže si tak uživatel nevědomky infikovat počítač. Tento škodlivý software pak poskytne útočníkovi vzdálený přístup do počítače, který může ukrást důvěryhodná data jako přístupové údaje, hesla apod.
- **Virus** se sám kopíruje a šíří se tak do jiných počítačů. To může být nastartováno tím, že uživatel spustí program, na který je virus navázán. Mohou se však také šířit pomocí souborových skriptů nebo dokumentů. Slouží k poškození hostitelských počítačů, k odcizení informací nebo vytváření botnetů.
- **Počítačové červi** jsou nejběžnější typ malwaru. Mohlo by se zdát, že jde o ten samý typ jako u počítačových virů. Rozdílem je však, že viry jsou závislé na uživatelské činnosti, tzn. začnou se šířit až poté co uživatel spustí určitý program. Červi se mohou kopírovat a šířit samostatně. Způsobují odcizení dat, odstraňovaní souborů nebo vytváření botnetů. Červ se nejčastěji šíří hromadnými emaily s infikovanými přílohami.

Škodlivé softwary se od sebe velmi liší, jak infikují počítač nebo jak se mohou šířit. Mohou však způsobovat stejné příznaky podle kterých lze zjistit, zda byl daný počítat infikován. Jedná se o příznaky:

- Problémy s připojením k síti.
- Upravené nebo smazané soubory.
- Vyšší využití procesoru.
- Pomalý počítač nebo rychlost webového prohlížeče.
- Podivný vzhled souborů, programů nebo ikon na ploše.
- E-maily nebo zprávy jsou odesílány automaticky bez interakce uživatele.

Další informace v kapitole Typy malwaru byly čerpány z [6].

4.2 Technologie MISP (Malware Information Sharing Platform)

Je otevřený software, který slouží pro shromažďování, ukládání a distribuci informací o bezpečnostních hrozbách. MISP je navržen pro bezpečnostní analytiky, profesionály v oboru informační bezpečnosti a malware analytiky, kteří zároveň podporují tento software a sdílejí skrze něj strukturované informace o každodenních operacích. Cílem MISP je sdílet právě tyto informace o škodlivém softwaru a bezpečnostních hrozbách v rámci bezpečnostní komunity.
Hlavní funkce MISP jsou:

- Efektivní databáze IoC (návrhový vzor – Inversion of Control) a indikátorů, které umožňují ukládat technické i netechnické informace o malwaru.
- Automatické zjištění a korelace vztahů mezi atributy a indikátory malwaru, mezi útokem a analýzou.
- Flexibilní datové modely, kde mohou být komplexní objekty vyjádřeny a propojeny společně a informovat tak objektivně o hrozbách a propojených funkcích.
- Vestavěné funkce sdílení usnadňují distribuci dat, MISP může automaticky provádět synchronizaci dat a událostí mezi různými MISP. Pokročilou funkci filtrování lze naopak využít ke splnění vyhledávaných informací. Informace lze flexibilně sdílet mezi různé organizace a skupiny lidí.
- Intuitivní uživatelské rozhraní pro koncové uživatele pro vytváření a práci s událostmi a grafické rozhraní pro bezproblémovou navigaci mezi událostmi a jejich korelací.
- Ukládání dat ve strukturovaném formátu, které mohou sloužit pro automatické vytváření databází s rozsáhlou podporou ukazatelů jako například pro finanční sektor.
- Výstup informací v souboru v široké škále formátů, stejně tak vstup.
- Nástroj pro flexibilní vkládání volného textu pro usnadnění vkládání nestrukturovaných informací.
- Systém pro spolupráci s uživateli umožňující jim navrhnout nové změny a aktualizace.
- Sdílení a automatická výměna dat a jejich synchronizace mezi ostatními skupinami a organizacemi.
- Nastavení taxonomie pro klasifikaci a označení událostí podle vlastních nebo existujících klasifikačních schémat. Taxonomie může být lokální, ale také sdílená mezi ostatní MISP.
- Dále rozšiřující moduly v Pythonu, podpora STIX (Structured Threat Infor-

mation Expression) a mnoho dalších funkcí.

Výměna informací je důležitá a vede k rychlejšímu odhalení cílených útoků a přispívá ke snížení chybné analýzy. Díky informacím můžeme rozpoznat malware, který již analyzovala jiná skupina nebo organizace a vyvarovat se tak možné hrozbě, že analýza bude vyhodnocena chybně. V kapitole technologie MISP jsou informace čerpány z [7].

4.2.1 Funkce MISP Galaxy Cluster

Tato funkce MISP slouží k vyjádření velkého počtu objektů nebo knihoven, které se týkají velkého množství možných škodlivých technik nebo možných hrozeb. Obecně mohou tyto knihovny sloužit jako šablona, ale mohou být nadále upravovány a aktualizovány. Jedná se také o velkou sadu slov. Tento slovník například popisuje různé typy platforem, které čelí potencionálním hrozbám (např. Android nebo mobilní zařízení obecně) dále různé typy malwaru (např. Svpeng, LokiBot apod.).

Nalézt lze zde také knihovny, které se týkají bankovního sektoru, poskytují široký seznam malwaru, který je cílený na získání bankovních informací uživatelů. Dále sekce android, backdoor, botnets, ramsomware, mobile attack a mnoho dalších.

Níže je uveden příklad, jaké informace je možné získat o hrozbě z aktuálního slovníku MISP.

NjRAT

Tento typ trojského koně je cílený na útok na základě získání přístupu k vzdálenému počítači oběti. Má schopnost získat přístup ke kameře, ukrást data uložená v prohlížeči, spustit shell, nahrávat a stahovat soubory, prohlížet pracovní plochu a manipulovat s procesy, se soubory nebo registry. Dále umožnuje útočníkovi provádět aktualizace a odinstalovat programy, ukončovat je nebo restartovat počítač. Lze ho použít i k vytváření a konfiguraci šíření škodlivého softwaru prostřednictvím USB disků. Díky tomu tak rozšiřovat dopad svého útoku.

NjRAT je také znám jako Bladabindi. Tento malware využívá pro svůj útok .NET framework a skládá se zejména ze dvou důležitých tříd nazývané kl a OK.

Třída kl využívá funkce jako *GetKeyboardLayout*, *MapVirtualKey* apod. pro zachycení interakce mezi obětí a klávesnicí. Třída OK obsahuje ostatní funkce například pro modifikaci registrů, zjištění, zda infikovaný počítač obsahuje ve svém zařízení kameru nebo smazání dat týkajících se nakažení malwarem.

NjRAT je velmi populární zvláště díky jednoduchosti stáhnutí toho malwaru oběťmi. Na internetu je celá řada videí a webových stránek poskytující podrobné návody, jak tento malware používat. Jeho snadnost použití také vyplívá ze skutečnosti, že je pro jeho útok použita dynamická služba DNS (systém doménových jmen – Domain Name System), což znemožňuje sledovat domény a použité IP adresy.

To znamená, že vzorky vycházející z analýz tohoto malwaru, se těžko používají při dalších analýzách. Informace o NjRAT čerpány z [8].

Díky MISP Galaxy je možné získat široké povědomí a informace o aktuálních malwarech a škodlivých technikách. Poskytuje jednoduché a stručné vysvětlení s dalšími odkazy na zkoumanou problematiku. Více informací k této kapitole na [9].

4.2.2 Funkce MISP Taxonomie

Taxonomie vyjadřuje v kybernetické bezpečnosti v užším slova smyslu knihovnu všech doposud známých událostí, indikátorů a hrozeb. Tato technika je základním mechanizmem pro klasifikaci, označování a organizaci informací v MISP a pro jejich distribuci mezi skupinami a organizacemi. Uživatelé mohou využívat již existujících taxonomií nebo si vytvořit svoji vlastní. Tyto klasifikace jsou distribuovány jako jednoduché JSON (JavaScriptový objektový zápis – JavaScript Object Notation) soubory pro použití v MISP a mohou být jednoduše zavedeny i do jiných informačních softwarů.

Taxonomie umožňuje používat stejnou slovní zásobu mezi distribuovanou skupinou uživatelů nebo organizací. Zmíněný JSON soubor má přesnou podobu, která je strojově čitelná. Používají se tzv. strojové značky. Ty se skládají ze tří částí: jmenný prostor a predikát, tyto dvě části jsou povinné. Třetí částí je hodnota, která je nepovinná. Strojové značky jsou někdy označovány jako trojité značení.

Příklad strojové a pro lidi čitelné značky:

ms-caro-malware-full : malware-family ="DoubleD"

2009 - an adware program that displays pop-up advertising, runs at each system start and is installed as an Internet Explorer toolbar.

- jmenný prostor
- predikát
- hodnota

Informace o funkci MISP Taxonoime čespány z [10]

4.3 Obranné techniky škodlivého kódu

Škodlivý software využívá různých technik, jak předejít odhalení nebo jak skrýt svoji škodlivou činnost. Sandbox evasion neboli „vyhýbání se“ a sandbox escape neboli „uniknutí“ karanténě.

4.3.1 Evazivní techniky proti odhalení

Již více než před deseti lety, nacházeli tvůrci škodlivých softwaru způsoby, jak obcházet programy založené na statické analýze, jako jsou běžné antivirové programy. Využívali běžných technik jako je polymorfismus, metamorfismus, šifrování, ochranu před reverzí aj. Dnes lze tzv. vyhýbání se sandboxu rozdělit do tří skupin.

- **Detekce sandboxu** – malware se snaží zjistit přítomnost sandboxu technikami, které hledají malé odlišnosti mezi sandboxem a reálným systémem oběti. Pokud malware zaznamená přítomnost sandboxu, tak reaguje ve většině případech jednou ze dvou možností. Ta první je, že se okamžitě ukončí tzn. ukončí veškerou svoji činnost. To je už samo o sobě podezřelé. Druhou možností je, že neprovádí podezřelé operace, ale jen operace, které analýza nevyhodnocuje jako podezřelé. Může však také ukázat falešnou chybovou zprávu, že je spustitelný soubor poškozen nebo určitý systémový modul chybí. Podrobnější popis technik, které malware využívá k tomu, aby zjistil, zda se právě nachází v sandboxu:
  - **Detekce virtualizace** – jedná se o metodu spíše zastaralou, dnes nejsou virtuální stoje jen pro výzkumníky nebo pro analytiky, ale jsou běžně používány na pracovních stanicích nebo serverech. Stále však existuje malware, který tuto techniku využívá, avšak není příliš účinná. Dnes je již vysoká podpora hardwarové virtualizace a tím pádem se i snižují viditelné odlišnosti které by malware mohl zaznamenat pro odhalení, že se nachází v hypervisoru. Co je však v dnešní době stále relevantní je odhalení vlastností, které jsou nějakým způsobem spojeny s výrobcem. Například MAC adresa (jednoznačný identifikátor síťového zařízení – Media Access Control), ID zařízení, existence určitých procesů nebo ovladačů, klíče registrů aj.
  - **Detekce samotného sandboxu** – v této metodě se malware nesnaží detekovat hypervisor, ale samotný sandbox. K tomu může opět využít znalostí o vlastnostech specifických pro dodavatele softwaru nebo detekovat určité technologie, které sandbox využívá. Většina prostředí pro analýzu využívá tzv. háčků. Hák je vlastně taková mezi vrstva, která zachycuje komunikaci mezi procesy, operačním systémem a ovladači. Jiný sandbox

zase využívá tzv. emulaci. Ta způsobuje rozdíly mezi emulovaným prostředím a skutečným systémem. Malware může například vyvolat instrukci CPU (centrální procesorová jednotka – Central Processing Unit) a pokud se volání nezdaří, malware zjistí, že se nachází v emulovaném prostředí.

  - **Detekce umělého prostředí** – sandboxi jsou speciálně určeny například pro analýzu malwaru nebo forenzní analýzu, a tak jsou odlišná od skutečných systémů. Tyto rozdíly pak může malware detekovat. Jedná se například o vlastnosti hardwaru, kdy mohou chybět některé ovladače například pro USB 3.0, malá velikost pevného disku nebo velikost paměti, malé rozlišení obrazovky. Vlastnosti softwaru, kde není žádný poštovní klient. Vlastnosti systému jako sítový provoz. Vlastnosti uživatele jako žádné soubory cookies, žádné soubory uživatelů nebo čistý souborový systém.
  - **Detekce na základě časování** – malware může měřit čas, který je potřebný pro nějaký proces, může časy porovnávat a zjistit tak určité zpoždění ke kterému při virtualizaci může docházet. Sandbox může tento čas předstírat tím, že mění jeho hodnoty. To však může malware obejít například začleněním externího zdroje času jako je NTP (protokol pro synchronizaci vnitřních hodin počítačů po paketové síti – Network Time Protocol).

  I když existují nedetekovatelné sandboxi, malware stále nemusí vykazovat své skutečné chování, pokud nejsou splněny jiné podmínky. Malware může být velmi citlivý na to v jakém prostředí se nachází. Může být určen pro útok na specifický systém a pokud prostředí ve kterém je malware analyzován neodpovídá těmto specifikacím, nezačne provádět svoji skutečnou škodlivou činnost. Proto existují sandboxi které spolehlivě detekují malware, který je citlivý na prostředí a automaticky určit potřebná nastavení prostředí a upravit se tak podle potřeb pro správnou analýzu. Pro správné nastavení prostředí pro analýzu podezřelého zařízení se používá tzv. Paranoidní ryba (Pafish). Jde o nástroj, který testuje detekovatelnost sandboxu. Využívá k tomu nepřeberné množství metod jako například pozice kurzoru, jméno uživatele přidruženého k aktuálnímu procesu, úplné cesty k souborům, velikost fyzických jednotek, kontrola registrů a další. Tyto testy jsou následně použity pro správné nastavení sandboxu, aby se předešlo odhalení malwarem a ten tak mohl začít vykazovat svoji skutečnou škodlivou činnost. [11]

- **Slabiny sandboxu** – druhá metoda vyhýbání se sandboxu říká, že se malware přímo snaží odhalit slabá místa nebo technologické slabiny sandboxu skrze které by pak mohl zaútočit. Takovouto slabinou může být například využití skutečnosti, že sandbox neumí analyzovat určitý typ souborů, to znamená,

že malware bude v nějakém obskurním formátu, který sandbox nepodporuje jako například .hta nebo .dzip. Další možností může být neschopnost sandboxu analyzovat soubory, které mají příliš velkou velikost. Malware se taky může pokoušet vyhýbat analýze využíváním API (rozhraní pro programování aplikací – Application Programming Interface). Tato metoda je účinná v případech kdy sandbox využívá pro analýzu tzv. háčkování. Každopádně v tomto případě se malware nemusí starat o to, zda bude detekován nebo zda se nachází v prostředí pro analýzu. Podrobnější popis technik využívaných malwarem:

  - **Odstavení monitorovacího zařízení** – monitorování karantény, ve které je prováděna analýza a která přijímá kód nebo procesy od hostitelského sandboxu může být zaslepena. Pokud jsou tyto instrukce mezi hostitelem a karanténou zrušeny, sandbox tak nevidí do analyzovaného prostředí. Takovéto oslepení může nastat například odstraněním háčku, a to pomocí obnovení původní instrukce nebo dat. Dále obejitím háčku pomocích přímých systémových volání nebo neoprávněného použití API. Háčky lze najít v systémových souborech, které jsou dále mapovány do paměti. Ty může malware odinstalovat.
  - **Odstavení analyzujícího prostředí** – jde o hrubou metodu, která překonává analyzující prostředí tím, že využívá neschopnost karantén analyzovat soubory s větší velikostí nebo soubory které mají více vrstev komprese.

- **Využití spouštěčů** – ve třetí metodě se malware nesnaží zjistit přítomnost sandboxu nebo na něj nějak zaútočit. Využívá, ale některé nedostatky, které může takový automatizovaný software jako je sandbox mít. Dnes již existuje obrovské množství unikátních malwarů a vzhledem k tomu stráví sandbox analýzou každého souboru jen pár minut. To muže být malwarem využito k tomu, aby počkal se škodlivou činností a zůstal tak neodhalen. Kromě toho může malware využít také jiné operace které sandbox neanalyzuje nebo není proti nim ochráněn, jako například restart systému. Obecně lze tzv. spouštěče rozdělit do čtyř kategorii.
  - **Časové bomby** – jde o jednu z nejběžnějších technik. Karantény obvykle analyzují vzorky jen několik minut, a tak malware jen čeká a oddaluje svoji skutečnou činnost. Malware a sandbox hrají mezi sebou takovou hru kdy se jeden druhého snaží přelstít. Malware spí a čeká, sandbox se tento spánek snaží odhalit, a tak zkracuje čas, malware tento zkrácený čas detekuje, sandbox se snaží zakrýt reálný čas aktualizováním také systémového času atd.

    Použité techniky:

    * Od jednoduchý po velmi složitý spánek, kde jsou použita souběžná

vlákna, která na sebe dohlížejí nebo jsou na sobě navzájem závislá.
    * Provádění škodlivé činnosti jen v určitý čas a den, např. v úterý ve 12:00.
    * Zpomalit co nejvíce výkon analyzujícího prostředí, např. zasláním milionů systémových volání které však nemají žádný účinek kromě zpomalení spuštění malwaru v analyzujícím prostředí.
  - **Systémové události** – malware se stane aktivním pouze při vypnutí počítače, po restartování nebo pokud se nějaký uživatel přihlásí nebo odhlásí ze systému.
  - **Interakce uživatele** – malware čeká na pohyb myši nebo na vstup z klávesnice. Může se stát aktivní až po interakci s určitými aplikacemi jako je prohlížeč nebo Skype nebo v případě Office dokumentů pouze tehdy pokud si uživatel dokument prohlíží.
  - **Detekce specifického systému** – specifický typ malwaru může pracovat pouze na cílovém systému pro který je určen. Identifikace může probíhat například na základě uživatelského jména, časového pásma, rozložení klávesnice nebo IP adrese. Jde o jednoduché až po velmi složité metody identifikace. Jedná se v podstatě o inverzní metody detekce analyzujícího prostředí. Pokud jsou nedávno použité dokumenty prázdné nebo je využití sítě příliš nízké, malware spí.

Informace v kapitole 4.3.1 čerpány z [12]

4.3.2 Technika uniknutí odhalení

Technika uniknutí odhalení, je velmi složitá, ve většině případů se jedná o konkrétní příklad použití. Technicky popsané metody jsou pro každou metodu použití jiné, ale cíl mají stejný. Hlavní myšlenkou je použití k útoku malware, který se pokouší spouštět libovolný kód. Na počítači se nachází účet, který využívá hlavní uživatel jako administrátorský. Existují však v zařízení ještě další účty nazývané SYSTEM a ADMINISTRATORS. Účet SYSTEM má vyšší oprávnění než administrátorský účet vytvořený uživatelem. Malware se snaží spustit libovolný kód ne jako admin, ale jako systém a získat tak vyšší oprávnění jako plný přístup ke všem souborům, registrům, bootloaderů, souborům chráněných hesel jako SAM (Sequence Alignment Map), kde jsou uložena hesla Windows nebo k hardwaru nižší úrovně a tím způsobit zmatek v systému tak, že skryje některé soubory.

5 Teorie analýzy - Mobilní zařízení

Druhá část realizace obsahuje již složitější případ analýzy podezřelého zařízení. Jedná se o mobilní zařízení s obsahem škodlivého softwaru. Ten má za úkol otevřít zadní vrátka do systému. Pomocí nichž je útočník schopen zmocnit se osobních údajů jako jsou kontakty, SMS zprávy, hesla apod.

5.1 Android - operační systém

V následující kapitole je popsán operační systém Android. Jeden z nejrozšířenějších operačních systémů používaných na mobilních a přenositelných zařízeních. Je založen na jádře Linuxu a je dostupný jako otevřený software. O jeho vývoj se stará společnost Google pod hlavičkou konsorcia firem Open Handset Alliance. Výrobci mobilních a přenosných zařízení mohou systém Android upravovat při dodržení určitých podmínek a ve většině případů vyvíjejí prostředí jejichž názvem systém Android doplňují, např. MIUI nebo Touchwiz.

5.1.1 Architektura systému Android

Architektura systému je rozdělena do 5 vrstev. Jádro operačního systému s vrstvou HAL, knihovny, Android runtime, Android application framework a aplikace.

- **Jádro operačního systému** je nejnižší vrstvou systému Android. Linuxové jádro zprostředkovává komunikaci mezi hardwarem a softwarem pomocí ovladačů zvuku, WIFI, USB, obrazovky a dalších. Dále poskytuje operace typu čti/zapiš do souboru, spusť aplikaci nebo otevři síťové spojení. Mobilní aplikace jsou otevřeny v sandboxu, což v tomto případě znamená, že operační systém pro každou aplikaci vyhradí nový proces s virtuálním Java strojem (JVM), který teprve interpretuje programový kód mobilní aplikace.
- **Vrstva HAL** je abstraktní vrstva hardwarového rozhraní, která umožňuje využívat hardware telefonu, například kameru, bez nutnosti znát detaily hardwaru a jeho obsluhy. HAL (hardwarová abstraktní vrstva – Hardware abstraction layer) vytváří rozhraní pro komunikaci vyšších vrstev s hardwarem, vývojáři tak díky HAL nemusí znát přesně hardwarové specifikace všech zařízení. Obsluha zařízení musí být rychlá, a proto je rozhraní HAL napsáno v jazyce C a C++.
- **Knihovny** poskytují přístup aplikacím k různým komponentám systému Android. Knihovny jsou napsány v C/C++ a jedná se o základní funkce systému. Open GL a SGL jsou knihovny pro práci s grafikou. Open GL pro 3D grafiku a SGL pro 2D Grafiku. Media Framework slouží k práci s mediálními soubory.

Obsahuje například kodeky pro různé formáty audia a videa. SQLite slouží pro ukládání a práci s daty. Webkit je open source vykreslovací jádro pro webový prohlížeč, FreeType se stará o vykreslování písma a SSL se stará o šifrování a zabezpečení přenosu dat.

- **Android runtime** obsahuje sadu základních knihoven. Jelikož aplikace pro Android jsou napsány v Javě a knihovny v C a C++, vznikl virtuální stroj DVM (Dalvik Virtual Machine) ve kterém aplikace jako samostatné procesy využívají vlastní instanci. Dalvik je optimalizovaný pro mobilní zařízení, a tak bere v úvahu omezené možnosti napájení, menší paměť apod. Od verze 4.4 je v systému použita dopředná kompilace. Java kompilátor přeloží soubory zdrojového kódu aplikace do více binárních souborů. Dále nástroj DX transformuje tyto binární soubory do jednoho souboru ve formátu DEX. Soubor je dále vykonán virtuálním strojem Dalvik. Výsledkem je pomalejší instalace aplikace, ale jelikož je jednou pro vždy aplikace zkompilovaná do nativního kódu v zařízení, je zařízení až dvakrát výkonnější s nižší spotřebou baterie.
- **Application framework** vrstva obsahuje důležité bloky jako je správce aktivit, správce hovorů, správce lokalizace, správce zdrojů apod. Tyto bloky jsou napsány v jazyce Java. Vrstva Application framework zprostředkovává komunikaci mezi aplikační vrstvou a vrstvou knihoven a virtuálním stojem Dalvik. Aplikace v systému android běží ve vlastním sandboxovém virtuálním stroji Dalvik a skládají se z různých komponent jako jsou aktivity, služby, příjem vysílání nebo poskytování obsahu. Tato vrstva umožňuje aplikacím využívat tyto služby a sdílet je mezi sebou.
- **Aplikace** jsou konečné aplikace využívané koncovým uživatelem. Může se jednat o aplikace systémové nebo uživatelské stažené.

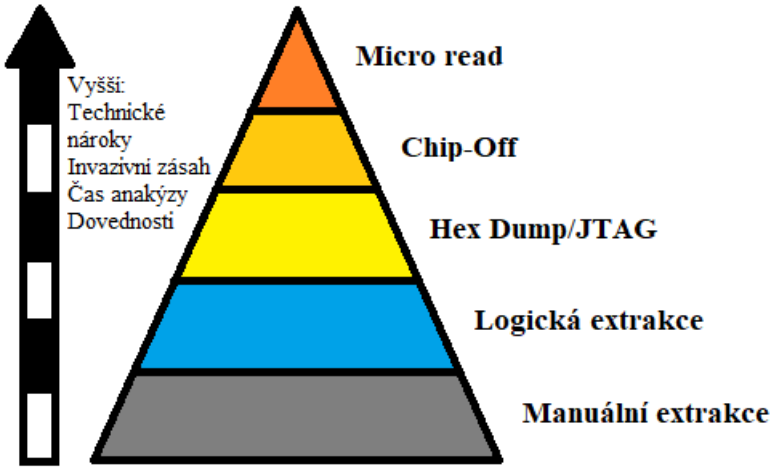
Informace v této kapitole čerpány z [13] prostředí GoogleScholar.

5.2 Typy analýz mobilního zařízení

Existuje několik způsobů, jak extrahovat data ze zařízení. Lze je rozdělit do dvou úrovní. Neinvazivní a invazivní metody, které se dále dělí podle způsobu provedení. Viz obr. č. 5.1.

5.2.1 Neinvazivní metody extrakce dat

Neinvazivní metody jsou použity v případech, kdy je potřeba odemknutí zámku mobilního zařízení nebo zámku SIM karty. Lze zde zařadit i aktualizaci operačního systému. Další použití je při zjištění, že se na zařízení nachází škodlivá aplikace a je



Obr. 5.1: Druhy extrakce dat z mobilního zařízení.

třeba ji analyzovat. Tato metoda není prakticky použitelná v případech kdy zařízení utrpělo vážné fyzické poškození.

- **Extrakce dat manuálně** - je provedena za pomocí dotykového displeje nebo klávesnice a informace objevené na mobilním zařízení jsou fotograficky dokumentována. Tento proces však neumožňuje získání smazaných dat. Na druhou stranu je jednoduchý a použitelný na každém mobilním zařízení.
- **Extrakce dat na logické úrovni** - tuto metodu extrakce dat lze provést pomocí nástroje Android Debug Bridge. Extrakce dat se provádí připojením mobilního zařízení přes kabel nebo bluetooth, dále lze provést zálohu pomocí příkazu adb backup, ta v sobě však neobsahuje důležitá data kontaktů, sms a podobně, neobsahuje však nainstalované aplikace. Další způsob je procházet adresářovou strukturu ručně a posílat příkazy na mobílní zařízení skrze ADB. Nevýhodou je provádění úkonů v právě běžící struktuře připojeného zařízení. Zasahuje se tak do integrity dat, jsou prováděny změny v informacích o datech nebo je mobilní zařízení obohaceno o nová data. Pokud však není důležité zachování původních dat, jde o nejpraktičtější metodu, jak získat citlivé informace nebo získat podezřelou aplikaci určenou k další analýze. Smazaná data lze touto metodou získat jen zřídka. Tato metoda je také použita v kapitole 8 při práci se souborovým systémem a získání souborů určených k analýze. Detailnější popis ADB v kapitole 6.3.3.
- **Metoda JTEG** - je neinvazivní forma fyzické extrakce, která může extrahovat data z mobilního zařízení i v případě, že je obtížné přistupovat k datům prostřednictvím softwarových cest. To z důvodu, že zařízení je poškozené, zamčené nebo šifrované. Zařízení však musí být alespoň částečně funkční. Proces zahrnuje připojení k testovacím přístupovým portům (TAP) na zařízení

a instruování procesoru k přenosu surových dat uložených na připojených paměťových čipech. Jedná se o standardní funkci, se kterou by se člověk mohl setkat v mnoha modelech mobilních telefonů. Digitální forenzní vyšetřovatelé se zajímají o JTAG, protože teoreticky umožňuje přímý přístup k paměti mobilního zařízení bez ohrožení nebo narušení dat. Navzdory tomu je to náročný a časově zdlouhavý postup, který vyžaduje předběžné znalosti (nejen model JTAG pro model zkoumaného telefonu, ale i to, jak nově vytvořit výsledný binární soubor složený z paměťových struktur telefonu).

- **Hex Dump** - je podobně jako JTAG další metodou pro fyzickou extrakci nezpracovaných informací uložených v paměti flash. Provádí se připojením forenzní pracovní stanice k zařízení a následným zavedením kódu nebo zavaděče do zařízení, přičemž každý z nich provede instrukce pro výpis paměti z telefonu do počítače. Výsledný obraz je v binárním formátu a vyžaduje, aby ho osoba, která má technické vzdělání, analyzovala. Touto metodou však lze získat velké množství dat a to i pokud byla smazána.

5.2.2 Invazivní metody extrakce dat

Tato metoda bývá časově náročnější a technicky složitější. V případech, kdy je zařízení zcela nefunkční z důvodu nějakého vážného poškození, je velmi pravděpodobné, že jediným způsobem, jak získat data ze zařízení, je ruční odstranění čipů a zobrazení paměti flash zařízení. I když je zařízení nebo předmět v dobrém stavu, okolnosti mohou vyžadovat, aby soudní znalec získal obsah čipu fyzicky.

- **Chip-off** - je proces, který odkazuje na získávání dat přímo z paměťového čipu mobilního zařízení. Po nutných přípravách je čip odpojen od zařízení a čtečka čipů nebo druhý telefon se používá k extrahování dat uložených v zařízení, které je předmětem vyšetřování. Je třeba poznamenat, že tato metoda je technicky náročná vzhledem k široké škále typů čipů, které existují na mobilním trhu. Proces vyžaduje velké znalosti a specifický hardware pro provádění pájení a ohřevu paměťového čipu. Bity a bajty nezpracovaných informací, které jsou načteny z paměti, musí být dále analyzovány, dekódovány a interpretovány. Dokonce i ta nejmenší chyba může vést k poškození paměťového čipu a způsobit tak, že data budou ztracena. V důsledku toho, je tato metoda doporučena, když je paměťový čip jediným nepoškozeným prvkem v mobilním zařízení.
- **Micro read** - tato metoda vyžaduje maximální úroveň odbornosti a je velmi nákladná a časově náročná. Je vyhrazena pro národní bezpečností krize a spočívá v manuálním pořizovaní pohledů přes čočky elektronového mikroskopu na paměťový čip.

Informace v této kapitole byly čerpány z [14].

6 Návrh hardwarového a softwarového prostředí

6.1 Platforma Raspberry Pi 3 Model B+

Platformou označujeme jak softwarovou, tak hardwarovou základnu, na které mohou pracovat programy nebo operační systémy. Platformou je i samotný operační systém jako například Linux, Windows nebo MacOS. Příkladem hardwarové platformy je základní deska Rasberry Pi.

Bez operačního systému se dnes již v podstatě neobejdeme. Operační systémy jsou hojně využívány jak běžnými uživateli, tak profesionály. Zatímco běžný uživatel může využívat operační systém jen pro domácí potřebu, profesionál jej může využívat například pro správu serverů, analýzu dat nebo pro penetrační testování. Tím upřednostňujeme systémy, které jsou pro náš účel a pro naši potřebu nejoptimálnější. Windows je dnes nejpoužívanějším operačním systémem pro běžné uživatele, naproti tomu takový Linux je nejčastěji používán správci serveru.

Existují však i operační systémy a softwary, které jsou přímo určené k forenzní analýze dat nebo zkoumání, zda data nejsou škodlivá, tzn. zda se nejedná o malware. A právě tyto systémy jsou pro nás důležité. Díky jejich struktuře a možnosti virtualizace těchto systémů se při jejich používání a analyzování dat nevystavujeme nebezpečí vniknutí škodlivého programu do nežádoucích míst, kde by mohl učinit nějaké škody. Díky analýze tak preventivně předcházíme možnosti útoku a případné ztrátě, odcizení nebo poškození aktiv. Řada těchto nástrojů jsou nabízeny jako samostatný operační systém s Linuxovým základem. Lze je však najít i jako samostatné nástroje, které lze nainstalovat do operačních systémů.

V praktické časti, bude pro forenzní nástroje použita platforma Rasberry Pi 3 Model B+. Rasberry Pi je základní deska o velikosti, řekněme platební karty. Byla vytvořena za účelem vyučovaní ve školách jako prostředek pro ovládání jiných elektronických zařízení. Rychle si však našla oblibu i mimo školství například mezi nadšenci do elektroniky. A to zejména díky své praktičnosti, jednoduchosti a ceně. Rasberry Pi není tak výkonný jako běžný notebook nebo stolní počítač, a proto se skvěle hodí tam kde není zapotřebí vysokého výkonu a je kladen ohled na nízkou úroveň spotřeby. Naproti tomu disponuje harwarem, který například u Arduina nenajdeme a který poskytuje jen základní mikrokontrolér.

Rasberry Pi je otevřený hardware s výjimkou hlavního čipu, který je chráněn ochranou známkou a který se stará o provoz dalších součástí desky jako grafiky, CPU, paměti nebo řadiče USB. Za pomoci Rasberry Pi vzniklo mnoho zajímavých projektů, které jsou velmi dobře zdokumentované, a tak si můžeme spoustu věcí

Obr. 6.1: Platforma Raspberry Pi 3 Model B+ [16]

budovat nebo upravovat sami.

Rasberry Pi nabízí řadu modelů od nejlevnějšího Rasberry Pi Zero po nejnovější Raserry Pi 3 Model B+. Jenž bude v tomto projektu použit jako základní hardware pro analýzu podezřelého zařízení. Více informací o platformě lze nalézt v [15].

Technické parametry:

- Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit SoC @ 1.4GHz.
- 1GB LPDDR2 SDRAM.
- 2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2, BLE.
- Gigabit Ethernet přes USB 2.0 (maximální propustnost 300 Mbps).
- Rozšíření přes 40-pin GPIO.
- HDMI.
- 4 USB 2.0 porty.
- CSI kamera port pro připojení Raspberry Pi kamery.
- DSI displej port pro připojení Raspberry Pi displeje.
- 4 pólový stereo výstup a port pro kompozitní video.
- Mikro SD port pro nahrávání operačního systému a uložení dat.
- 5 V/2.5 A DC napájecí vstup.
- Power-over-Ethernet (PoE) (vyžaduje samostatnou PoE HAT).

Technické parametry byly čerpány z [17].

6.2 Hardware Raspberry Pi

V ukázce analýzy mobilního telefonu, provádí zařízení Raspberry Pi již náročnější analýzu. V této kapitole jsou uvedeny nové hardwarové a softwarové prvky, které mají za úkol rozšířit možnosti a schopnosti vyšetřovatele při provádění analýzy.

6.2.1 Rožšíření o HDD/SSD

Jedním z nových hardwarových prvků Raspberry Pi je zavádění systému Raspbian z SSD nebo HDD. Hlavní výhodou je zvýšení rychlosti zápisu a zvládání vyššího výkonu zapisování oproti SD kartě. Dále zvětšení datového prostoru. To je důležité z důvodu, že dnešní moderní zařízení mají velikost paměti i v řádech stovek gigabajtů dat. Při analýze je proto potřeba mít dostatečně velký prostor pro vytvoření obrazu analyzovaného zařízení. Přináší to však i drobné nevýhody jako větší napájecí zátěž samotného zařízení nebo možné přehřívání.

Samotná realizace zavádění systému z externího disku může být poněkud složitá. První možnost je zavést systém Raspbian na disk stejným způsobem jako na SD kartu. Dále přepsat konfigurační soubor systému na SD kartě config.txt. Do něj je potřeba přidat řádek textu `program_usb_boot_mode=1`, ten říká, aby bootování systému bylo provedeno z uložiště připojeného pomocí UBS. Tento způsob samotný funguje však jen jednou, při prvním spuštění. Po znovu zapnutí Raspberry Pi nebo při restartu se systém nezavede.

Důvodů proč rozšíření zařízení o externí pevný disk nemusí fungovat, může být mnoho. Je důležité vědět, že pokud použijeme pevný disk o velikosti 2,5 palce bude stačit pro jeho napájení připojení USB. Raspberry Pi však musí být napájeno originálním adaptérem nebo jiným s výstupním napětím alespoň 2,5 A. V případě disku o velikosti 3,5 palce je zapotřebí jeho externí napájení. Problém se zavedením systému může být způsoben i typem použitého disku. Jeho výměna je však vhodná až po vyzkoušení všech možností.

Jako první ověřená možnost zavádění systému z externího pevného disku je použití SD karty a externího disku zároveň. Raspberry Pi při startu přečte soubor cmdline.txt umístěný na SD kartě. Ten říká, aby zavádění systému probíhalo z externího disku. Je proto nutné mít SD kartu vždy při spuštění zapojenou. Systém pak nadále běží na externím disku stejně tak veškeré operace. Při restartu nebo znovu spuštění Raspberry Pi se však již nemůže stát, že systém nebude zaveden. Důvodem absence bližšího popisu je, že na internetu lze nalézt spoustu návodů.

V této práci je však zvolena druhá ověřená varianta, a to použití převodníku z USB 3.0 na 2x SATA II s externím napájecím kabelem. Tímto způsobem lze na disk zavést systém běžným způsobem a Raspberry Pi bude fungovat i bez SD karty a

dalšího zdlouhavého konfigurování. Výhodou při pořízení převodníku se dvěma porty pro připojení disků je možné připojení dalšího disku určeného například pro analýzu a není tak nutné obsazovat USB porty na Raspberry Pi.

6.2.2 Chlazení a šasi základní desky

S vyššími nároky na výkon přichází i problém s přehříváním, a to i z toho důvodu, že Raspberry Pi není navrženo na zvládání vyšší teplotní zátěže. Je prakticky bez chlazení. Na internetu lze zakoupit pasivní, aktivní, ale také vodní chlazení, to ovšem není nutně nezbytné. Aby byla zachována praktická i cenová efektivita, je pro tento projekt vytvořen hardwarový díl, který lze vytisknout na 3D tiskárně. Díky němu je možné mít Raspberry Pi i externí disk pohromadě s rozšířením o aktivní chlazení. To lze připojit na dva ze 40 pinů, kterými Raspberry Pi disponuje pro rozšíření o další hardwarové prvky.

6.3 Software Raspberry Pi

S vyššími nároky na možnosti analýzy různých zařízení (mobilní, diskové apod.) je Raspberry Pi rozšířeno o forenzní programy. Opět jsou využity open source varianty, avšak výběr je dosti omezený. Velké firmy zabývající se vývojem forenzního softwaru a hardwaru přicházejí s opravdu účinnými, avšak drahými nástroji. Práce s open source variantami je proto značně omezená.

6.3.1 Raspbian - operační systém

Platformu Raspberry Pi lze zakoupit s již předinstalovaným operačním systémem Raspbian, který se nachází na přiložené SD kartě. Raspberry Pi je vhodný a praktický nástroj pro kontrolu podezřelého zařízení, hlavně díky své jednoduchosti a velikosti. Zároveň je tento novější model univerzálnější v rámci možných použitelných operačních systémů. Díky architektuře procesoru ARM lze použít jen vybrané operační systémy. Již předinstalovaný Rasbian je vhodným operačním systémem pro vytvoření prostředí pro analýzu. Je možné do systému naistalovat rozšiřující nástroje, které podporují architekturu ARM. Díky experimentálním a testovacím distribucím Linuxu, jako jsou například Kali nebo Parrot můžeme převzít některé nástroje, které jsou použitelné i pro operační systém Raspbian.

6.3.2 Platforma Autopsy

Jedná se o platformu, která poskytuje grafické rozhraní pro knihovnu a sbírku nástrojů Sleuth Kit a dalším forenzním nástrojům. Je využíván forenzními vyšetřova-

teli, armádou nebo podnikovými vyšetřovateli pro efektivní analýzu pevných disků nebo chytrých telefonů. Díky své architektuře umožňuje uživateli rozšiřovat platformu o doplňkové moduly nebo si vytvářet své vlastní. Sleuth Kit je samotná sada nástrojů, která umožňuje analyzovat disk nebo obnovovat data z něj. Právě ona je použita v pozadí programu Autopsy a mnoha dalších forenzních nástrojů.

Instalace

Před instalací samotného Autopsy je třeba aktualizace Raspbianu.

~#apt-get install update
~#apt-get istall upgrade

Pomocí příkazu se nainstaluje nástroj Autopsy.

~#apt-get install autopsy

Pro spuštění Autopsy pak.

~#autopsy

Tento příkaz poskytne odkaz na webové rozhraní Autopsy, na který lze pomocí webového prohlížeče přistoupit. V případě zobrazení chybové hlášení, je třeba spustit vykonání operace s oprávněními uživatele root.
#sudo autopsy

Při úspěšném spuštění je třeba ponechat proces spuštěný na pozadí.

Do webového prohlížeče se zkopíruje adresa, která je zobrazena ve výpise spuštěného okna Autopsy. Tím se získá přístup do grafického rozhraní.

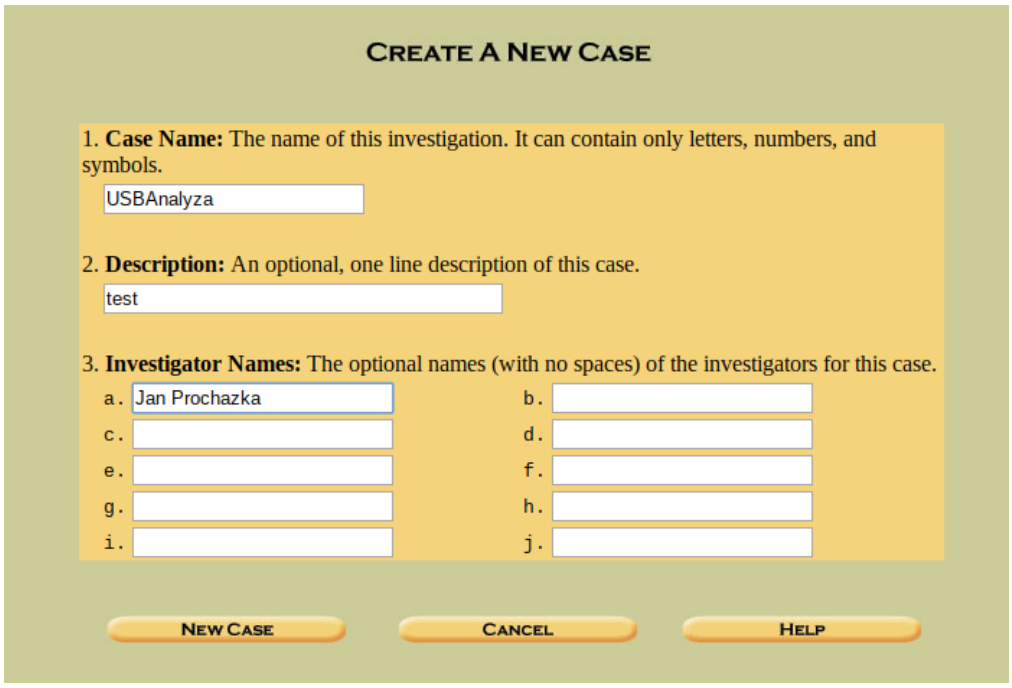
http://localhost:9999/autopsy

První spuštění

Před samotnou analýzou podezřelého zařízení je třeba vyplnit základní informace a nastavení nástroje Autopsy. Dále bude uveden postup tohoto nastavení a příklad analýzy podezřelého zařízení, na kterém se nachází škodlivý soubor. Vytvoření škodlivého souboru v kapitole 7.1.

V případě varovaní o spuštění Java Script podpory ve webovém prohlížeči je třeba z bezpečnostních důvodů tuto funkci vypnout. Raspbian požívá webový prohlížeč

Google Chrome. Návod pro vypnutí v následujícím odkazu:
https://www.computerhope.com/issues/ch000891.htm
Po vypnutí se pomocí tlačítka „New Case“ vytvoří nový případ analýzy. Na obr. 6.2 je vyplněn příklad vytvoření případu obsahující název případu, popis a jméno vyšetřujícího. Po vytvoření nového případu se zobrazí stránka s cestami k adresáři a

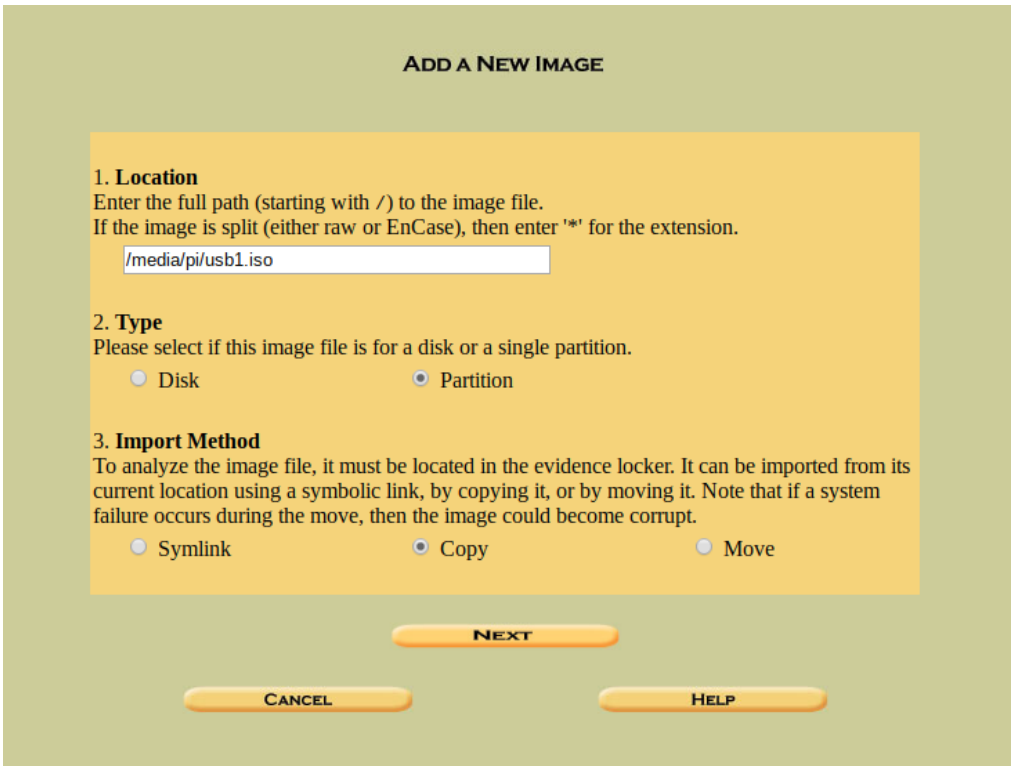


Obr. 6.2: Příklad vytvoření případu.

konfiguračního souboru. Na této straně se může nebo nemusí zobrazit možnost pro vytvoření hostitele. Pokud se nezobrazí je třeba stránku aktualizovat a přesunout se do menu kde se pomocí volby a tlačítka „OK“ vybere případ. V tomto příkladu případ „USBAnalyza“.

Následuje vytvoření hostitele pomocí tlačítka „Add Host“. V nastavení stačí vyplnit název vyšetřovaného zařízení a nepovinný popis zařízení. Opět bude vytvořen adresář a konfigurační soubor k příslušnému hostiteli. Následuje přidání obrazu disku zařízení, které bude analyzováno. Vytvoření obrazu disku v kapitole 7.2. Po otevření případu pomocí tlačítka „Add Image File“ přidáme obraz disku viz obr. 6.3. V nastavení přidávání se vyplní cesta k iso souboru, typ a nahrávací metoda. Jakmile je obraz disku nahrán, spustí se pomocí tlačítka „Analyze“ analýza obrazu. Ta otevře různé módy práce s obrazem. Viz obr. 6.4. Stručný popis módů:

- **File Analysis** – pro zobrazení a procházení souborů a adresářů.
- **Keyword Search** – pro prohledávání obrazu na základě zadaného obsahu.



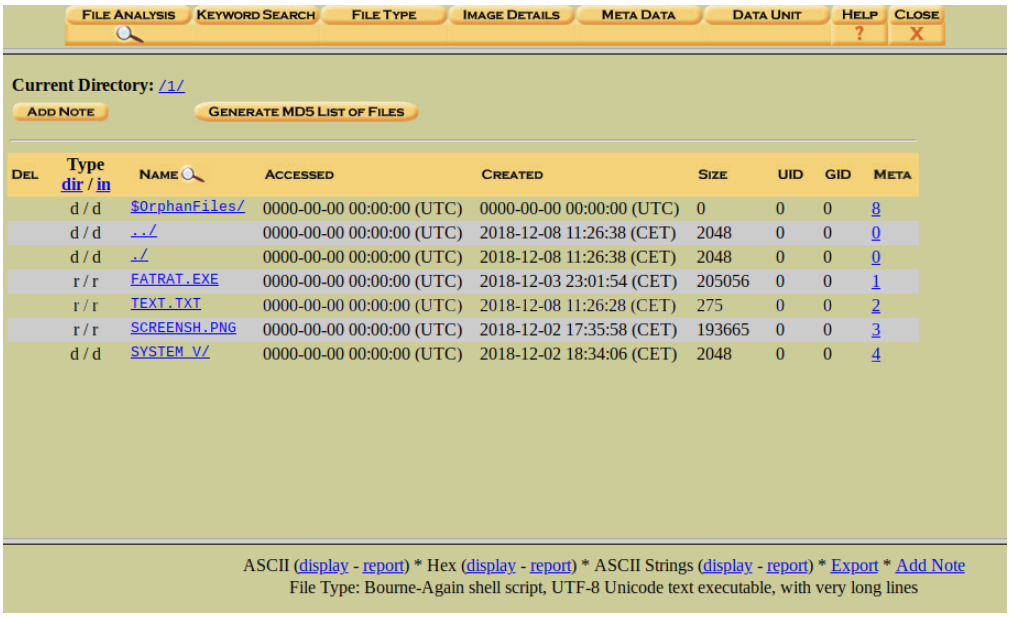
Obr. 6.3: Nastavení přidání obrazu disku.

- **File Type** – tento mód provede přezkoumání alokovaných a nealokovaných souborů a setřídí je do kategorii, to umožňuje najít. soubory podle typu nebo najít soubory skryté.
- **Image Details** – zobrazí informace o analyzovaném obrazu disku.
- **Meta Data** – zde si lze prohlédnout podrobnosti o libovolném záznamu adresáře v souborovém systému.
- **Data Unit** – zde si lze zobrazit obsah libovolného bloku souborového systému.

6.3.3 Nástroj ADB

Velmi rozšířený nástroj Android Debug Bridge, který umožňuje komunikovat s mobilním zařízením se systémem Android. Jedná se o program klient-server a samotný příkaz adb umožňuje provést různé akce jako instalace a ladění aplikací, poskytuje přístup k prostředí Unix a spouštění příkazů na samotném zařízení. Skládá se ze tří částí:

- **Klient**, který je spuštěn na vývojářském zařízení a kterým pomocí příkazu adb je možné odesílat příkazy na připojené zařízení.
- **Démon** běží jako proces na pozadí každého zařízení a vykonává přijaté příkazy na připojeném zařízení.



Obr. 6.4: Hlavní menu analýzy.

- **Server** je spuštěn na vývojářském zařízení a spravuje komunikaci mezi klientem a démonem.

Při spuštění jakéhokoliv příkazu adb, tedy při spuštění klienta, je zjištěno, zda je spuštěn také adb server. Pokud není, spustí se na pozadí a naváže se na lokální TCP port 5037 kde naslouchá příkazům odeslané z adb klientů. Server dále lokalizuje připojená zařízení skenováním portů 5555 až 5585. Tento rozsah je použitý prvními 16 emulátory kde adb server může najít démona běžícího na připojeném zařízení. Pokud je nalezne, nastaví připojení k portům emulátorů. Každý emulátor využívá dvou portů pro připojení zařízení. Jeden pro konzoly a druhý pro adb. Jakmile je připojení nastaveno lze pomocí jakéhokoliv klienta posílat příkazy na kterékoliv zařízení. Ukázka použití adb v kapitole 8. [18]

6.3.4 Nástroj Ghidra

Software, který byl vyvinut jako aplikační rámec reverzního inženýrství. Je vytvořen a spravován výzkumníky z NSA a obsahuje sadu špičkových nástrojů pro analýzu softwaru. Umožňuje uživatelům analyzovat zkompilovaný kód na platformách Windows, MacOS a Linux. Ghidra podporuje širokou paletu instrukčních sad a spustitelných formátů a poskytuje uživateli plnohodnotný a přehledný výstup analýz. Dovoluje také uživatelům vyvíjet své vlastní komponenty nebo skripty s podporou jazyků Java a Python. [19]

6.3.5 Nástroj SQLite Browser

Byl vyvinut Mauricio Piacentinim a původně pod názvem Arca Database Browser byl doprovodným nástrojem Arca Database Xtra. Jednalo se o komerční produkt, který vkládal SQLite databáze pro zpracování komprimovaných a binárních dat. Původní kód byl upraven, aby byl kompatibilní se standardními databázemi SQLite 2.x a přejmenován na SQLite Database Browser. V dnešní době se jedná o vysoce kvalitní, uživatelsky přívětiví, open source nástroj pro vytváření, navrhování a editaci databázových souborů kompatibilních s SQLite. Umožnuje vytvářet, vyhledávat a upravovat databáze. Proto je použit v kapitole 8 jako prohlížeč databázových souborů mobilního zařízení. [20].

6.3.6 Nástroj Radare

Radare je nástroj určený pro reverzní inženýrství, který umožňuje rozebrat (a sestavit) mnoho různých architektur. Dále umožňuje ladění s lokálním nativním a vzdáleným ladicím programem (gdb, rap, webui, r2pipe, winedbg, windbg). Poskytuje uživateli provádět forenzní analýzy na souborových systémech a vytvářet skripty v jazyce Python, Javascript, Go a další. Je k dispozici i v grafickém uživatelském rozhraní pomocí integrovaného webového serveru. Vizualizujte datové struktury několika typů souborů. [21].

7 Realizace analýzy podezřelého zařízení - USB flash disk

7.1 Vytvoření testovacího malwaru

Pro příklad analýzy, názornou ukázku a ověření funkčnosti programu Autopsy, bude vytvořen jednoduchý škodlivý soubor pomocí operačního systému Kali Linux. Tento soubor je dále umístěn na externím zařízení, které se podrobí analýze.

7.1.1 Nástroj The Fat Rat

Jde o jednoduchý nástroj pro vytvoření škodlivého souboru. Tento soubor může otevřít zadní vrátka do systému a obejít většinu antivirových programů. Do souboru jsou zkompilovány populární škodlivé kódy. Výsledný soubor je spustitelný například na platformě Windows, Android, Mac a jiné.
Více informací o nástroji The Fat Ran na adrese:
https://github.com/Screetsec/TheFatRat

Instalace a spuštění nástroje Fat Rat:
Do systému Kali se stáhne balíček potřebný pro instalaci nástroje. Následující příkazy balíček stáhnou a otevřou adresář s instalačním skriptem.
~#git clone https://github.com/Screetsec/TheFatRat.git

~#cd TheFatRat

Dále je nastaveno uživateli právo spuštění souboru a samotné spuštění instalace.

~#chmod +x setup.sh && ./setup.sh

Samotná instalace trvá cca 10 minut. Veškeré nástroje, které v systému Kali chybí a jsou potřebné pro fungování nástroje Fat Rat, jsou samy staženy a nainstalovány. Pro spuštění nástroje slouží příkaz:

~#fatrat

Tím se dostaneme do menu kde si lze zvolit typ vygenerovaného malwaru podle toho k čemu má být určen nebo jakou metodu pro útok použije.

7.2 Vytvoření obrazu nakaženého disku

V této kapitole bude vypsán jednoduchý postup pro vytvoření vstupního souboru do nástroje Autopsy. Pro vytvoření souboru obsahující digitální kopii dat disku, který bude analyzován, bude použit nástroj genisoimage.

Instalace balíčku genisoimage:

~#apt-get install genisoimage

Po připojení externího disku se v adresáři /media/<jmeno_uzivatele>/ vytvoří složka do adresáře připojeného disku. Tato složka nese název připojeného disku. Vstup do adresáře:

~#cd /media/<jmeno_uzivatele>

Ze složky připojeného disku bude vytvořen obraz pomoci genisoimage příkazu:

~#genisoimage -output <nazev_souboru.iso> <nazev_disku>

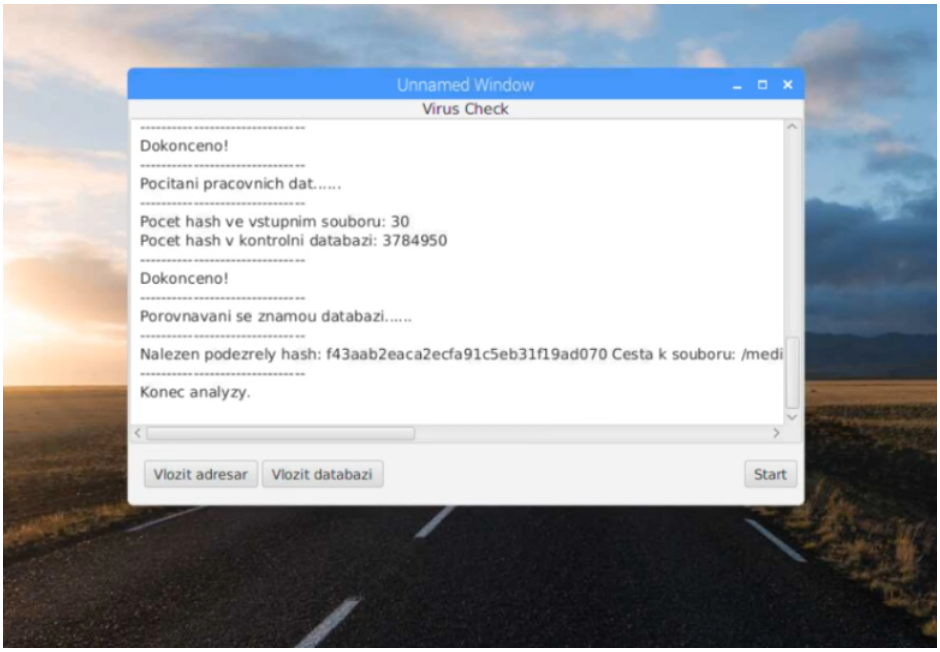
7.3 Výsledky analýzy

Ukázky z výsledných zpráv analýzy podezřelého zařízení ve výpisech A.1 a A.2. Ve výpise A.1 lze nalézt hlavičku, která informuje, že tento soubor je škodlivý. Obsahuje přímý název malwaru a k čemu slouží.. Ve výpise A.2 je výčet obsahu testového textového souboru obsaženého na podezřelém zařízení. Kontrolní součty veškerých souborů ve výpise A.3. Celkový výstup analýzy je obsažen na přiloženém CD společně s oficiální zprávou. V této zprávě jsou uvedeny informace o zkoumaném zařízení, o nalezeném obsahu, výsledky analýzy a doporučená opatření.

7.4 Hesh Check - vytvořený program pro analýzu externího disku

Za účelem usnadnění kontroly podezřelého zařízení je vytvořen program pro předběžnou kontrolu. Program a grafické rozhraní vyžadující vstup uživatele jsou napsány v programovacím jazyce JAVA a JAVA(fx) a pracují s databází známých hashů. Databáze hash souborů obsahuje celkem 3 784 950 položek, které jsou aktuální z

webového portálu <https://virusshare.com/hashes.4n6>. Tento portál poskytuje aktuální databázi za pomoci aplikace VirusTotal. Lze jej stáhnout a v případě aktualizace potřeba naformátovat jako kontrolní soubor pro program Hash Check. V této práci pod názvem HashData.txt umístěný na pracovní ploše Raspberry Pi. Zde se také nachází spustitelný program HashCheck.jar. Po spuštění je prvním krokem vybrání kořenového adresáře stisknutím tlačítka „Vlozit adresar“ tento adresář je vstupním adresářem USB flash disku a bude použit pro následnou kontrolu a tvorbu hash souborů. Tlačítkem „Vlozit databazi“ vybere uživatel databázi známých hash souborů. Po stisknutí tlačítka „Start“ jsou provedeny tři hlavní vnitřní operace. Iterace kořenového adresáře a vytvoření hash souborů vnořených a jejich zápis do souboru „HashLog.txt“. Dále je provedeno načtení dat z tohoto souboru a z databáze pro následné testování. Jednotlivé hash soubory jsou porovnány se známou databází a v případě shody je zobrazen výpis podezřelého hash souboru a cesta k tomuto souboru. Ukázka na obr. č. 7.1.



Obr. 7.1: Ukázka hash analýzy programu Hash Check.

Obsluha výše popsaná je také znázorněna ve videu analýzy USB flash disku umístěném na přiloženém CD. Společně s JAVA programem HashCheck.jar, kódem programu a vstupní hash databází. Viz. kapitola B.

8 Realizace analýzy podezřelého zařízení - Mobilní zařízení

V této kapitole jsou pro praktickou ukázku vytvořeny dva scénáře analýzy. V prvním případě se jedná o mobilní zařízení Xiaomi Mi2s s verzí Androidu 5.0.2. Na tomto zařízení se nacházejí běžná uživatelská data jako kontakty, SMS zprávy, historie volání, uložená hesla, galerie a nejpopulárnější aplikace jako Facebook nebo Instagram. Pro analýzu se tak zařízení co nejvíce přibližuje reálnému běžnému používání, které by mohla vlastnit například osoba podezřelá ze spáchání trestného činu. Pro druhý případ je stejné zařízení nakaženo škodlivou aplikací. Účelem této analýzy je nejen odhalit o jakou nákazu se jedná a jaké má následky, ale i zda je vytvořené prostředí schopné provádět požadované analýzy.

8.1 Analýza mobilního zařízení Xiaomi Mi2s

První analýza mobilního telefonu je zaměřena na uživatelsky běžně používané mobilní zařízení. Analýza má za účel získat důležitá data ze zařízení nalezeného například na místě činu, na kterém by se mohli vyskytovat informace důležité při vyšetřování. Analýza je provedena v rámci možností použitých open source variant. Pro tuto analýzu jsou na vývojářském zařízení použity programy ADB a SQLite Browser.

8.1.1 Příprava zařízení

V reálné situaci je důležité oddělit mobilní zařízení od okolní komunikace uzavřením do stíněného pouzdra tzv. Faradayovi klece. Pokud je mobilní zařízení chráněno například heslem je potřeba jej prolomit. To však není obsahem této práce.

Po získání přístupu je důležité co nejvíce zachovat původní obsah, pro další manipulaci je však nutné v mobilním zařízení získat administrátorský přístup tzv. root. Pro to je použita aplikace SuperSU. Dále je potřeba v mobilním zařízení otevřít možnosti pro vývojáře. To se provede v nastavení v záložce informace o telefonu. Pěti kliknutími na položku s verzí androidu, v případě Xiaomi na položku s verzí MIUI. V možnostech pro vývojáře povolit ladění přes USB. Po připojení mobilního telefonu do Raspberry Pi již lze provádět veškeré operace prostřednictvím ADB. Při této použité metodě je provedeno mnoho zásahů do zařízení, avšak vycházíme-li pouze z open source variant, nelze tuto analýzu srovnávat s analýzou při které jsou použity profesionální placené nástroje vyvíjené tak, aby do zařízení zasahovaly co nejméně.

8.1.2 Extrakce citlivých dat na logické úrovni

Samotnou analýzu lze jen stěží automatizovat. Existuje obrovské množství různých typů mobilního zařízení s různými verzemi Androidu. Nejúčinnějším způsobem je procházet adresářovou strukturu ručně. Některá data se však ukládají do podobných adresářů. V této analýze je zaměřena pozornost na získání dat pro prolomení hesla obrazovky, SMS zprávy, historii volání, seznam kontaktů, historii prohlížeče a galerii.

Po otevření příkazového řádku v prostředí Raspbian je třeba ověřit, zda je zařízení připojeno. To se provede příkazem:

~#adb devices

Výpis připojených zařízení (ID zařízení se může lišit):

List of devices attached
4df798d76f98cf6d unauthorized

Hodnota „unauthorized“ znamená, že zařízení není připojeno. V mobilním zařízení v záložce pro vývojáře je potřeba znovu ověření autorizace pro připojená zařízení. To se provede kliknutím na položku „Revoke USB debugging authorizations“ a povolením vyskakovacího okna s MAC adresou zařízení. Po znovu provedení příkazu je zařízení připojeno:

List of devices attached
4df798d76f98cf6d device

Následujícím příkazem se lze přepnout do příkazového řádku mobilního zařízení:

~#adb shell

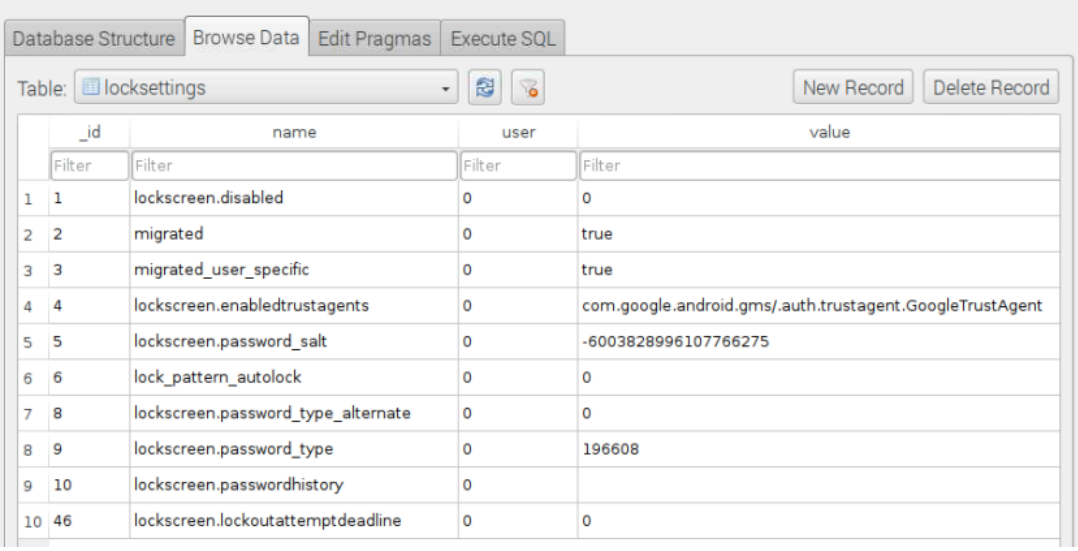
Dále je potřeba přepnout uživatele na root, který disponuje plným oprávněním při práci s androidem:

~#su root

Tím je zajištěn přístup k výpisům adresářů a možnosti stažení souborů ze zařízení. V tomto zařízení je pár důležitých adresářů, ve kterých se nacházejí námi požadovaná data. Z těchto adresářů jsou staženy ve většině případů databázové soubory. Stažení souboru ze zařízení lze provést následujícím příkazem:

~#adb pull /úplná cesta k souboru/

- **Heslo zamykací obrazovky** - pro prolomení hlavního hesla jsou potřeba dva soubory. Prvním je /data/system/password.key. Druhý soubor obsahuje tzv. sůl, která zajišťuje vyšší bezpečnost hesla a která je uložena v databázi lockscreen.password_salt. Databáze je uložena v adresáři /data/data/-com.android.providers.settings/databases v souboru settings.db. Tzv. sůl na obr. č. 8.1. Jakmile máme k dispozici tyto dva souboru, lze hrubou silou získat

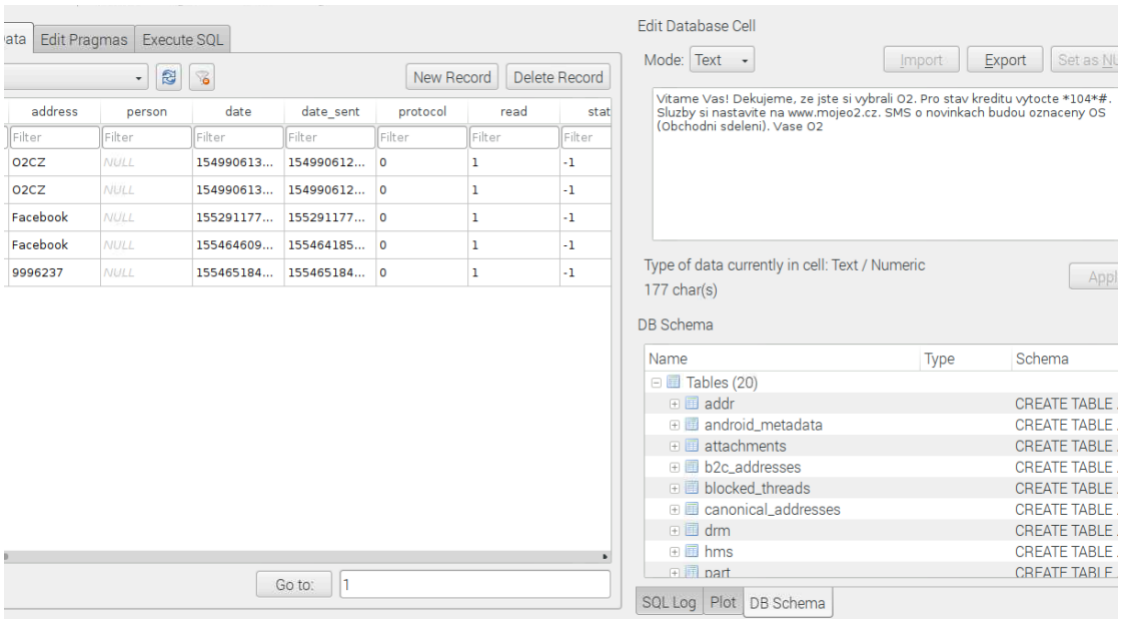


	_id	name	user	value
	Filter	Filter	Filter	Filter
1	1	lockscreen.disabled	0	0
2	2	migrated	0	true
3	3	migrated_user_specific	0	true
4	4	lockscreen.enabledtrustagents	0	com.google.android.gms/.auth.trustagent.GoogleTrustAgent
5	5	lockscreen.password_salt	0	-6003828996107766275
6	6	lock_pattern_autolock	0	0
7	8	lockscreen.password_type_alternate	0	0
8	9	lockscreen.password_type	0	196608
9	10	lockscreen.passwordhistory	0	
10	46	lockscreen.lockoutattemptdeadline	0	0

Obr. 8.1: Soubor settings.db obsahují sůl hesla.

požadované heslo.

- **SMS zprávy** - databázový soubor s uloženými SMS zprávami se nachází v adresáři /data/data/com.android.providers.telephony/databases a jedná se o soubor mmssms.db. Opět za pomoci SQLite Browser programu lze SMS zprávy zobrazit. Zprávy na obr. č. 8.2.
- **Historie volání** - databázový soubor s uloženou historií volání se nachází v adresáři /data/data/android.providers.contacts/databases v souboru contacts2.db. V programu SQLite Browser jsou data uložena v tabulce calls. Viz obr. č. 8.3.
- **Seznam kontaktů** - databázový soubor s uloženými kontakty se nachází ve stejném souboru v adresáři /data/data/android.providers.contacts/databases v souboru contacts2.db. Data jsou uloženy v tabulce data. Viz obr. č. 8.4.



Obr. 8.2: Soubor mmssms.db obsahující sms a mms zprávy.

- **Historie prohlížeče** - databázový soubor s uloženou historií prohlížeče se nachází v adresáři /data/data/com.android.browser/databases v souboru browser2.db. Viz obr. č. 8.5.
- **Galerie** Galerii lze najít v běžné adresářové struktuře. V tomto případě /sdcard/DCIM/Camera/.

Výstupem analýzy je získání osobních až citlivých dat. Zároveň ukázka extrakce dat na logické úrovni, při které je provedeno mnoho zásahů do integrity dat. Pomocí této analýzy je však jednoduché procházet adresářovou strukturu mobilního zařízení a extrahovat potřebné soubory či databáze.

8.2 Analýza škodlivé aplikace na mobilním zařízení

Druhý scénář analýzy je zaměřen na odhalení aplikace, která obsahuje škodlivý kód. Pro tuto ukázku je vybrán malware Android Rootkit, který je implementován na mobilním zařízení Xiaomi Mi2S. Pro analýzu jsou použity nástroje ADB, Radare, Java Decompiler a Classy Shark.

8.2.1 Raspberry Pi jako přístupový bod

Při nakažení mobilního zařízení škodlivým kódem, lze očekávat komunikaci mezi mobilním zařízením a C&C serverem. Ten má za úkol přijímat a odesílat požadavky

Database Structure | Browse Data | Edit Pragmas | Execute SQL

Table: calls

New Record | Delete Record

	_id	number	presentation	date	duration	type	new	name	numbe
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	2	750 632 498	1	155574867...	-1	10	1	NULL	0
2	3	764 318 643	1	155574870...	-1	10	1	NULL	0
3	4	543 494 674	1	155574872...	-1	10	1	NULL	0

Obr. 8.3: Soubor contacts2.db obsahující historii volání.

na mobilní zařízení skrze aplikaci, která škodlivý kód obsahuje. Aby bylo možné tuto komunikaci odposlouchávat, je třeba překonfigurovat Raspberry Pi jako přístupový bod Wi-Fi sítě. Napadené mobilní zařízení se připojí k tomuto bodu a tím je zajištěno, že veškerá komunikace bude procházet skrze Raspberry Pi kde ji lze zachytávat programem Wireshark. Dále bude následovat postup konfigurace.

Prvním krokem je vždy ověřit, zda je operační systém Raspbian aktuální. Aktualizace se provádí příkazy:

```
~#sudo apt-get update
~#sudo apt-get upgrade
```

Po aktualizaci je dobré provést restart systému příkazem sudo reboot.

Druhým krokem je nainstalovat chybějící programy, které budou pro další konfiguraci potřeba.

```
~#sudo apt-get install hostapd
~#sudo apt-get install dnsmasq
~#sudo apt-get install iptables-persistent
```

Database Structure | Browse Data | Edit Pragmas | Execute SQL

Table: data

New Record | Delete Record

	_super_primar	data_version	data1	data2	data3	data4	data5	data6
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	0	1	1	NULL	NULL	NULL	NULL	NULL
2	0	1	750 632 498	2	NULL	750632498	NULL	NULL
3	0	1	Pavel Novak	Pavel	Novak	NULL	NULL	NULL
4	0	1	NULL	NULL	NULL	NULL	NULL	NULL
5	0	1	1	NULL	NULL	NULL	NULL	NULL
6	0	1	764 318 643	2	NULL	764318643	NULL	NULL
7	0	1	Petra Novak...	Petra	Novakova	NULL	NULL	NULL
8	0	1	NULL	NULL	NULL	NULL	NULL	NULL
9	0	1	1	NULL	NULL	NULL	NULL	NULL
10	0	1	543 494 674	2	NULL	543494674	NULL	NULL
11	0	1	Tomas Tichy	Tomas	Tichy	NULL	NULL	NULL
12	0	1	NULL	NULL	NULL	NULL	NULL	NULL
13	0	0	NULL	NULL	NULL	NULL	NULL	NULL
14	0	0		NULL	NULL	NULL	NULL	NULL
15	0	0	NULL	NULL	NULL	NULL	NULL	NULL

Obr. 8.4: Soubor contacts2.db obsahující historii volání.

Balíček hostapd dovoluje vytvořit přístupový bod na Raspberry Pi, dnsmasq umožnuje jednoduchou konfiguraci DHCP a DNS serveru a iptables-persistent zachovává pravidla uložená v iptables i po restartu systému.

Před konfigurací je dobré tyto programy vypnout příkazy:

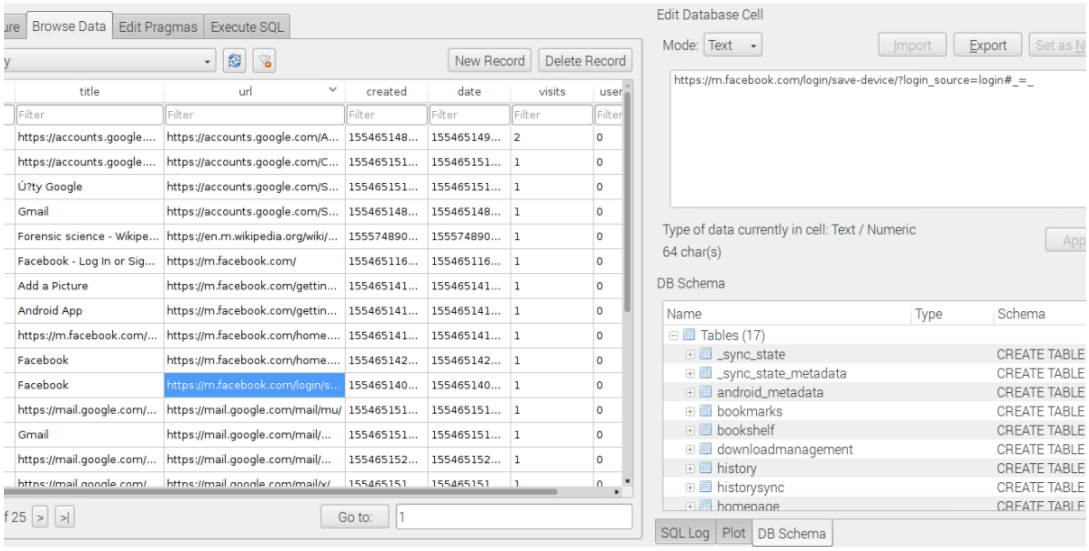
~#sudo systemctl stop hostapd
~#sudo systemctl stop dnsmasq

Třetím krokem je konfigurace rozhraní wlan0 se statickou IP adresou. Pro tento účel lze použít standardní adresu 192.168.xxx.xxx. V konfiguračním souboru dhcpcd lze rozhraní wlan0 přiřadit IP adresu, v tomto případě je použita adresa 192.168.0.10. Příkazem:

~#sudo nano /etc/dhcpcd.conf

Přejdeme do konfiguračního souboru a za poslední řádek přidáme:

interface wlan0



Obr. 8.5: Soubor contacts2.db obsahující historii volání.

static ip_address=192.168.0.10/24
denyinterfaces eth0
denyinterfaces wlan0

Poslední dva řádky umožní vytvoření mostu mezi rozhraními eth0 a wlan0. Souboru uložíme a uzavřeme.

Čtvrtým krokem je nastavení DHCP serveru. Pro to použijeme nainstalovaný dnsmasq, avšak původní konfigurace obsahuje spoustu nepotřebných informací, a proto je jednodušší tento soubor uložit pod jiným názvem a vytvořit nový prázdny. To lze provést příkazy:

~#sudo mv /etc/dnsmasq.conf /etc/dnsmasq.conf.orig
~#sudo nano /etc/dnsmasq.conf

Do nového konfiguračního souboru je třeba vložit dva řádky:

interface=wlan0
dhcp-range=192.168.0.11,192.168.0.30,255.255.255.0,24h

Ty říkají, že pro rozhraní wlan0 bude pro distribuci IP adres použit rozsah 192.168.0.11 až 192.168.0.30.

V pátém kroku upravíme konfigurační soubor hostapd:
~#sudo nano /etc/hostapd/hostapd.conf

A vložíme do něj informace potřebné pro vytvoření přístupového bodu:
interface=wlan0
bridge=br0
hw_mode=g
channel=7
wmm_enabled=0
macaddr_acl=0
auth_algs=1
ignore_broadcast_ssid=0
wpa=2
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
rsn_pairwise=CCMP
ssid=NETWORK
wpa_passphrase=PASSWORD

Na místo „NETWORK“, třeba vložit nový název přístupového bodu a místo „PASSWORD“ heslo. Dále je potřeba říci systému kde tento konfigurační soubor nalezne. Otevřeme si proto soubor hostapd:

~#sudo nano /etc/default/hostapd

V tomto souboru třeba najít řádek obsahující #DAEMON_CONF=””, smazat # a mezi uvozovky vložit cestu ke konfiguračnímu souboru. Řádek by měl vypadat takto:

DAEMON_CONF="/etc/hostapd/hostapd.conf"

Myšlenka je taková, že když se mobilní zařízení připojí k Raspberry Pi, komunikace bude směrována z rozhraní wlan0 přes ethernetový kabel do sítě. K tomu je zapotřebí změnit konfigurační soubor sysctl:

~#sudo nano /etc/sysctl.conf

V tomto souboru najít řádek obsahující:

#net.ipv4.ip_forward=1

A opět smazat #.

Předposlední krok je přidat pravidlo do iptables obsahující maškarádu pro odchozí komunikaci na rozhraní eth0:

~#sudo iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE

Uložit pravidlo do souboru rules.v4, který bude použit při startu systému:

~#sudo sh -c "iptables-save > /etc/iptables/rules.v4"

A načíst pravidlo do systému, tím si ověříme, že je napsáno správně:

~#sudo iptables-restore < /etc/iptables.ipv4.nat

Nyní Raspberry Pi funguje jako přístupový bod a mobilní zařízení se k němu mohou připojit. Nemají však ještě přístup do internetu, k tomu je zapotřebí vytvořit most, který propustí veškerou komunikaci mezi rozhraními wlan0 a eth0. Pro vytvoření mostu je potřeba nainstalovat balíček:

~#sudo apt-get install bridge-utils

Vytvořit most:

~#sudo brctl addbr br0

Propojit rozhraní eth0 s br0:

~#sudo brctl addif br0 eth0

Upravit soubor rozhraní:

~#sudo nano /etc/network/interfaces

A na konec souboru přidat řádky:

auto br0
iface br0 inet manual
bridge_ports eth0 wlan0

Nyní je vše připraveno a zbývá provést restart systému pomocí sudo reboot. Po

zapnutí systému je mobilní zařízení schopné přihlásit se k přístupovému bodu a má přístup do internetu.

8.2.2 Analýza podezřelé aplikace

Cílem této kapitoly je prokázat, že navržené zařízení je schopné provést analýzu srovnatelnou s použitím profesionálních nástrojů. Pro tento účel je vybrána škodlivá aplikace, kterou je nakaženo mobilní zařízení. Tato aplikace je dále analyzována za pomoci open source nástrojů, aby byl odhalen její škodlivý obsah a techniky které používá.

Škodlivá aplikace, kterou je mobilní zařízení nakaženo se nazývá Android Rootkit. Jak z názvu vyplívá, jedná se o malware, jehož cílem je získat úplný, tedy root, přístup do mobilního zařízení. Aplikace vystupuje a maskuje se za aplikaci File Helper, která uživateli umožňuje procházení souborové systému. Aplikace používá pokročilé techniky proti zpětnému zkonstruování a po získání oprávnění uživatele root začne zobrazovat nežádoucí reklamu, porno, instaluje další aplikace nebo vytváří odkazy aplikací na domovské obrazovce.

Jako první krok, je pokusit se odchytit komunikaci mezi škodlivou aplikací a C&C serverem. Z toho důvodu bylo nutné překonfigurovat Raspberry Pi jako přístupový bod. Postup popsán v kapitole 8.2.1. K odchycení komunikace použijeme nástroj Wireshark, kde je možné oddělit běžnou komunikaci mobilního zařízení od podezřelé za pomoci filtrů. Po tomto oddělení bylo zjištěno, že škodlivá aplikace komunikuje se třemi IP adresami.

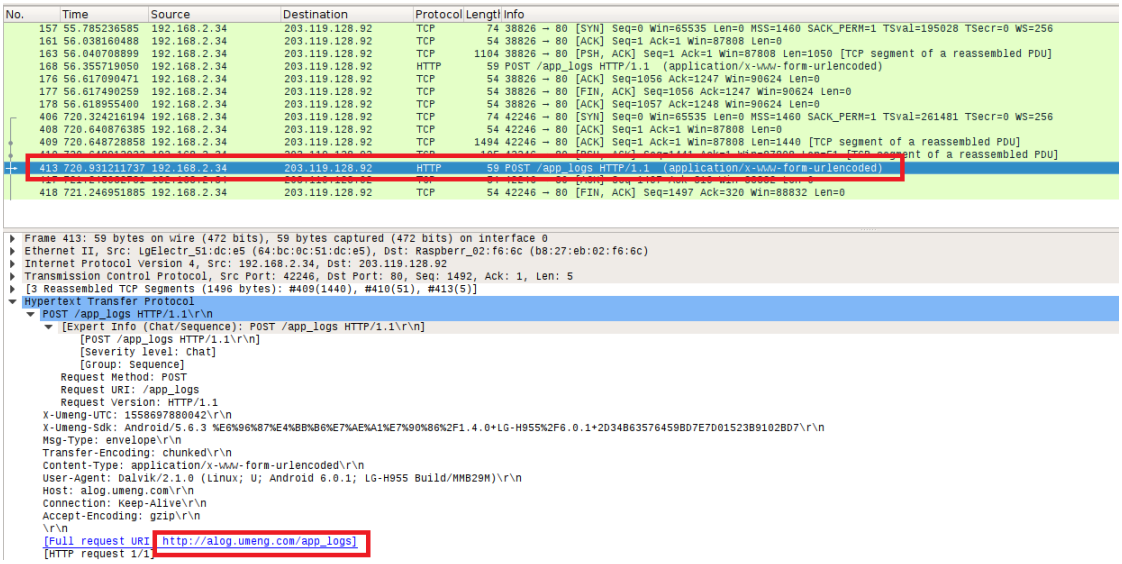
- 123.56.206.59
- 203.119.128.92
- 50.3.254.101

Na obrázku č. 8.6 a 8.7 je znázorněna komunikace pomocí protokolu HTTP, mezi škodlivou aplikací a dvěma podezřelými servery.

- http://alog.umeng.com/app_logs
- http://gt.rogsob.com/stmp/ad.png

Z analýzy lze usoudit, na IP adrese 50.3.254.101, server http://gt.rogsob.com/ již není dostupný a proto aplikace pravděpodobně neobdrží potřebné instrukce pro své správné fungování.

Po připojení nakaženého mobilního zařízení lze použít postup popsaný v předchozí kapitole pro získání souborů užitečných pro analýzu. Aplikace je uložena v paměti mobilního zařízení ve formátu APK. Při instalaci je aplikace nejprve zkompilována a poté zabalena do souboru APK společně s dalšími užitečnými soubory, certifikáty nebo souborem manifest. Po instalaci je soubor APK uložen v souborovém systému, pro systémové aplikace nejčastěji /sdcard/system/app a pro uživatel-



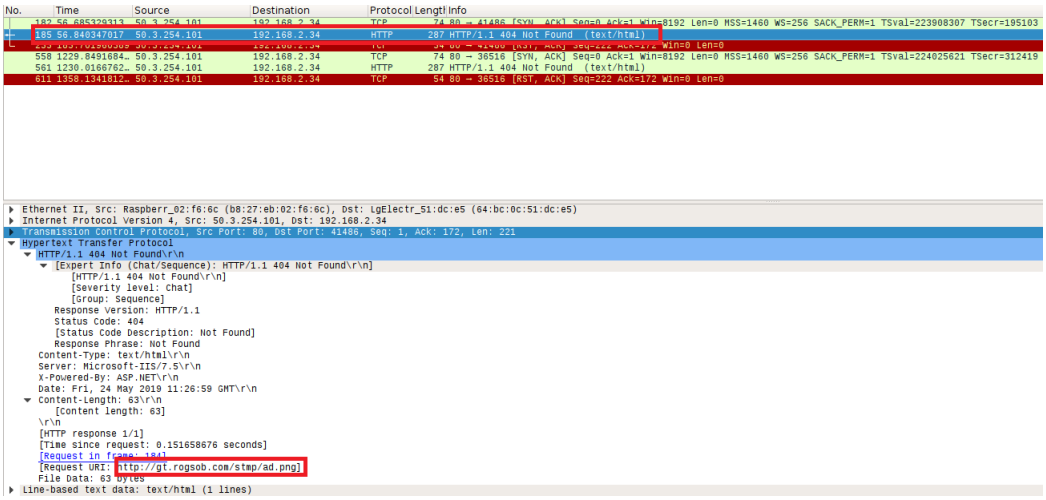
Obr. 8.6: Síťová komunikace škodlivé aplikace.

ské /sdcard/data/app. V těchto adresářích jsou uloženy adresáře s názvy nainstalovaných aplikací. V našem případě se jedná o složku com.android.filehelper, který stáhneme do testovacího zařízení přkazem:

~#adb pull <úplná cesta k adresáři>

V této složce je vše potřebné pro analýzu aplikace. Je zde uložena spustitelná část aplikace v souboru classes.dex. Jedná se o archív, který obsahuje veškeré zkompilované třídy prezentované ve formě bajtů. Dalším důležitým souborem je AndroidManifest.xml, který obsahuje informace o oprávněních požadovaných aplikací během instalace. Analýza tohoto souboru je základ pro zjištění škodlivých funkcí aplikace. Začneme tedy tímto souborem. I když je soubor ve formátu XML, nelze jej otevřít v předinstalovaných aplikacích. Aby bylo možné získat čitelnou podobu toho souboru, je potřeba jej otevřít v nástroji Classy Shark. Jde o samostatný binární nástroj pro vývojáře, který umožňuje procházek spustitelný soubor Android. Dále podporuje formáty a knihovny (.dex, .aar, .so) a spustitelné soubory (.apk, .jar, .class) včetně binárních souborů XML.

Na obrázku 8.8 nelze najít hlavní aktivitu a logické funkce com.sd.clip.activity.FileManagerActivity. Hlavní logika aplikace není v souboru classes.dex. Aplikace načítá druhý DEX soubor, ve kterém je hlavní logika obsažena. Avšak soubor napovídá jít hlouběji do třídy SecAppWrapper, který je obsažen v souboru classes.dex. Dalším krokem je tedy extrakce tříd z toho souboru. Aby bylo možné soubor classes.dex dekompilovat je třeba jej přeformátovat z formátu DEX



Obr. 8.7: Síťová komunikace škodlivé aplikace.

na JAR. K tomu slouží nástroj dex2jar. Jedná se o skriptovací nástroj, kterému předáme cestu k souboru a převedeme do požadovaného souboru příkazem:

~#sudo sh cesta k nástroji/d2j-dex2jar.sh cesta k souboru/classes.dex

Vytvořený JAR soubor lze zkompilovat a získat tak čitelnou podobu tříd. K tomu slouží další nástroj Java Decompiler. Ve kterém si otevřeme vytvořený soubor classes-dex2jar.jar. Tím získáme přístup do struktury tříd. Nás bude zajímat již zmíněná třída SecAppWrapper.

V části kódu třídy SecAppWrapper na obr. č. 8.10 jsou zobrazeny tři kroky použitých technik. Prvním krokem je provedení statického blokového kódu, který má za úkol načíst knihovnu libSecShell.so, která obsahuje dynamické odkazy. Aplikace dále vstoupí do nativní vrstvy a provede několik operací proti debuggování, dešifruje a načte druhý soubor DEX, který obsahuje hlavní logiku aplikace. Ve třídě DexInstall je provedena metoda pro instalaci druhého souboru DEX. Viz obr. č. 8.11. Tato metoda je provedena v nativní vrstvě.

Dále na obrázku č. 8.8 funkcí attachBaseContext aplikace načte třídu com.sd.clip.base.MyApplication, což je prováděcí položka sekundárního DEX.

Ve funkci onCreate aplikace provede metodu onCreate třídy com.sd.clip.base.MyApplication.

První DEX soubor je poměrně jednoduchý. Z výsledků analýzy lze získat předběžný přehled o provádění funkcí podezřelé aplikace.

Dále je třeba provést hloubkovou analýzu nativního kódu vrstvy, což je velmi složité a nad rámec znalostí potřebné pro bakalářskou práci. Proto je dále popsáno,

```
</uses-permission>
<application
  theme='@res/0x0103000D'
  label='@res/0x7F070000'
  icon='@res/0x7F020003'
  name='com.secshell.shellwrapper.SecAppWrapper'
  debuggable='true'
  allowBackup='true'>
 <activity
   label='@res/0x7F070000'
   name='com.sd.clip.activity.SDManagerActivity'
   exported='true'
   launchMode='2'
   screenOrientation='1'>
 </activity>
 <activity
   theme='@res/0x0103000D'
   name='com.sd.clip.activity.FileManagerActivity'
   screenOrientation='1'>
  <intent-filter>
   <action
     name='android.intent.action.MAIN'>
   </action>
   <category
     name='android.intent.category.LAUNCHER'>
   </category>
  </intent-filter>
 </activity>
 <activity
   theme='@res/0x0103000D'
   name='com.sd.clip.activity.FileDeleteActivity'
```

Obr. 8.8: Soubor AndroidManifest.xml.

jak by vypadal další postup analýzy.

Kód nativní vrstvy používá některé pokročilé anti-debug a anti-hook techniky, a také používá několik dešifrovacích algoritmů pro dešifrování některých bajtových polí pro získání prostého textového řetězce. Následující část obsahuje analýzu funkcí knihovny liSecShell.so. Viz obr. č. 8.12. Která obsahuje dynamické odkazy. Nás zajímá funkce JNI_Onload ve kterém je veškerý anti-debug kód.

Následuje úryvek kódu funkce JNI_Onload, který registruje nativní metodu v nativní vrstvě. Obr. č. 8.13. K analýze souboru libSecShell.so je použit nástroj Radare.

Dalším postupem analýzy by bylo najít umístění anti-debug kódu a obejít jej, analyzovat a zjistit klíč potřebný pro dešifrování souboru secData0.jar a druhého DEX souboru classes.dex. Soubor secData0.jar lze najít v adresáři intalované aplikace. Bohužel jej nelze díky šifrování analyzovat.

Anti-debug a anti-hook metody vytvářejí velkou překážku pro reverzní inženýrství. Takže obcházení těchto anti-metod je nejtěžším úkolem.

Po získání druhého souboru DEX, který obsahuje hlavní logiku aplikace by byl proveden stejný postup analýzy jako u prvního DEX souboru popsaný výše. Teprve tato logika provádí škodlivou činnost. Získali bychom tak informace o jednotlivých metodách a postupech škodlivé činnosti aplikace.

Tato analýza dále odhalila, že i přesto, že aplikace komunikuje s C&C serverem

```
    theme='@res/0x0103000D'
    name='com.sd.clip.activity.FileDeleteActivity'
    screenOrientation='1'>
  </activity>
  <service
    name='com.sd.clip.urr.UploadErrorInfoService'
    exported='false'>
  </service>
  <service
    name='com.hg.mer.PG'
    process=':dys'>
  </service>
  <receiver
    name='com.hg.mer.Nws'>
    <intent-filter>
      <action
        name='android.intent.action.BOOT_COMPLETED'>
      </action>
    </intent-filter>
    <intent-filter>
      <action
        name='android.intent.action.USER_PRESENT'>
      </action>
    </intent-filter>
    <intent-filter>
      <action
        name='android.net.conn.CONNECTIVITY_CHANGE'>
      </action>
    </intent-filter>
```

Obr. 8.9: Soubor AndroidManifest.xml.

a obsahuje veškeré škodlivým kódem nakažené soubory. Tak není schopna získat dodatečné soubory z C&C serverů. Viz obr. č. 8.7. Lze předpokládat, že podpora škodlivé aplikace již není dostupná.

8.2.3 Popis funkce podezřelé aplikace

Po stažení, instalaci a spuštění škodlivé aplikace je načtena knihovna secShell, viz obr. č. 8.10. Tato knihovna je obsažena již v instalačním balíčku APK a poskytuje přístup k anti-hook metodám a dešifrování souboru secData0.jar. Tento soubor je také součástí instalačního balíčku, nelze však provést jeho analýzu dokud je zašifrován. Po dešifrování souboru secData0.jar je provedeno rozbalení tohoto souboru. Nachází se v něm již zmíněný druhy DEX soubor, který je opět zašifrovaný. Po dešifrování je tento soubor mobilním zařízením spuštěn díky metodě multidex a jsou zahájeny již škodlivé logické funkce. Aplikace skrze mobilní zařízení naváže komunikaci s C&C serverem a zahájí se stahování škodlivých dat. Tyto data jsou dále dešifrovány a za jejich pomoci je vytvořen soubor psneure.js. Tento soubor má za úkol pomocí příkazových úkolů, které přinutí mobilní zařízení nainstalovat dvě škodlivé aplikace BSetting a Riuy. Tyto aplikace spustí své škodlivé funkce. Zobrazování nežádoucí reklamy, odinstalace aplikací, vytváření ikon aplikací na obrazovce mobilního telefonu. Viz blokové schéma 8.14.

Výstupem analýzy je ověření, že zařízení je schopné provádět hloubkovou analýzu a disponuje nástroji potřebnými pro tuto analýzu. Hloubková analýza však vyžaduje

```
SecAppWrapper.class
public class SecAppWrapper extends Application {
  public static Application realApplication = null;

  static  {
    System.loadLibrary("SecShell");
    if (Helper.PPATH != null)
      System.load(Helper.PPATH);
  }

  protected void attachBaseContext(Context paramContext) { super.attachBaseContext(paramContext);
    try {
      realApplication = (Application)getClassLoader().loadClass(Helper.APPNAME).newInstance();
      Helper.attach(realApplication, paramContext);
      return;
    } catch (Exception paramException) {
      realApplication = null;
      return;
    }  }

  public void onConfigurationChanged(Configuration paramConfiguration) { super.onConfigurationChanged(paramConfiguration);
    if (realApplication != null)
      realApplication.onConfigurationChanged(paramConfiguration);  }

  public void onCreate() { super.onCreate();
    try {
      huawei_share();
    } catch (Exception exception) {}
    if (realApplication != null) {
      Helper.attach(realApplication, null);
      realApplication.onCreate();
    }  }
```

Obr. 8.10: Třída SecAppWrapper.

již vyšší znalosti a dovednosti v této problematice.

Zařízení se vyrovná profesionálním produktům při dodržení nízké ceny a efektivnosti. Zároveň je zařízení použitelné pro další vývoj a je možné jej dále rozvíjet o nové nástroje ať už hardwarové či softwarové.

Informace v této kapitole byly čerpány z [22].

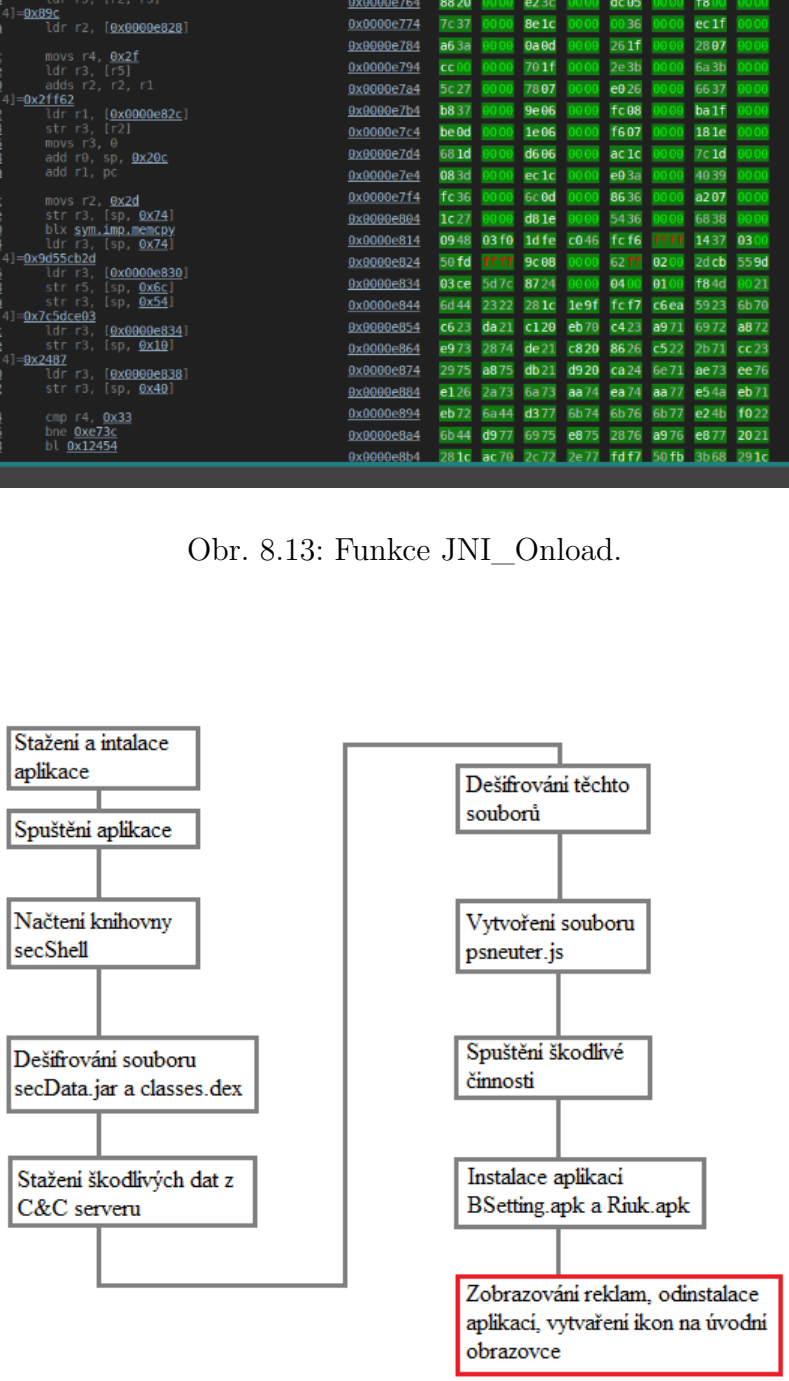
```
SecAppWrapper.class   DexInstall.class

  private static Method findMethod(Object paramObject, String paramString, Class<?>... paramVarArgs) throws NoSuchMethodException {
    Class clazz = paramObject.getClass();
    while (clazz != null) {
      try {
        Method method = clazz.getDeclaredMethod(paramString, paramVarArgs);
        if (!method.isAccessible())
          method.setAccessible(true);
        return method;
      } catch (NoSuchMethodException noSuchMethodException) {
        clazz = clazz.getSuperclass();
      }
    }
    throw new NoSuchMethodException("Method " + paramString + " with parameters " + Arrays.asList(paramVarArgs) + " not found in " + paramObje
  }

  public static void install(ClassLoader paramClassLoader, String paramString) {
    try {
      File file = new File(paramString);
      ArrayList arrayList = new ArrayList();
      arrayList.add(file);
      installSecondaryDexes(paramClassLoader, file.getParentFile(), arrayList);
      return;
    } catch (Exception paramClassLoader) {
      paramClassLoader.printStackTrace(System.out);
      return;
    }
  }

  private static void installSecondaryDexes(ClassLoader paramClassLoader, File paramFile, List<File> paramList) throws IllegalArgumentExceptic
      if (Build.VERSION.SDK_INT >= 19) {
        V19.access$000(paramClassLoader, paramList, paramFile);
        return;
      }
    } else {
      return;
    }
    if (Build.VERSION.SDK_INT >= 14) {
      V14.access$100(paramClassLoader, paramList, paramFile);
      return;
    }
    V4.access$200(paramClassLoader, paramList); }
```

Obr. 8.11: Třída DexInstall.

```
Classes   Methods count
E5E22B357893BC15A50DC35E
  AndroidManifest.xml
  classes
    classes.dex
      com.secshell.shellwrapp
        BuildConfig
        DexInstall
        DexInstall$V14
        DexInstall$V19
        DexInstall$V4
        Helper
        R
        R$attr
        R$dimen
        R$drawable
        R$id
        R$layout
        R$menu
        R$string
        R$style
        SecAppWrapper
  res
  libs
    lib/armeabi/libSecShell-x8
    lib/armeabi/libSecShell.so

File size - 279 KB

Native Dependencies

  liblog.so
  libz.so
  libstdc++.so
  libm.so
  libc.so
  libdl.so

Dynamic Symbols

  -- JNI_OnLoad
  -- RAW_DEX
  -- _Unwind_Backtrace
  -- _Unwind_Complete
  -- _Unwind_DeleteException
  -- _Unwind_ForcedUnwind
  -- _Unwind_GetCFA
  -- _Unwind_GetDataRelBase
  -- _Unwind_GetLanguageSpecificData
  -- _Unwind_GetRegionStart
  -- _Unwind_GetTextRelBase
  -- _Unwind_RaiseException
  -- _Unwind_Resume
  -- _Unwind_Resume_or_Rethrow
  -- _Unwind_VRS_Get
```

Obr. 8.12: Knihovna libSecShell.so.

Obr. 8.13: Funkce JNI_Onload.



Obr. 8.14: Blohové schéma funkce aplikace File Helper.

9 Závěr

Tato bakalářská práce je zaměřena na vytvoření prostředí pro analýzu podezřelého zařízení. V teoretické části poskytuje základní, ale také hlubší informace s tímto tématem spojené. Důležité a základní znalosti pro pochopení analýzy podezřelého zařízení. V první části teorie je zahrnut stručný popis definice sandbox a jakých technik je využíváno pro detekci malwaru. Ve druhé části jsou popsány typy malwaru a jakým způsobem získávat informace jak o již známých, tak i nových typech malwaru. Dále následuje popis technik, jakých malware využívá, aby byl schopný předejít odhalení, bránit se odhalení nebo jej obejít. Třetí část teorie obsahuje informace o operačním systému Android a typech analýz mobilního zařízení.

Praktická část již popisuje použité zařízení pro tvorbu prostředí. Použité hardwarové a softwarové prvky, jejich tvorbu a implementaci na základní desku Raspberry Pi. Možné konfigurace operačního systému Rasbian pro rozšíření možností prováděných analýz. Dále je praktická část rozdělena na dvě ukázky analýz podezřelého zařízení. V první ukázce, dokázalo analyzovat USB flash disk, na kterém se nacházel škodlivý soubor. V této části je popsáno vytvoření tohoto škodlivého souboru, postup a výsledek analýzy. Dále je zde popsán program, který poskytuje jednoduchou kontrolu připojeného USB flash disku a který byl vytvořen a naprogramován speciálně pro vytvořené prostředí. Druhá ukázka obsahuje dvě analýzy mobilního zařízení. V první je provedena analýza mobilního zařízení, na kterém se nachází běžná uživatelská data a postup získání těchto dat ze zařízení. Ve druhé je provedena analýza mobilního zařízení nakaženého škodlivou aplikací. Je provedena hloubková analýza této aplikace a ověření funkcionality a schopností vytvořeného zařízení pro analýzu podezřelého zařízení. Výstupem práce je prostředí, které je schopné jak základní, tak hloubkové analýzy podezřelého zařízení. Při dodržení efektivního a levného řešení s použitím open source nástrojů, se takřka vyrovná profesionálním produktům.

Všechny cíle, kterých mělo být dosaženo jsou splněny. Poslední provedená analýza vyžaduje vyšší znalosti a dovednosti v dané problematice. S ohledem na to je proveden alespoň nástin této analýzy a ověření schopností zařízení. To je použitelné i pro další vývoj a je možné jej dále rozvíjet o nové nástroje ať už hardwarové či softwarové.

Literatura

[1] *Sandbox software development. Techopedia* [online]. 2018, [cit. 5. 11. 2018]. Dostupné z URL: <https://www.techopedia.com/definition/27681/sandbox-software-development>

[2] ROUSE, Margaret.: *Sandbox. Techtarget.com.* [online]. 2005, [cit. 8. 11. 2018]. Dostupné z URL: <https://searchsecurity.techtarget.com/definition/sandbox>

[3] *Sandbox. Stackoverflow.com.* [online]. 2018, [cit. 8. 11. 2018]. Dostupné z URL: <https://stackoverflow.com/questions/2126174/what-is-sandboxing>

[4] IDIKA, N., Mathur, P., A.: *A Survey of Malware Detection Techniques* [online]. West Lafayette: Department of Computer Science. 2007, [cit. 9. 11. 2018]. Dostupné z URL: <https://www.researchgate.net/publication/229008321_A_survey_of_malware_detection_techniques>

[5] *Základní definice vztahující se k tématu kybernetické bezpečnosti* [online]. Praha: Ministerstvo vnitra české republiky. 2009, [cit. 15. 11. 2018]. Dostupné z URL: <www.mvcr.cz/soubor/cyber-vyzkum-studie-pojmy-pdf.aspx>

[6] Neil DuPaul: *Common Malware Types: Cybersecurity 101* [online]. 2009, [cit. 15. 11. 2018]. Dostupné z URL: <https://www.veracode.com/blog/2012/10/common-malware-types-cybersecurity-101>

[7] *Features of MISP, the open source threat sharing platform.* [online]. 2018, [cit. 16. 11. 2018]. Dostupné z URL: <https://www.misp-project.org/features.html>

[8] *NjRAT. In: MISP Galaxy Clusters.* [online]. s. 1758 2018, [cit. 16. 11. 2018]. Dostupné z URL: <https://www.misp.software/galaxy.pdf>

[9] *Funkce MISP Galaxy Cluster.* [online]. 2018, [cit. 16. 11. 2018]. Dostupné z URL: <https://github.com/MISP/misp-galaxy>

[10] *Funkce MISP Taxonomie.* [online]. 2018, [cit. 16. 11. 2018]. Dostupné z URL: <https://github.com/MISP/misp-taxonomies>

[11] Dejan Lukan: *Pafish (Paranoid Fish).* [online]. 2014, [cit. 20. 11. 2018]. Dostupné z URL: <https://resources.infosecinstitute.com/pafish-paranoid-fish>

[12] Dr. Carsten Willems: *Sandbox Evasion Techniques – Part 1-3.* [online]. X-RAY VISION FOR MALWARE. 2016, [cit. 20. 11. 2018]. Dostupné z URL: <https://www.vmray.com/cyber-security-blog/sandbox-evasion-techniques-part-1>

[13] *What is android* [online]. 2011, [cit. 15. 3. 2019]. Dostupné z URL: <http://developer.android.com/guide/basics/what-is>

[14] *Metody extrakce dat z mobilního zařízení* [online]. 2019, [cit. 16. 4. 2019]. Dostupné z URL: <http://www.spy-soft.net/izvlech-dannye-android/>

[15] *What is a Raspberry Pi?* [online]. 2018, [cit. 22. 11. 2018]. Dostupné z URL: <https://opensource.com/resources/raspberry-pi>

[16] *Obrázek platformy Raspberry Pi* [online]. 2018, [cit. 25. 11. 2018]. Dostupné z URL: <https://www.modmypi.com/raspberry-pi/raspberry-pi-a-plusb-plus23-1015/rpi3-model-b-plus/raspberry-pi-3-model-b-plus/>

[17] *Technické parametry.* [online]. 2018, [cit. 22. 11. 2018]. Dostupné z URL: <https://www.raspberrypi.org/products/raspberry-pi-3-model-b-plus/>

[18] *Android Debug Bridge* [online]. 2019, [cit. 13. 4. 2019]. Dostupné z URL: <https://developer.android.com/studio/command-line/adb/>

[19] *Ghidra* [online]. 2019, [cit. 14. 4. 2019]. Dostupné z URL: <https://github.com/NationalSecurityAgency/ghidra/>

[20] *SQLite Browser* [online]. 2019, [cit. 14. 4. 2019]. Dostupné z URL: <https://sqlitebrowser.org>

[21] *Radare framework* [online]. 2019, [cit. 10. 5. 2019]. Dostupné z URL: <https://rada.re/r/>

[22] *Android application malware analysis* [online]. 2019, [cit. 18. 5. 2019]. Dostupné z URL: <https://hub.hacken.io/blog/android-application-malware-analysis/>

Seznam symbolů, veličin a zkratek

NTP	protokol pro synchronizaci vnitřních hodin počítačů po paketové síti – Network Time Protocol
API	rozhraní pro programování aplikací – Application Programming Interface
IoC	návrhový vzor – Inversion of Control
STIX	Structured Threat Information Expression
DNS	systém doménových jmen – Domain Name System
JSON	JavaScriptový objektový zápis – JavaScript Object Notation
MAC	jednoznačný identifikátor síťového zařízení – Media Access Control
CPU	centrální procesorová jednotka – Central Processing Unit
SAM	Sequence Alignment Map
ARM	Acorn RISC Machine
HAL	hardwarová abstraktní vrstva – Hardware abstraction layer
DVM	Dalvik Virtual Machine

Seznam příloh

A Ukázka výstupních zpráv analýzy podezřelého zařízení

A.1 Zpráva analýzy škodlivého souboru

Výpis A.1: První strana zprávy o analýze malwaru.

```
                    Autopsy ASCII Report

---------------------------------------------------------------
                     GENERAL INFORMATION

File: /1//FATRAT.EXE
MD5 of file: 4762d128597fd22116be0539494b3341  -
SHA-1 of file: 3b0abdc83bfa6f79ac05b897c4db9f171943d55e  -

Image: '/var/lib/autopsy/USBAnalyza/test1/images/usb1.iso'
Offset: Full image
File System Type: iso9660

Date Generated: Sat Dec  8 15:09:49 2018
Investigator: unknown

---------------------------------------------------------------
                     META DATA INFORMATION

Entry: 1
Type: File
Links: 1
Flags:
Name: FATRAT.EXE
Size: 205056
Owner-ID: 0
Group-ID: 0
Mode: -r-xr-xr-x

File Times:
Created:  2018-12-03 23:01:54 (CET)
File Modified:  0000-00-00 00:00:00 (UTC)
Accessed: 0000-00-00 00:00:00 (UTC)

```

```
Sectors:
25 26 27 28 29 30 31 32
33 34 35 36 37 38 39 40
41 42 43 44 45 46 47 48
49 50 51 52 53 54 55 56
57 58 59 60 61 62 63 64
65 66 67 68 69 70 71 72
73 74 75 76 77 78 79 80
81 82 83 84 85 86 87 88
89 90 91 92 93 94 95 96
97 98 99 100 101 102 103 104
105 106 107 108 109 110 111 112
113 114 115 116 117 118 119 120
121 122 123 124 125

File Type: Bourne-Again shell script, UTF-8 Unicode text
executable, with very long lines

---------------------------------------------------------------
            CONTENT (Non-ASCII data may not be shown)

#!/usr/bin/env bash

#################################################################
#                                                               #
#                                                               #
# THEFATRAT-1.9.6                                               #
#                                                               #
# Welcome and dont disclaimer                                   #
# TheFatRat Author By Edo -maland- { screetec }                 #
# Tested On , Backbox , kali Linux and Kali sana v.2            #
# contact me in screetsec@gmail.com or                          #
#    screetsec@dracos-linux.org                                 #
# Distro Penetration From Indonesia :                           #
#    https://dracos-linux.org/                                  #
#                                                               #
# www.github.com/screetsec                                      #
#                                                               #
#################################################################
```

A.2 Zpráva analýzy textového souboru

Výpis A.2: Zpráva analýzy textového souboru.

```
                    Autopsy ASCII Report
                                                                  1
                                                                  2
-------------------------------------------------------------     3
                     GENERAL INFORMATION                          4
                                                                  5
File: /1/TEXT.TXT                                                 6
MD5 of file: f1a264804bcca73956326a3806f6f0fc  -                  7
SHA-1 of file: 3c2c8573dcec9c6a384246e36ccccef7dd2e27a0  -        8
                                                                  9
Image: '/var/lib/autopsy/USBAnalyza/test1/images/usb1.iso'        10
Offset: Full image                                                11
File System Type: iso9660                                         12
                                                                  13
Date Generated: Sat Dec  8 15:25:33 2018                          14
Investigator: unknown                                             15
                                                                  16
-------------------------------------------------------------     17
                     META DATA INFORMATION                        18
                                                                  19
Entry: 5                                                          20
Type: File                                                        21
Links: 1                                                          22
Flags:                                                            23
Name: TEXT.TXT                                                    24
Size: 275                                                         25
Owner-ID: 0                                                       26
Group-ID: 0                                                       27
Mode: -r-xr-xr-x                                                  28
                                                                  29
File Times:                                                       30
Created:  2018-12-08 11:26:28 (CET)                               31
File Modified:  0000-00-00 00:00:00 (UTC)                         32
Accessed: 0000-00-00 00:00:00 (UTC)                               33
                                                                  34
Sectors:                                                          35
1355                                                              36
                                                                  37
File Type: UTF-8 Unicode text                                     38
```

```
                                                                  39
-------------------------------------------------------------     40
           CONTENT (Non-ASCII data may not be shown)              41
                                                                  42
Testový text obsažený v textovém souboru.                         43
Testový text obsažený v textovém souboru.                         44
Testový text obsažený v textovém souboru.                         45
Testový text obsažený v textovém souboru.                         46
Testový text obsažený v textovém souboru.                         47
Testový text obsažený v textovém souboru.                         48
-------------------------------------------------------------     49
                    VERSION INFORMATION                           50
                                                                  51
Autopsy Version: 2.24                                             52
The Sleuth Kit Version: 4.4.0                                     53
```

A.3 Kontrolní součty souborů

Výpis A.3: Generované kontrolní součty souborů na zařízení.

```
MD5 Values for files in /1/ (usb1.iso-0-0)                        1
                                                                  2
4762d128597fd22116be0539494b3341  - FATRAT.EXE                    3
85d0b4569d9d93c11cceff04b7c9e09  - SCREENSH.PNG                   4
f1a264804bcca73956326a3806f6f0fc  - TEXT.TXT                      5
```

B Obsah přiloženého CD

Na CD je obsažena elektronická verze bakalářské práce. Dále celkový výstup analýzy USB flash disku v podobě textových souborů společně s oficiální zprávou. V této zprávě jsou uvedeny informace o zkoumaném zařízení, o nalezeném obsahu, výsledky analýzy a doporučená opatření. Dále je zde obsažen kompletní program Hash Check společně s video ukázkou použití tohoto programu.

```
/ ..................................... kořenový adresář přiloženého CD
├── Bakalarska prace pdf ..............elektronická verze práce
│   └── Prostredi_pro_analyzu_podezreleho_zarizeni.pdf
├── Analyza USB flash disku
│   ├── 12-2018-malware-usb.pdf ....... oficiální zpráva o analýze
│   ├── fatrat.txt ..................... zpráva o analýze škodlivého souboru
│   ├── indexerv.txt
│   ├── screensh.txt ................... zpráva o analýze obrázkového souboru
│   ├── text.txt ....................... zpráva o analýze textového souboru
│   ├── wpsettin.txt
│   └── Program v jazyce JAVA ......... kód programu Hash Check
│       └── hashcheck_javawork.zip
└── Program Hash Check
    ├── HashCheckVideo.mkv ............ video ukázka
    ├── HashCheck.jar ................. spustitelný program
    └── HashData.txt .................. vstupní hash databáze
```