Analýza datových sad umožňujících detekci mobilního malwaru

Bc Ladislav Dorotík

Diplomová práce
2022

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav informatiky a umělé inteligence

Akademický rok: 2021/2022

ZADÁNÍ DIPLOMOVÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Bc. Ladislav Dorotík**
Osobní číslo:	**A20187**
Studijní program:	**N0613A140022 Informační technologie**
Specializace:	**Kybernetická bezpečnost**
Forma studia:	**Prezenční**
Téma práce:	**Analýza datových sad umožňujících detekci mobilního malwaru**
Téma práce anglicky:	**Analysis of Data Sets for Mobile Malware Detection**

Zásady pro vypracování

1. Vypracujte literární rešerši která zahrnuje výzkum využívající datové sady vzorků mobilního malwaru.
2. Proveďte analýzu dostupných datových sad obsahujících vzorky mobilního malwaru. A to včetně hodnocení jejich silných a slabých stránek.
3. Zpracujte vybrané sady do podoby, která umožní navazující výzkumnou činnost, tzn. odstranění duplicit, falešných a poškozených vzorků, atd.
4. Proveďte extrakci významných charakteristik mobilního malwaru z datových sad, které byly zpracovány v rámci práce.
5. Výstupem práce budou zpracované datové sady, jejich analýza a množina charakteristik mobilního malwaru, které jsou vhodné pro navazující výzkumy zabývající se detekcí mobilního malwaru.

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam doporučené literatury:

1. KLEYMENOV, Alexey a Amr THABET. Mastering Malware Analysis: The complete malware analyst's guide to combating malicious software, APT, cybercrime, and IoT attacks. Birmingham: Packt Publishing, 2019. ISBN 1789610788.
2. VANDERPLAS, Jacob T. Python data science handbook: essential tools for working with data. Sebastopol, CA: O'Reilly Media, 2016. ISBN 1491912057.
3. VELU, Vijay Kumar. Mobile Application Penetration Testing: Explore real-world threat scenarios, attacks on mobile applications, and ways to counter them. Birmingham: Packt Publishing, 2016. ISBN 9781785883378.
4. JOSHI, Prateek. Artificial Intelligence with Python [online]. Birmingham: Packt Publishing, 2017 [cit. 2021-10-20]. ISBN 9781786464392. Dostupné z: https://www.packtpub.com/product/artificial-intelligence-with-python/9781786464392
5. MUELLER, Bernhard, Sven SCHLEIER, Jeroen WILLEMSEN a Carlos HOLGUERA. OWASP Mobile Security Testing Guide: The Ultimate Guide to Mobile App Security Testing and Reverse Engineering. Maryland: OWASP, 2021. ISBN 9781257966363.

Vedoucí diplomové práce: **Ing. Milan Oulehla, Ph.D.**
Ústav informatiky a umělé inteligence

Datum zadání diplomové práce: **3. prosince 2021**
Termín odevzdání diplomové práce: **23. května 2022**

UNIVERZITA TOMÁŠE BATI VE ZLÍNĚ
Fakulta aplikované informatiky
L.S.

doc. Mgr. Milan Adámek, Ph.D. v.r.
děkan

prof. Mgr. Roman Jašek, Ph.D., DBA v.r.
ředitel ústavu

Ve Zlíně dne 24. ledna 2022

Prohlašuji, že

- beru na vědomí, že odevzdáním diplomové práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomové práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové práce bude uložen v příruční knihovně Fakulty aplikované informatiky. Univerzity Tomáše Bati ve Zlíně;
- byl/a jsem seznámen/a s tím, že na moji diplomovou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má Univerzita Tomáše Bati ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou práci nebo poskytnout licenci k jejímu využití jen připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- beru na vědomí, že pokud bylo k vypracování diplomové práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně, dne

Ladislav Dorotík, v.r

podpis studenta

ABSTRAKT

Práce se věnuje mobilnímu malwaru zaměřenému na operační systém Android. Teoretická část popisuje základy Android OS a malware, který na platformu cílí. Součástí je analýza dostupných datových sad mobilního malwaru. V praktické části jsou zpracovány vybrané datové sady do podoby umožňující další výzkumnou činnost, což zahrnuje jejich preprocessing, dekompilaci, statickou analýzu a klasifikaci. Výstupem této práce je datová sada garantované kvality, spolu s charakteristikami mobilního malwaru, které byly získány při analýze jednotlivých datových sad. Takto zpracované datasety jsou vhodné pro navazující výzkumnou činnost.

Klíčová slova: Android, Dataset, Dekompilace, Hrozby, Klasifikace, Malware, Preprocessing

ABSTRACT

This thesis includes an analysis of available mobile malware datasets. The work deals with mobile malware focused on the Android operating system. The theoretical part describes the basics of Android OS and malware that targets the platform. In the practical part, selected data sets were processed into a form enabling further research activities, including preprocessing, decompilation, static analysis, and classification. Datasets processed in this way are suitable for follow-up research activities. The output of this work is a data set of guaranteed quality, together with the characteristics of mobile malware, which were obtained during the analysis of individual data sets.

Keywords: Android, Classification, Dataset, Decompilation, Malware, Preprocessing, Threats

Nechť mé poděkování doputuje k členům laboratoře PTLAB za poskytnutí infrastruktury. Zároveň bych chtěl vyzdvihnout poděkování Ing. Milanu Oulehlovi, Ph.D., se kterým mám tu čest již několik let spolupracovat. Tato spolupráce mne obohatila nejen ve směru informačních technologií, ale i lidsky.

V neposlední řadě patří poděkování celé mé rodině, která mi vždy byla a stále je, největší oporou a to nejen ve studiu, ale i mimo něj.

OBSAH

ÚVOD

Mobilní telefony a tablety představovaly v prosinci roku 2021 57,35 % [1] celkového světového tržního podílu. Při srovnání tradičních počítačů a chytrých zařízení tvoří majoritní skupinu právě chytrá zařízení. Toto je dáno jak jejich cenovou dostupností, tak jejich využitelností.

Chytrá zařízení jako mobilní telefony je dnes možné využít na naprostou většinu základních uživatelských aktivit. Vedle svých původních schopností zprostředkovat obousměrnou hlasovou a textovou komunikaci, jsou moderní chytrá zařízení vybavena celou řadou dalších funkcí, umožňující například hrát hry, využívat elektronickou poštu a sociální sítě, číst noviny, nebo spojit se s rodinou pomocí videohovoru. Na druhou stranu ale chytrá zařízení často slouží i k jiným, než zábavním účelům. Právě činnosti jako přístup k internetovému bankovnictví, komunikace se státní správou, nebo využívání tabletů jako pracovních nástrojů (ve skladech, restauracích, pojišťovnách aj.) představují značné riziko. V zemích třetího světa, je pak podíl klasických desktopových zařízení vůči mobilním ještě nižší, než je tomu globálně, což naznačuje, že uživatelé ke zmíněným činnostem raději volí levnější mobilní zařízení, nebo tablet místo často dražších desktopových zařízení. [2]

Z celkového podílu mobilních zařízení a tabletů zastupoval na trhu s operačními systémy většinu Android OS. Celosvětově v březnu roku 2022 platforma Android OS pokrývala 70,65 % trhu a spolu se 28,64 % zastoupení operačního sytému iOS tvořila 99,29% pokrytí trhu. [3] Zbylá zařízení běžela na operačních systémech, které již často nejsou podporovány. V květnu roku 2021 navíc společnost Google v rámci uvedení beta verze Android 12 oznámila, že na světě je aktuálně osazeno operačním systémem Android více, než 3 miliardy zařízení. [4] Převaha Android OS je dána zejména tím, že jde o open-source software, což má za následek, že jej různí výrobci mohou adaptovat do svých zařízení, která lze prodávat na trhu, na rozdíl od operačního systému iOS, který smí oficiálně distribuovat pouze firma Apple jako součást svých produktů.

Zmíněná fakta jako otevřenost systému, celkový podíl na trhu a široké pole uživatelské působnosti jsou do jisté míry výhodou, nicméně představují i značné riziko. Právě tyto klíčové vlastnosti Android OS totiž často lákají i tvůrce malwaru. Tato diplomová práce vznikla, jako reakce na aktuální situaci a proto je její součástí i shrnutí bezpečnostní situace na poli mobilního malwaru a s tím související témata, jako: sběr škodlivých aplikací, některé techniky skrývaní před detekcí a nebo zneužití komponent aplikací.

Práce shrnuje aktuální stav výzkumů, využívajících dostupné datové sady mobilního malwaru. Součástí práce je nejen nalezení aktuálně dostupných datových sad, ale i základní seznámení s platformou Android OS.

Nalezené datové sady byly analyzovány se záměrem zjistit jejich silné a slabé stránky. Zároveň s analýzou klíčových vlastností datových sad proběhlo i zpracování vzorků a to do podoby, která umožňuje následující výzkumnou činnost. Výslednou datovou sadu (bez duplicit, falešných souborů a dalších nežádoucích jevů) je následně možné použít k dalšímu výzkumu. Jako demonstraci, že datovou sadu je opravdu možné při výzkumu použít, byla provedena dekompilace a extrakce významných charakteristik mobilního malwaru, tedy automatizovaný proces statické analýzy - tzv. extrakce atributů, které se věnuje zejména praktická část. Jednotlivé vzorky malwaru byly navíc ohodnoceny nástrojem VirusTotal [5], což zaručuje integritu dat ve smyslu jejich malignity.

Praktická část práce klade celkový důraz na datové sady, jejich kvalitu a rozmanitost. Spíše než samotnou aplikací umělé inteligence, se zabývá zpracováním a přípravou dat právě pro nasazení metod inteligentních výpočtů.

I. TEORETICKÁ ČÁST

1 AKTUÁLNÍ STAV ŘEŠENÉ PROBLEMATIKY

V roce 2021 byl publikován výzkum [6] jehož součástí je unikátní analýza 81 odborných článků zaměřených na problematiku malwaru pro operační systém Android. Tyto byly vybrány z osmi bezpečnostních konferencí. Autoři výzkumu zde shrnují aktuální situaci, kdy je evidentní, že většina prací se zaměřuje především na samotnou detekci mobilního malwaru, nebo jeho analýzu a statistiky. Další práce se poté věnují tvorbě vlastních nástrojů a klasifikaci rodin mobilního malwaru, jak je patrné z tabulky 1.1. Právě malware family classification, tedy určování rodiny malwaru je významnou částí zmíněného výzkumu a je to i jeden z důvodů, proč je kritizován nástroj VirusTotal. Navzdory kritice kvůli limitům VirusTotalu, sami autoři výzkumu zmiňují, že zatímco dřívější datasety mobilního malwaru např. MalGenome [7], bylo možné vzhledem k menší velikosti analyzovat manuálně, je u dnešních datových sad nutné automatizované zpracování.

Výzkumný záměr	Počet článků
Detekce	34
Analýza/Měření (překlad z Measurement)	23
Nástroje	21
Klasifikace rodin	8
Jiný	5

Tabulka 1.1 Výzkumný záměr článků [6] (přeloženo)

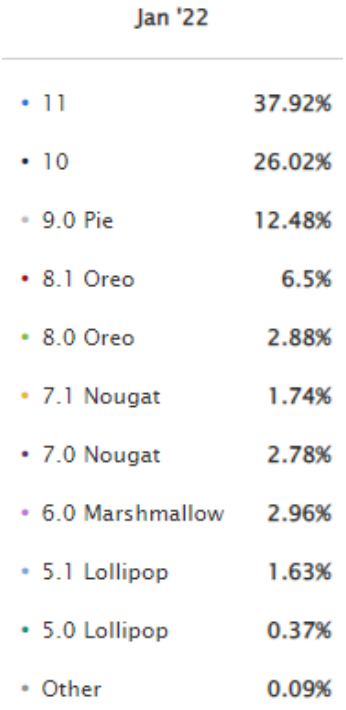
V rámci analýzy také autoři kladli důraz na používané datasety. Podařilo se identifikovat více, než 30 datových sad. Je však nutné upozornit, že některé z uvedených datasetů již v roce 2022 nejsou dostupné - například AMD: Android Malware Dataset [8], [9] a Genome: Android Malware Genome [7], [10]. Výzkum ukázal, že používané datové sady se často opakují a dokonce mohou mít i neznámý původ. Zřídka jsou pro výzkumné účely tvořeny datové sady vlastní. Z analýzy dále vyplývá, že velké množství prací se zakládá na datových sadách, které obsahují vzorky z let 2009-2013. [6]

Z uvedených skutečností vyplývá, že situace datových sad, které jsou používány v rámci výzkumů není ideální. Tento trend lze pozorovat i u kvalifikačních prací, které nejsou součástí odborných bezpečnostních konferencí. To potvrzuje například, jinak velmi kvalitně zpracovaná bakalářská práce [11], jejíž cílem bylo vytvoření nástroje pro extrakci atributů, avšak datová sada, která byla v rámci práce použita je specifikována jako poskytnuta profesorem, obsahující 1000 benigních a 1000 maligních vzorků. Z dostupných informací tak kupříkladu není jasné, z jakého období pochází použité vzorky. Pro účely zmíněné práce to sice nepředstavuje zásadní problém, nicméně pro budoucí úspěšné nasazení umělé inteligence je zajištění kvality a znalosti datové sady

klíčové. [12]

Vzhledem k tomu, že operační systém Android a jeho funkcionality se s novými verzemi mění, je zřejmé, že používání obsoletních vzorků malwaru je spíše kontraproduktivní, než efektivní. S operačním systémem Android 6.0 (Marshmallow) byl navíc v roce 2015 mimo jiné změny představen nový model tzv. runtime oprávnění, který má zásadní vliv jak na analýzu, tak na chování aplikací. [13]

Zároveň je celkový výskyt zařízení s operačním systémem starším, než verze Marshmallow procentuálně velmi nízký, jak prezentuje obrázek 1.1. To však neznamená, že není potřeba tato zařízení chránit. Je-li celkový počet zařízení s Android OS roven třem miliardám, činí součet zařízení se staršími verzemi, než 6.0 až 62,7 milionů. Na druhou stranu se však útočníkům mohou jevit jako lákavější cíle jiné obsoletní verze, které pokrývají mnohem větší počet zařízení. Dle komunitně spravované služby endoflife.date, která dokumentuje projekty, jimž končí oficiální podpora (tzn. že vycházejí bezpečností aktualizace a další modifikace), je pak poslední podporovanou verzí Android 10. [14] Tomu nasvědčuje i bezpečnostní přehled, kdy poslední bezpečnostní úprava pro Android 9 proběhla v lednu roku 2022, následně již bezpečnostní opravy ovlivňují pouze Android verze 10, 11 a 12. [15] Pokud by tedy potenciální útočník chtěl cílit na specifické nepodporované verze, velmi pravděpodobně by si vybral tu nejpočetnější, kterou je verze 9.0, součet případných obětí by pak dosahoval hodnoty přes 374 milionů.



Obrázek 1.1 Podíl verzí Android OS [16]

Alarmující je údaj, vyplývající z výše uvedených skutečností a sice, že 31,34 % zařízení využívá nepodporované verze Android OS. [16] Pokud tedy při uvedení beta verze Androidu 12 [4] uvedl autor reálné informace (tedy více než tři miliardy zařízení s Android OS, zaokrouhleno na 3 mld.), znamenalo by to teoreticky až 940 200 000 zařízení, kterým již nevycházejí bezpečnostní aktualizace. Do součtu nebyla zařazena kategorie "Other" z obrázku 1.1, jelikož by teoreticky mohla zahrnovat i již zmíněný Android 12 Beta.

Naopak příspěvek [17], je založen na vlastní datové sadě. Kanadský institut pro kybernetickou bezpečnost (Canadian institute for Cybersecurity, dále jen CIC) se zabývá tvorbou datových sad zaměřených na kybernetickou bezpečnost a právě datová sada vytvořená v [17] je poskytnuta veřejně pro výzkumníky na webových stránkách CIC. [18] Na rozdíl od datasetů s neznámým původem, zde autoři poskytují kromě dat z analýzy i kvazi kontextuální atributy. Díky tomu je možné rozlišit druh vstupní aplikace a sice do pěti kategorií: Adware, Banking malware, SMS malware, Riskware, a Benign. Dalším důležitým faktorem je zde znalost období ve kterém byly vzorky sbírány, tedy prosinec 2017 - prosinec 2018. Je však nutné zmínit, že vzorky sbírané v roce 2017 mohou pocházet i z let předešlých. Mezi kontextuální data lze zařadit i informaci o původu jednotlivých vzorků, tedy z jakého zdroje pochází. Problém je zde však ten, že veřejně dostupná data zahrnují pouze informaci o tom, z jakých datových sad autoři vzorky čerpali. To znamená, že tuto informaci nelze přímo přiřadit k jednotlivým aplikacím. V rámci praktické části pak byly zjištěny další nežádoucí jevy v datové sadě, například duplicity. Celkově datová sada čerpá z několika zdrojů, je tedy zajímavé, že obsahuje méně vzorků, než pramen, ze kterého čerpá: AMD - Android Malware Dataset. [8]

Zajímavý je také přístup k analýze, kdy z článku [17] vyplývá, že analýza byla prováděna jak statická, tak dynamická. Avšak data byla sbírána dynamicky, to znamená, že pro sběr dat bylo nutné aplikaci nahrát do emulátoru. Právě interakce aplikací s emulátorem může představovat problém. Komplikace mohou způsobit jak nekompatibilní verze, tak i anti-analysis techniky, které útočníci využívají právě ke ztížení analýzy. Sami autoři uvádí také různé poškození aplikací a nevalidní vzorky. Tento přístup tak vedl k maximalizaci zisku informací z jednotlivých aplikací, na druhou stranu však vedl také ke ztrátě až 5 743 aplikací, které mohly být analyzovány alespoň staticky.

Historicky tak v roce 2012 vznikla pravděpodobně první akademické datová sada Android Malware Genome [7], která byla ve své době jedinečná a svým manuálním zpracováním je unikátní i do dnes. Tato datová sada obsahovala 1260 vzorků a byla označována jako jediná kvalitně popsaná. V roce 2015 však bylo její sdílení ukončeno. Vzorky již dostatečně nereprezentovaly moderní mobilní malware, protože pocházely z období let 2009-2012. [6], [8] V reakci na vzniklou situaci byl v roce 2017 vydán již zmiňovaný dataset AMD, který si kladl za cíl mobilní malware popsat ještě detailněji

ve smyslu objevených rodin malwaru a jejich charakteristik. Datová sada čítala 24 650 vzorků malwaru, který byl rozřazen do 71 rodin. [8] Vzorky obsažené v sadě pocházely z období 2010-2016. [6] Stejně jako Genome i AMD je již nedostupný, rozdílem mezi zmíněnými sadami je ale fakt, že autoři projektu Genome alespoň uvedli důvod ukončení sdílení, kdežto projekt AMD k ukončení neposkytuje bližší informace.

Zatím poslední datovou sadou tohoto typu je již zmíněný dataset od CIC: CICMalDroid2020 [17]. V průběhu let vznikaly i jiné datové sady [6], často byly však velmi specificky zaměřeny, viz. [19] - zaměření na botnet, nebo se jedná o různé deriváty dostupných sad. Za zmínku však stojí i udržovaný projekt AndroZoo [20], který je zajímavý hlavně svou velikostí, protože obsahuje přes 18 milionů aplikací, kde v roce 2021 bylo 2,24 milionů z nich označeno jako malware. [6] S tak obrovským množstvím vzorků je ale nutné umět pracovat. Pakliže by byla náhodně vybrána část vzorků, nemusí být dostatečně reprezentativní, může například, obsahovat jen specifický typ malwaru. Na druhou stranu zpracování všech 18 milionů aplikací by bylo velmi zdlouhavé a přináší další potenciální problémy, jako velikost úložiště a další.

Tato práce si klade za cíl zlepšit a popsat situaci na poli datových sad, ověřit jejich vlastnosti a reagovat tak na aktuální problémy s veřejnými datovými sadami mobilního malwaru.

2 VÝZKUM ZAMĚŘENÝ NA ANDROID OS

Android je open-source platforma, založená na Linuxu a vyvíjená společností Google. Původně byla platforma využívána primárně jako mobilní operační systém, postupem času se však její pole působnosti značně rozšířilo. Kromě mobilních telefonů a tabletů jsou osazovány tímto operačním systémem chytré televize, nositelná zařízení (wearables) a další smart technologie. [21] Výjimkou však nejsou ani automobily, které mohou být vybaveny tabletem s Android OS, místo klasického autorádia.

Zařízení běžící na Android OS bývají distribuovány s již předinstalovanými aplikacemi a podporují instalaci aplikací třetích stran, ať už se jedná o ty z oficiální distribuční platformy Google Play Store a nebo z jiných zdrojů. Tyto internetové trhy aplikací bývají často označovány, jako marketplace - tedy tržště. [21]

Jak ukázala studie, na které se podílela antivirová společnost Norton [22], tato tržiště jsou signifikantním zdrojem mobilního malwaru. V souvislosti se stažením škodlivé aplikace však bývají často spojovány tržiště třetích stran, nicméně jak vyplývá z tohoto výzkumu je hlavním zdrojem mobilního malwaru právě oficiální distribuční platforma Google Play Store. Vůči tomuto faktu se ohradila i samotná společnost Google a tak na základě tohoto autoři zavedli tzv. VDR - vector detection ratio. Tento ukazatel je důležitý zejména proto, že klade důraz na celkový objem stažených aplikací vůči těm škodlivým. Tento ukazatel tedy říká, že ačkoliv je Google Play Store zdrojem 67% všech nechtěných aplikací, je také hlavním zdrojem pro stahování aplikací jako takových. Protože právě z oficiální platformy pochází také 87% všech distribuovaných aplikací, činí VDR Google Play Store 0.6%. Z toho vyplývá, že většina škodlivých aplikací pochází z Google Play Storu, navzdory tomu se však jedná o nejbezpečnější variantu stahování aplikací. [22]

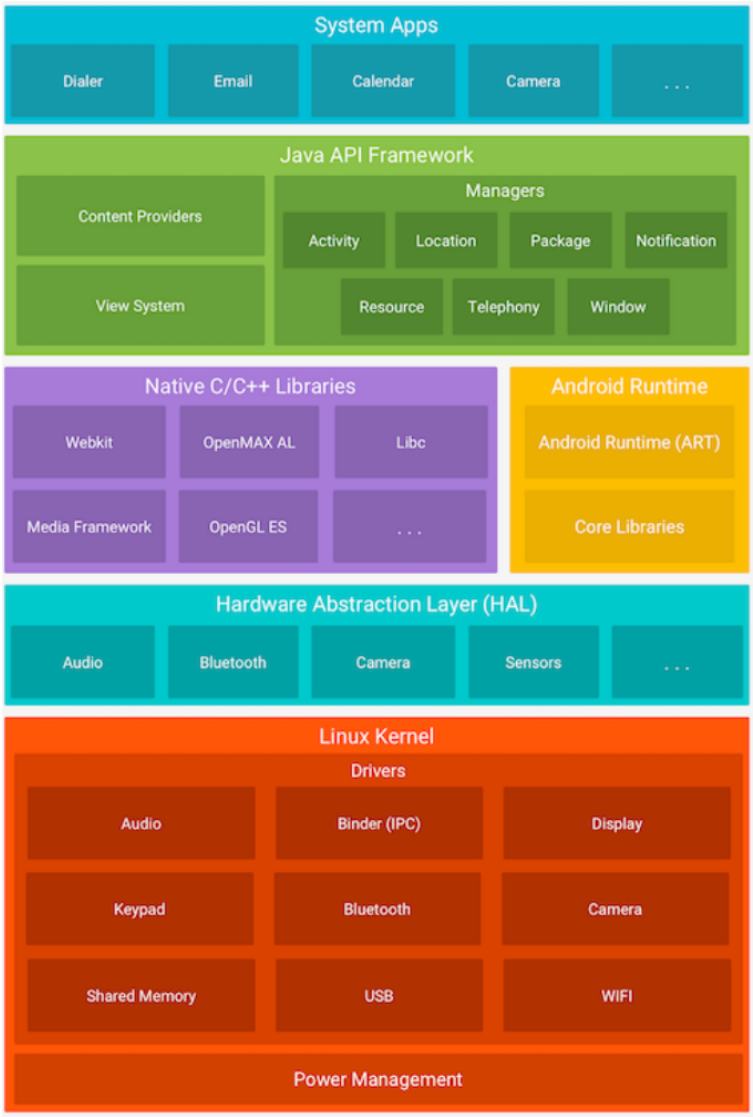
	Installs		Installer					Children			
Vector	**All**	**Unw.**	**All**	**Unw.**	**Plat.**	**Pkg.**	**Sig.**	**Pkg.**	**Sig.**	**VDR**	**RVDR**
Playstore	87.2%	67.5%	10	3	0	2	9	1.2M	816K	0.6%	1.0
Alt-market	5.7%	10.4%	102	31	15	87	67	128K	77K	3.2%	5.3
Backup	2.0%	4.8%	49	2	24	31	39	528K	355K	0.9%	1.5
Pkginstaller	0.7%	10.5%	79	5	25	11	74	197K	127K	2.4%	4.0
Bloatware	0.4%	6.0%	54	2	28	37	41	2.1K	1.3K	1.2%	2.0
PPI	0.2%	0.1%	21	0	2	20	11	1.5K	1.3K	0.3%	0.5
Fileshare	<0.1%	<0.1%	13	3	4	13	11	8.8K	7.4K	1.3%	2.1
Themes	<0.1%	<0.1%	2	0	2	2	2	634	14	0.3%	0.5
Browser	<0.1%	<0.1%	47	4	3	40	38	4.8K	3.3K	3.8%	6.3
MDM	<0.1%	<0.1%	7	1	1	7	6	766	489	0.3%	0.5
Filemanager	<0.1%	<0.1%	58	11	9	32	43	6.6K	4.7K	2.6%	4.3
IM	<0.1%	<0.1%	13	2	0	10	11	2K	1.2K	2.9%	4.8
Other	<0.1%	0.3%	151	68	28	125	98	9.1K	5.3K	3.9%	6.5
Unclassified	3.7%	<0.1%	3.5K	2.4K	386	3.3K	814	91K	16K	<0.1%	0.1
All	100.0%	100.0%	4.2K	2.5K	79	3.6K	1.0K	1.6M	992K	1.6%	2.6

Obrázek 2.1 Shrnutí distribuce aplikací [22]

Za bezpečné však nelze automaticky považovat ani zmíněné předinstalované aplikace, jak ukazuje bezpečností zpráva společnosti Lookout, která upozorňuje na zranitelnosti v předinstalovaných aplikacích na zařízeních Samsung. Tyto zranitelnosti mohou mít za následek neoprávněné čtení i zápis na zařízení oběti, což útočníkům umožní měnit nastavení zařízení, manipulaci s kontakty, hovory, nebo třeba zprávami. [23]

Pro vývojáře i bezpečnostní výzkumníky je tedy důležité porozumět architektuře operačního systému Android, které se anglicky nazývá Android software stack (viz obrázek 2.2). Již zmíněné aplikace, interagují přímo s uživateli, proto zde představují jakousi špičku ledovce a jsou tak umístěny ve stacku na vrcholu celého systému. Naopak linuxové jádro, které je základem celého systému, představuje nejbližší element ve vztahu k hardwaru a ve stacku je tedy umístěn jako první.

2.1 Architektura Android OS



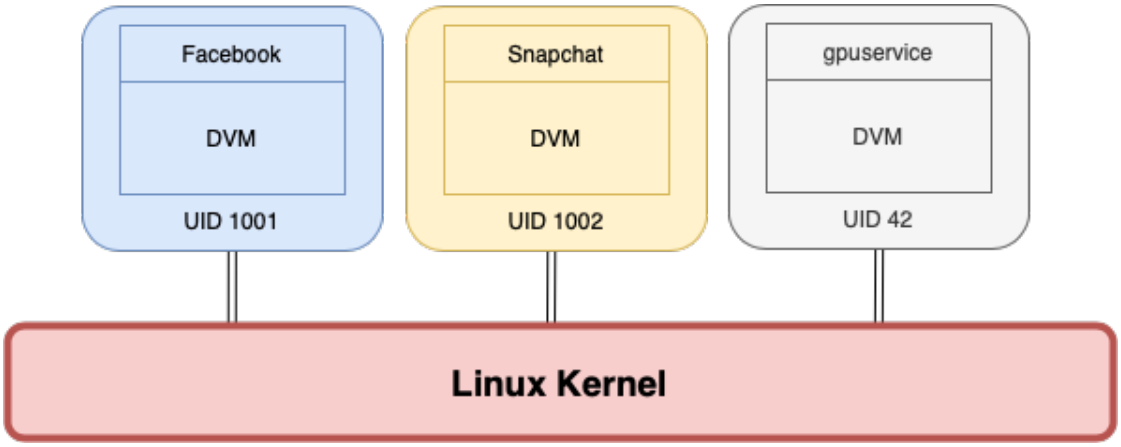
Obrázek 2.2 Android Software Stack [24]

2.1.1 Linux Kernel

Srdcem Android OS je linuxové jádro, to bylo zvoleno zejména kvůli své přenositelnosti mezi různými hardwarovými komponenty, což umožňuje snadnou kompilaci programů na různých zařízeních. Jádro obstarává základní funkce operačního systému, jako jsou správa paměti, nebo plánování a management systémových procesů. Představuje však také důležitou vrstvu tím, že poskytuje přístup k ovladačům. [25] Použití linuxového jádra tak znamená, že výrobci zařízení mohou vyvíjet ovladače hardwaru pro již známé jádro, což značně usnadňuje nejen samotný vývoj, ale i nasazení. [24]

Linux Security: Z bezpečnostního hlediska kernel umožňuje Androidu převzít klíčové bezpečnostní funkce. V praxi to znamená, že Android využívá tzv. mandatorní řízení přístupu nad všemi procesy (přístup procesu k objektu podléhá autorizačním pravidlům) a k vymáhání využívá Security-Enhanced Linux (SELinux) model. Tento model je založen na základním principu, kdy vše co není povoleno je ve výchozím stavu zakázáno. Pokud tedy proces - aplikace, chce číst kontakty, musí požádat o příslušné oprávnění. Pakliže uživatel, nebo operační systém nepovolí procesu využít oprávnění ke čtení kontaktů, aplikace nemůže tyto zdroje využít. [26]

Aplikační sandbox Každé aplikaci v Android OS je vytvořen vlastní uživatel a je jí přířazeno UID - identifikátor uživatele. Aplikace běží jako individuální proces. Z toho vyplývá, že jednotlivé aplikace - procesy jsou od sebe izolovány, jak je patrné z obrázku 2.3. Ve výchozím stavu mezi sebou aplikace nemohou interagovat a mají omezený přístup k operačnímu systému. Tímto způsobem Android implementuje tzv. Aplikační sandbox. [26]



Obrázek 2.3 Aplikační sandbox [27]

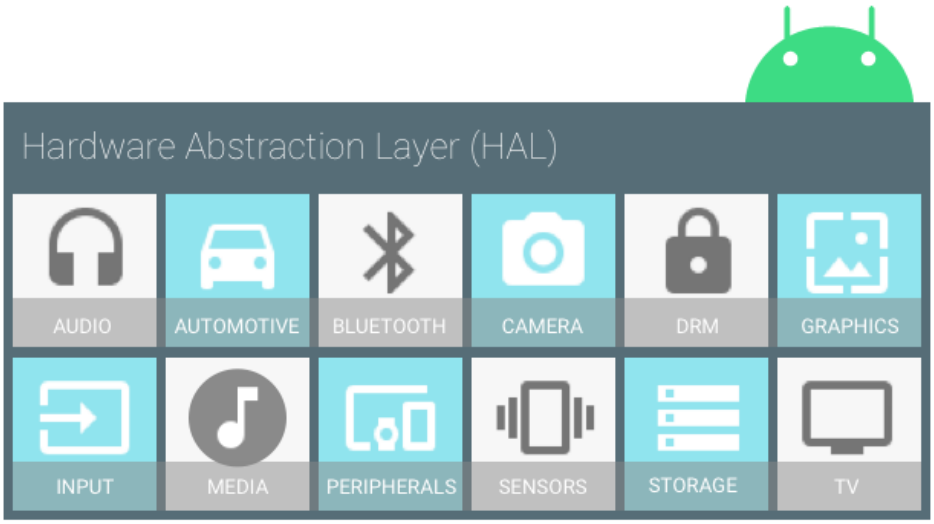
Klíčové bezpečnostní mechanismy plynoucí z linuxového jádra, dle AOSP [28]:

1. Izolace procesů
2. User-based model oprávnění
3. Komunikace mezi procesy (IPC)

2.1.2 Abstraktní vrstva HAL

HAL, tedy Hardware Abstraction Layer definuje standartní rozhraní pro interakci se zabudovaným hardwarem. Rozhraní implementuje tvůrce hardwaru, tato rozhraní jsou následně zabalena do tzv. shared library modulů (soubory s příponou ".so"), které jsou volány systémem Android, když je vyžadována interakce s některým ze zabudovaných prvků, např. kamerou. Vrstva umožňuje využití hardwarových komponent různých výrobců bez nutnosti úpravy samotného operačního systému. Pro vývojáře to znamená, že nemusí brát v úvahu typ kamery, nebo jiná specifika. Je tak možné, aby stejný zdrojový kód napsaný programátorem, běžel na dvou zařízeních, které používají jiný hardware a to bez nutnosti upravovat zdrojový kód aplikace. [21]

Do verze Android 9 se k implementaci rozhraní využíval jazyk HIDL (HAL interface definition language), ten je však od verze Android 10 zastaralý, proto Android migruje tato rozhraní do jazyka AIDL (Android Interface Definition Language), které jsou zabaleny v souborech s příponou ".aidl". [29] Z těchto souborů, avšak v binární podobě, jsou následně tvořeny konstrukce v C++, nebo Javě. [30] Úkolem výrobců hardwaru je tedy vytvoření “.aidl” souboru, o zpracování rozhraní se poté stará samotné zařízení.



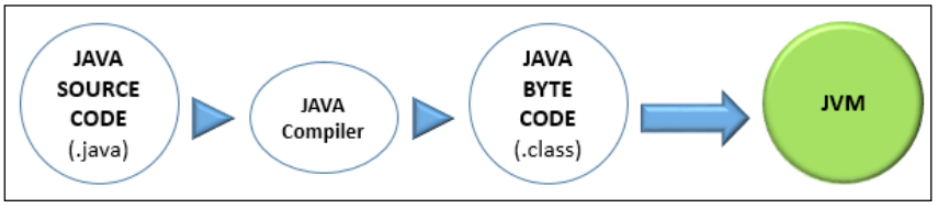
Obrázek 2.4 Komponenty HAL [31]

2.1.3 Běhové prostředí Android (Dalvik a ART)

Pro zařízení osazená Androidem verze 5.0 (API level 21 a vyšší), každá aplikace běží ve svém vlastním procesu a zároveň s vlastní instancí běhového prostředí - Android Runtime (ART). Toto prostředí bylo navrženo speciálně pro potřeby operačního systému Android. ART byl napsán tak, aby umožnil běh několika virtuálních prostředí najednou a to i na zařízeních s omezenými zdroji. ART pracuje s balíčky APK, potažmo soubory s příponou ".dex", což je formát byte kódu navržený speciálně pro operační systém Android, který je optimalizován tak, aby kladl minimální paměťové nároky na systém. [24] Tento spustitelný soubor je zkompilovaný zdrojový kód a je možné do něj zkompilovat, jak zdrojový kód napsaný v jazyce Java, tak i Kotlin. Jazyk Kotlin je totiž kompilován do byte kódu a následně do formátu ".dex". [21], [32]

Dalvik je stejně jako ART virtuální prostředí. Jedná se o předchůdce Android Runtime, kdy jsou však obě prostředí částečně kompatibilní, jelikož pracují se zmiňovanými ".dex" soubory a APK balíčky. V praxi platí, že aplikaci původně vyvíjenou pro Dalvik lze spustit v Android Runtime, naopak to nemusí být pravda. Jelikož se dá považovat Dalvik za zastaralý, protože byl od Androidu 5.0 plně nahrazen ART, nepředstavuje tato situace zásadní komplikace. [33]

JVM - Java Virtual Machine Programy napsané v klasické Javě jsou multiplatformní, jelikož kompilovaný kód Javy běží ve virtuálním stroji, který se jmenuje Java Virtual Machine. Celý proces pak demonstruje obrázek 2.5. Z diagramu je patrné, že zdrojový kód je kompilován do Java byte kódu, což je vlastně nativní sada instrukcí pro virtuální stroj.

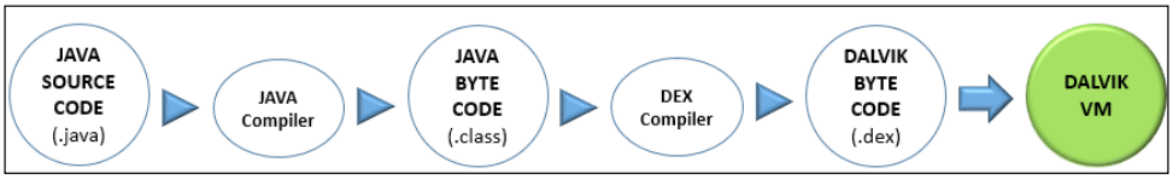


Obrázek 2.5 Proces kompilace: Java pro JVM [25]

DVM - Dalvik Virtual Machine Java používaná v Androidu se však liší od klasické Javy SE. Od standardní Javy SE se Java pro Android odlišuje tzv. core knihovnami, toto zahrnuje knihovny specifické pro DVM - Dalvik Virtual Machine, ale také knihovny zajišťující interoperabilitu. [25]

Do procesu kompilace Javy pro Android OS tak vstupuje již zmiňovaný soubor ".dex". Začátek kompilace je totožný s postupem u JVM, to znamená, že je nejprve

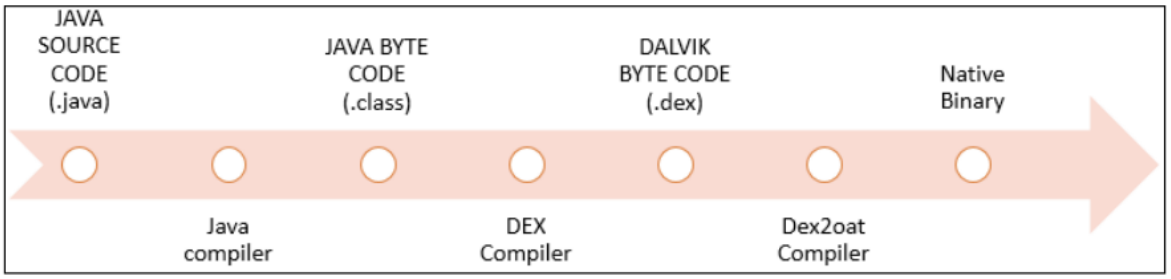
zkompilován zdrojový kód do byte kódu Javy. Tento bytecode pro JVM je následně znova kompilován do formátu dex, jak ukazuje obrázek 2.6. Tato kompilace je orientována na optimalizaci pro běh na mobilních zařízeních, takže patřičně nakládá s redundantními informacemi a využívá kompresních technik. Výsledkem je spustitelný soubor formátu dex - Dalvik Executable, což je opět byte kód, tedy sada instrukcí. Nikoliv však pro JVM, nýbrž pro DVM. [25]



Obrázek 2.6 Proces kompilace: Java pro DVM [25]

ART - Android Runtime Již bylo zmíněno, že jak ART, tak DVM využívají dex soubory. Hlavní rozdíl je však v tom, jak s dex soubory nakládají. Z obrázku 2.7 je patrné, že byte kód formátu dex je pro potřeby ART ještě jednou kompilován. Tato kompilace využívá nástroj dex2oat, zabudovaný v operačním systému Android. [34]

Vše spočívá v tom, že při instalaci aplikace, ART pomocí dex2oat ze vstupního souboru vygeneruje jeden, či více kompilovaných souborů, které jsou načteny prostředím ART. Tento typ kompilace se nazýva AOT: Ahead-of-time a nahrazuje JIT: Just-in-time kompilaci (při každém spuštění je aplikace profilována a často používané bloky byte kódu jsou kompilovány do kódu nativního). Počet a typy vygenerovaných souborů se s verzemi mění, například Android 8.0 generuje soubory: ".odex", ".vdex" a ".art". [34]



Obrázek 2.7 Proces kompilace: Java pro ART [25]

.odex: Obsahuje metody kompilované AOT. [34]

.vdex: Obsahuje nekomprimovaný DEX kód aplikace s metadaty pro zrychlení ověření [34]

.art (fakultativní): Využíván ke zrychlení spuštění aplikace, obsahuje reprezentace tříd a řetězců specifických pro ART. [34]

ART - Hybridní kompilace Reálný scénář moderních zařízení však kombinuje AOT a JIT. Tyto scénáře se mohou lišit napříč zařízeními, jelikož je možné tyto konfigurace upravovat. Zároveň mají značně pozitivní vliv na rychlost aplikací. Tento přístup je možné využít na zařízeních s Android OS verze 7.0 a vyšší.

Reálný scénář na zařízení Google Pixel [34]:

1. Prvotní instalace aplikace proběhne bez AOT kompilace. Několik prvních spuštění je tak aplikace interpretována a často používané metody jsou kompilovány jako JIT.

2. Když se zařízení nabíjí a je v nečinném stavu, spouští se kompilační daemon. Na základě profilu prvních spuštění je často využívaný kód AOT-kompilován, tomuto kódu se přezdívá profile-guided (řízen profilem) a je tvořen právě kompilačním démonem.

3. Při dalším spuštění aplikace využívá profile-guided kód (AOT). Tímto se vyhne JIT kompilaci za běhu u metod, které již byly kompilovány. Během dalších spouštění aplikace se kód kompilovaný pomocí JIT a často používané metody, jež ještě nebyly kompilovány pomocí AOT, přidává do profilu aplikace. Tyto metody jsou následně opět zpracovány kompilačním daemonem, jako v kroku 2.

Mezi hlavní výhody ART tak patří: hybridní kompilace, vylepšený garbage collector (správa paměti) a tím rychlejší běh i zavádění aplikací. [33]

2.1.4 Nativní C/C++ knihovny

Mnoho základních komponent a služeb operačního systému Android je tvořeno nativním kódem. Mezi tyto komponenty patří například ART a HAL. Nativní kód však vyžaduje nativní knihovny, které jsou psány v programovacím jazyce C, nebo C++. Přístup k funkcím z těchto knihoven zajišťuje Android OS pomocí Java API frameworku, který je standardně využíván vývojáři mobilních aplikací. Díky tomu je ku příkladu možné v aplikaci využívat 2D a 3D vykreslování. K vykreslování v Androidu slouží nativní knihovna OpenGL ES, pro přístup k této knihovně je poskytnuto rozhraní Java OpenGL API. [24] Dle AOSP představují nativní knihovny pro legitimní aplikace dva hlavní způsoby využití. [35]

1. Získání vyššího výkonu zařízení za účelem snížení latencí, nebo vykonání výpočetně náročných operací (vykreslování grafiky ve hře).

2. Znovupoužitelnost vlastních, nebo cizích knihoven psaných v C/C++.

Znovupoužitelnost knihoven je využívána například v situaci, kdy je aplikace distribuována ve dvou verzích: placená a zdarma. Pakliže aplikace využívají stejné základní komponenty, je možné tyto vytvořit v nativním kódu a sdílet mezi aplikacemi. [36] Lze tak implementovat specifické funkce pouze pro jednu aplikaci, ale zároveň není vyžadováno psát celou aplikaci od začátku.

Znamená to však, že aplikace pro Android může obsahovat i zkompilovaný kód v jazyce C/C++. Tato vlastnost ale může být využita i pro škodlivé účely. Soubory formátu dex je možné reverzním inženýrstvím dekompilovat do Javy a následně analyzovat čitelný kód. Ačkoliv je možné zpětně analyzovat i zkompilované nativní knihovny, často je nutné analyzovat již kompilovaný binární kód, což není tak obvyklá sada dovedností, jako analýza zdrojového kódu v Javě. Tohoto zneužívají tvůrci malwaru, tak aby bezpečnostním analytikům ztížili analýzu aplikace. [37]

2.1.5 Java API Framework

Rozhraním pro programování je Java API Framework, proto je také naprostá většina kódu pro Android OS analyzována právě v Javě. Jak již bylo zmíněno, operační systém Android implementuje vlastní API, aby umožnil interakci s operačním systémem, potažmo komponenty zařízení, což je nutné pro korektní fungování aplikací. V praxi je tak část API stejná, například třída "System". Druhá část API je potom jednak upravená, a jednak implementuje i specifické funkce unikátní pro operační systém Android. Typickým zástupcem druhé části API je třída "DexClassLoader", která je využívána ke spouštění kódu, jenž nebyl součástí aplikace. [26] Obrázek 2.8 ukazuje zneužití třídy "DexClassLoader" malwarem Android Rootnik.

```
if(!v4.exists()) {
    return v9;
}

Dwol.chmod(v4.getAbsolutePath());
String v8 = this.cxt.getDir(Bow.oDex, 0).getAbsolutePath();  // Bow.oDex="outdex"
Dwol.chmod(v8);
e. DexClassLoader v3 = new DexClassLoader(v4.getAbsolutePath(), v8, null, this.cxt.getClassLoader());
v4.delete();
```

Obrázek 2.8 Ukázka zneužití Java API Frameworku [38]

Analytici mobilního malwaru se již běžně setkávají s tímto specifickým typem malwaru, kterému se přezdívá packed malware, tedy zabalený malware. Základní princip je takový, že škodlivý soubor typu dex je dodatečně dodán legitimní aplikaci, ať už byl stažen z internetu, nebo ukryt v samotné aplikaci, například v adresáři "assets". Soubor typu dex je následně dynamicky rozbalen třídou "DexClassLoader".

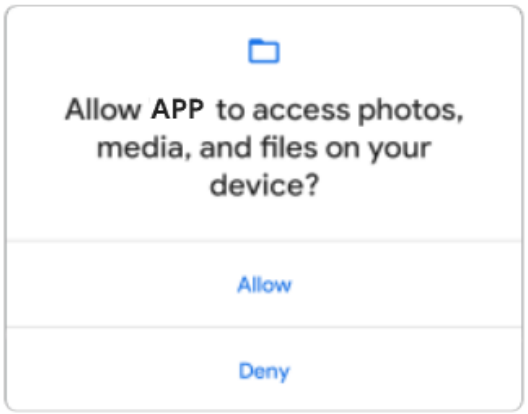
Tento proces je nazýván 'malware unpacking' a je využíván, jak pro ztížení reverzního inženýrství, tak jako technika skrývání, tzv. evasion technique. Důvod skrývání je

vcelku elementární a sice předcházení detekci na distribučních platformách a zařízení. Tento útok může být kombinován např. s Command and Control servery a Domain Generation Algoritmy, což umožňuje útočníkům vzdáleně manipulovat se zařízením pomocí šifrovaných instrukcí. [38], [39]

2.2 Model oprávnění

O tom jaká data a funkcionality budou jednotlivým aplikacím zpřístupněny rozhoduje tzv. permission model. Jeho hlavním účelem je ochrana uživatele a jeho soukromí.

Před uvedením Androidu 6.0 byla všechna oprávnění aplikaci přidělena uživatelem při instalaci, ve verzích 6.0 a výše je však používán nový model oprávnění - runtime permissions. V praxi tak aplikace vyžaduje oprávnění pouze za běhu, např. při přístupu aplikace k fotografiím, je uživateli zobrazena výzva k přijetí, či odmítnutí oprávnění, toto je zobrazeno na obrázku 2.9. Oprávnění při instalaci však nevymizela, pouze byl upraven způsob jejich použití tak, že jsou akceptována automaticky. Všechna oprávnění která jsou aplikací vyžadována, musí být obsažena v manifestu aplikace. [26]



Obrázek 2.9 Výzva k udělení runtime oprávnění [40]

Oprávnění je tak dle oficiální dokumentace Androidu možné rozdělit na dva základní typy. [40]

1. Install-time permissions. [40]
    - Oprávnění, která jsou aplikacím přidělena automaticky při instalaci.
    - Signature permissions:
        - Oprávnění definována jinou aplikací, mající stejný certifikát.
    - Normal permissions:
        - Oprávnění, která dle [40] představují velmi malé riziko pro uživatele, či běh jiných aplikací.

– Zahrnuje například: Přístup k Bluetooth, manipulace s Wi-Fi a mobilním internetovým připojením.

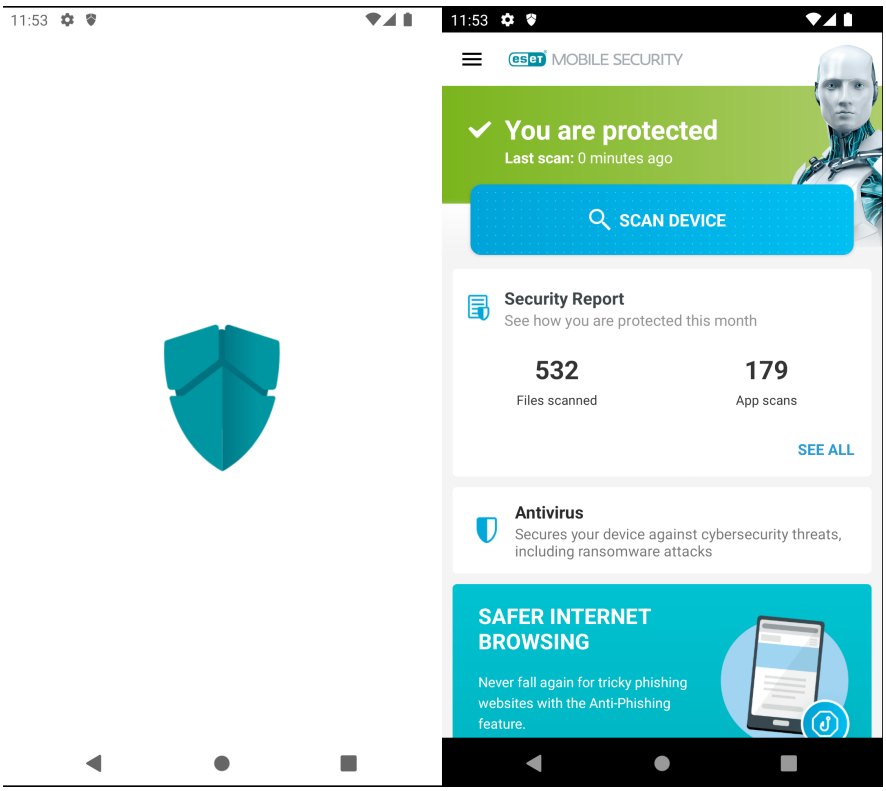
2. Runtime permissions. [40]

- Oprávnění manuálně schvalovaná uživatelem, představující potenciální bezpečnostní riziko. [40]
- Jsou shlukována do skupin, pakliže je udělen přístup k jednomu oprávnění z dané skupiny, je automaticky zpřístupněna celá skupina. [26]

2.3 Komponenty aplikace

Oficiální dokumentace pro Android OS rozlišuje čtyři základní typy komponent aplikace a to: Aktivity (Activities), Služby (Services), Broadcast Receivers a Content Providers. Tyto komponenty představují základní stavební kameny, které definují aplikaci. [42] Autoři OWASP Mobile Security Testing Guide přidávají ještě Fragment (Inter-Process Communication) a Intent. [21]

2.3.1 Activity



Obrázek 2.10 Ukázka dvou aktivit aplikace ESET [41]

Aktivita představuje vstupní bod aplikace pro uživatele. Většina programovacích schémat má metodu main(), která představuje zmíněný entry point aplikace. V Android OS tímto způsobem fungují aktivity, které na základě stavu životního cyklu, ve kterém se aktivita právě nachází vyvolávají příslušná zpětná volání, tzv. callback. Jedna aktivita reprezentuje jednu obrazovku, jež obsahuje uživatelské rozhraní pro interakci s aplikací. Typicky má aplikace těchto aktivit několik. [42]

Pokud aplikace ke své funkci potřebuje zobrazit dvě rozdílné obrazovky, pak implementuje dvě rozdílné aktivity. Na obrázku 2.10 jsou zobrazeny dvě aktivity, vlevo se nachází tzv. SplashScreen aktivita, což je vlastně klasická úvodní obrazovka. Vpravo je poté zobrazena běžná aktivita, která slouží jako grafické uživatelské rozhraní a umožňuje uživateli interakci se samotnou aplikací.

Každá aktivita má svůj životní cyklus, stavy kterých aktivita může nabývat jsou: aktivní (active), pozastavená (paused), zastavená (stopped) a inactive (nekativní). Tyto stavy jsou řízeny systémem, nicméně vývojáři mohou chování aplikace v daných stavech upravovat přepsáním (@Override) výchozích manažerů událostí. Jediný callback, který vývojáři musí implementovat je "onCreate", který je volán při vytvoření dané aktivity. Každá aktivita pak musí být pomocí "<activity>" tagu deklarována v přiloženém manifestu, ten se v Android OS nachází v kořenu APK balíčku pod názvem "AndroidManifest.xml". Pokud by se aplikace pokusila spustit aktivitu, která není deklarována v manifestu, je vyvolána výjimka. [21]

Ukázka deklarace aktivity je demonstrována na obrázku 2.11. Na obrázku je jeden XML element, mající dva atributy. Bylo zmíněno, že aplikace pro Android OS nedisponují klasickou main() metodou, známou z většiny programovacích paradigmat, avšak jeden z atributů má hodnotu ".MainActivity_Thesis". Běžná konvence na Android OS je spíše: ".MainActivity" a značí hlavní, domovskou, aktivitu aplikace - jedná se však pouze o název aktivity, takže tato volba je čistě na vývojáři. Druhým atributem je poté použitý styl. Aktivita deklarovaná na obrázku neznačí vstupní bod aplikace, není tedy ekvivalentem k metodě main().

```
<activity
        android:name=".MainActivity_Thesis"
        android:theme="@style/MainTheme_Thesis">
</activity>
```

Obrázek 2.11 Deklarace aktivity [zdroj vlastní]

2.3.2 Service

Na rozdíl od aktivit jsou služby procesem běžícím na pozadí, mohou být spuštěny i zastaveny bez uživatelského rozhraní. Služba však ve výchozím stavu nevytváří pro svou činnost vlastní vlákno, ani oddělený proces. Naopak běží v hlavním vlákně hostitelského procesu, pokud to není vývojářem specifikováno jinak. Primární účel služeb je vykonávání dlouhodobě běžících úkolů na pozadí. Hlavní výhodou je pak jejich systémová priorita, která zaručuje delší životnost služby, než je tomu u aktivity. Služby mají sice nižší prioritu, než aktivní aplikace, zároveň však vyšší, než ty neaktivní. [25], [42] To v praxi znamená, že při případném nedostatku zdrojů jsou nejprve ukončeny neaktivní aplikace, až poté běžící služby na pozadí.

Tyto služby poskytované Android OS lze však využít i k záškodnickým aktivitám, dobrým příkladem je zneužití služby Accessibility Service, ta je používána zejména v bankovním malwaru. Rodiny bankovního malwaru prokazatelně zneužívající Accessibility Service jsou: TeaBot [43] a SharkBot [44]. Dle vědců z laboratoře Cleafy je SharkBot novou generací tzv. android banking trojanů. Jedná se o komplexní malware kombinující několik typů útoků, podstatnou část škodlivé činnosti však obstarává zmíněná služba Accessibility Service, díky které malware může například obcházet biometrická ověření využívané různými bankovními subjekty. Zjednodušeně interpretováno to znamená, že po udělení oprávnění k využití této služby je malware schopen sám sobě udělit jakákoliv další oprávnění a tím umožnit manipulaci s funkcemi a daty na zařízení. Nežádoucí vlastnosti přímo plynoucí ze zneužití služby [44]:

- plný vzdálený přístup k zařízení.
- Získání všech potřebných oprávnění.
- Zachycení a modifikace obsahu zobrazovaného na displeji.
- ATS (Automatic Transfer System), modul zodpovědný za schopnost automatického vyplnění (auto-fill) textových polí v legitimních bankovních aplikacích.
- Anti-delete (evasion technique), zajišťuje ochranu před smazáním aplikace pomocí nastavení zařízení.

V rámci operačního systému Android jsou rozlišovány dva typy služeb: bound a started. Started services jsou také označovány jako unbounded, tedy nevázané služby. Tyto komponenty jsou zodpovědné za start služeb, které pokračují v běhu na pozadí i pokud jejich původní inicializační komponenta zanikne. Nejsou tedy vázány na stálou existenci komponenty. [25]

Na druhou stranu služby vázané (bound services) zanikají právě se zánikem inicializační komponenty. [25]

Všechny služby pak musí být deklarovány, stejně jako aktivity, v souboru "AndroidManfifest.xml". Při deklaraci lze zabezpečit aplikaci, proti zneužití služby jinou aplikací a to nastavením hodnoty false atributu "android:exported", což zajistí, že ostatní aplikace nemohou spustit službu "ThesisService" na obrázku 2.12. [45]

```
<service
    android:name=".ThesisService"
    android:exported="false" />
```

Obrázek 2.12 Deklarace služby [zdroj vlastní]

2.3.3 Broadcast Receiver

Umožňuje reagovat na specifické události, které byly zaznamenány operačním systémem a zaregistrovat k nim systémovou, či aplikační činnost. Legitimní způsob užití je ku příkladu scénář s připojením nabíječky do zařízení, reakcí na tuto událost je zobrazení dialogového okna o aktivním nabíjení. [25] Toto vývojářům umožňuje rychle a relativně snadno obsluhovat různé události, což by jinak bylo velmi komplikované.

Broadcast receivery, stejně jako mnoho světových vynálezů a také funkcionalit operačního systému Android, měly přinášet hlavně užitek. Avšak i broadcast receiver je aktivně využíván k provádění škodlivé činnosti. Společnost Mitre na svých webových stránkách eviduje výzkumy a analýzy antivirových společností a analytiků mobilního malwaru, které zneužívají právě broadcast receiver. [46] Velmi populární je pro tvůrce mobilního malwaru receiver, který reaguje na událost "BOOT_COMPLETED". Tento receiver je zaregistrován k události dokončení bootování, tedy zapnutí zařízení. To tvůrci malwaru umožňuje vykonat škodlivou činnost hned po zapnutí zařízení.

V roce 2021 byl jedním z diskutovaných témat v cyber security komunitě špionážní software Pegasus, tento software byl v médiích označovaný jako military-grade, tedy vojenské úrovně. Nástroj vyvíjený izraelskou NSO Group v roce 2017 využíval "BOOT_COMPLETED" k aktivaci hned po spuštění, zároveň si tím zajistil perzistenci na zařízení. [47] Rok 2017 nebyl posledním výskytem zneužití receiverů, jak ukazuje web MITRE ATT&CK, kde jsou evidovány případy i z roku 2020 a 2021. [46]

Na rozdíl od aktivit a služeb je receiver možné deklarovat dvěma způsoby, jednak staticky v manifestu (obrázek 2.13), jak tomu bylo u předchozích komponent, jednak také dynamicky v rámci zdrojového kódu aplikace. [21]

```
<receiver android:name=".ReceiveBootCompleted"
    android:exported="false"
    android:enabled="true">
    <intent-filter>
        <action android:name="android.intent.action.BOOT_COMPLETED" />
    </intent-filter>
</receiver>
```

Obrázek 2.13 Statická deklarace Broadcast receiveru [zdroj vlastní]

2.3.4 Content Provider

Pro sdílení dat mezi aplikacemi implementuje Android OS komponentu Content provider. Pokud chce aplikace WhatsApp přistoupit k datům z aplikace Kontakty, lze využít Content provider. Používání poskytovatelů obsahu však není limitováno pouze na sdílení dat mezi aplikacemi, poskytovatele lze používat i pro data samotné aplikace a to ať už se jedná o strukturované databázové záznamy z SQLite databáze, nebo grafiku. Komponentu je možné implementovat vlastním způsobem, což může vést k lepšímu zabezpečení dat v rámci aplikace. [25]

Content providery používané jinou aplikací, musí být deklarovány v manifestu. V opačném případě o nich není systém vyrozuměn a není možné je spustit. [48]

```
<!--  Make app vulnerable  -->
<provider
    android:authorities="cz.thesis.RealBankingApp.SensitiveDataProvider"
    android:name="SensitiveDataProvider"
    android:exported="true">
</provider>
```

Obrázek 2.14 Deklarace zranitelného Content provideru [zdroj vlastní]

Na obrázku 2.14 je ukázkový provider s atributem exported nastaveným na hodnotu true, což umožňuje ostatním aplikacím přístup k tomuto poskytovateli obsahu s citlivými daty. [48]

Hlavní riziko ve spojení s poskytovateli obsahu na Android OS je únik dat, data z MITRE ATT&CK ukazují desítky případů zneužití Content provideru za účelem přístupu k citlivým informacím jako: kontakty [49], SMS zprávy [50] a logy hovorů [51].

2.3.5 Intent

K propojení jednotlivých komponent aplikace je využíván Intent, který je implementován stejnojmennou třídou. Intent typicky disponuje alespoň dvěma parametry. První je akce, jež má být vykonána (např. zobrazení, editace). Druhý parametr pak specifikuje samotná data, se kterými bude intent operovat. Intent má v aplikacích tři hlavní využití:

- spuštění aktivity.
- Vyrozumění registrovaných Broadcast receiverů.
- Spuštění vázané, či nevázané služby. [52]

Praktické využití intentu je, kromě zmíněného, například deklarace intent-filtrů v Manifestu aplikace. U aktivity bylo zmíněno, že samotná deklarace aktivity neznační vstupní bod aplikace. K deklaraci entry pointu je na Androidu využíván právě intent-filtr. Často aplikace využívají konvence ".MainActivity" pro hlavní prvek aplikace, avšak vstupním bodem aplikací většinou bývá tzv. SplashScreen aktivita - tedy úvodní obrazovka. Často je aktivita označována: ".SplashScreenActivity". Využití intent-filtru pro deklaraci launcher (spouštěcí) aktivity je demonstrováno na obrázku 2.15, zobrazené atributy elementu intent-filter se poté dají považovat za ekvivalent tradičních metod main(). Za zmínku na obrázku stojí opět atribut aktivity a sice atribut "android:exported", který v demonstrovaném stavu umožňuje přístup k aktivitě i jiným aplikacím.

```
<activity android:exported="true" android:name=".SplashScreenActivity_Thesis">
    <intent-filter>
        <action android:name="android.intent.action.MAIN"/>
        <category android:name="android.intent.category.LAUNCHER"/>
    </intent-filter>
</activity>
```

Obrázek 2.15 Deklarace aktivity s intent-filtrem [zdroj vlastní]

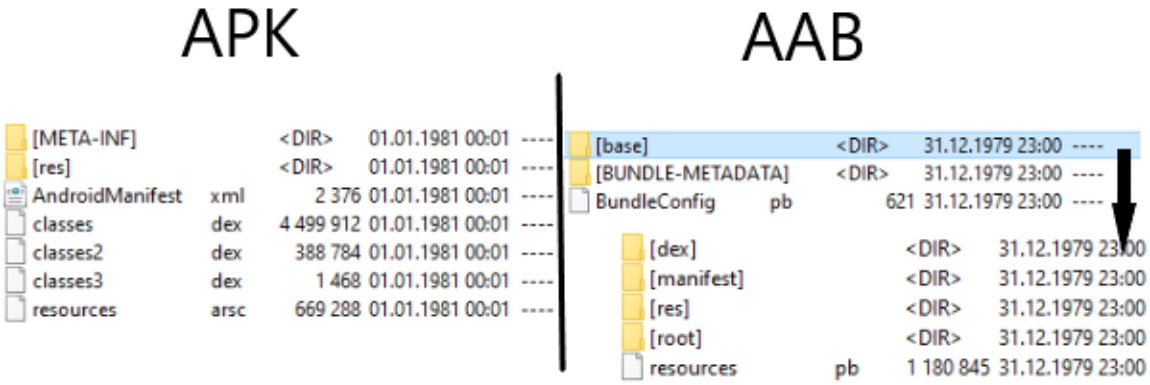
2.3.6 Fragment

Hlavní účel fragmentů je zapouzdření části grafického rozhraní aktivity pro znovupoužitelnost zejména na různých displejích. Fragmenty jsou přímo závislé na aktivitách a nemohou být vytvořeny mimo ně. Lze je však implementovat i jako perzistentní, takže mohou existovat i po zániku inicializační aktivity. Fragment není deklarován v manifestu, jelikož se jedná o součást aktivity. [21]

2.4 Balíčky aplikací - AAB a APK

Operační systém Android v roce 2022 využívá dva balíčky pro své aplikace. Android Application Bundle (.aab) je publikační formát a slouží pro publikaci aplikací na oficiální distribuční platformě. Dříve sloužil jako publikační formát Android Application Package (.apk), APK však je nadále využíván jako distribuční formát. AAB poskytuje všechny potřebné zdroje pro aplikace a deleguje tak generování APK i jejich podpis na Google Play Store. [21]

Formát AAB slouží zejména vývojářům, ke zjednodušení a optimalizaci celého procesu distribuce aplikace. Není tak nutné sestavovat a podepisovat jednotlivé APK balíčky. Zároveň je uživateli dodáno APK, které je menší a lépe optimalizované. Tento formát je vyžadován pro všechny aplikace od srpna roku 2021. [53] Podobnost mezi formáty je patrná z obrázku 2.16, kde jsou sestaveny (build) oba formáty pro totožnou aplikaci. Za povšimnutí stojí i datum modifikace, ačkoliv byla aplikace sestavena v roce 2022, pod operačním systémem s aktuálním nastavením a aktualizovanými nástroji. Tato data se pak ukazují i ve výzkumech, popsaných níže, zaměřených na datové sady a tyto balíčky bývají vyloučeny ze statistických údajů o první a poslední modifikaci.



Obrázek 2.16 Sestavené balíčky: AAB, APK [zdroj vlastní]

Doručený APK balíček je v podstatě ZIP archiv a může být rozbalen jakýmkoliv dekomprimačním nástrojem, podporujícím formát ZIP. Ne všechny soubory jsou rozbalením tímto způsobem lidsky čitelné. Při analýze APK souborů je místo klasické extrakce doporučeno využít nástroj pro reverzní inženýrství Apktool. [26] Nástroj Apktool v rámci dekompilace vykoná i dekódování nečitelných souborů. [54]
Android Application Package (APK) je:

- generován z Android Application Bundle .aab.
- Distribuční a instalační formát.
- Zkompilovaný zdrojový kód a všechny potřebné zdroje aplikace.

- Hlavní zkoumaný subjekt při analýze.

2.4.1 Struktura distribuované aplikace

Obvyklá struktura APK balíčku dle Alexandra Kleymenova [26], pokud není uvedeno jinak:

- ***resources.arsc:*** Soubor obsahující předkompilované zdroje a prostředky aplikace, jako XML pro tvorbu GUI [21]
- ***res:*** Adresář obsahující různé zdrojové soubory - XML, obrázky a další prostředky, které nebyly zkompilovány do arsc souboru [21]
- ***META-INF:*** Ukládá metadata, týkající se balíčku
  - *MANIFEST.MF:* Soubor manifest obsahuje názvy a SHA1/SHA2 otisky souborů v APK
  - *<název>.RSA:* Soubor RSA obsahuje podpis a certifikát aplikace.
  - *<název>.SF:* Soubor SF obsahuje SHA1/SHA2 otisk odpovídajících řádků v *MANIFEST.MF:*
- ***AndroidManifest.xml:*** Kardinální manifest soubor, který je pro Android OS povinný. Popisuje strukturu aplikace a její komponenty. [25]
- ***classes<n>.dex:*** Zkompilovaný soubor obsahující DEX byte kód aplikace, pokud je těchto souborů v balíčku více, jsou pojmenovány dle obrázku 2.16, v části popisující APK

Kromě uvedeného se mohou v APK balíčku vyskytovat i jiné soubory a adresáře, např. adresář assets, který již byl zmíněn výše, v souvislosti s dynamickým načítáním souborů dex. [26]

2.4.2 AndroidManifest.xml

Již bylo zmíněno, že AndroidManifest.xml je povinný soubor každé aplikace, kromě informací o obsažených komponentách však obsahuje i další důležité informace, jako cílovou verzi OS, nebo název balíčku. Zařízení tento soubor využívají při instalaci, ale i při spouštění aplikací. [25]

AndroidManifest.xml podle B. Muellera a dalších [21] obsahuje:

- Komponenty: Aktivity, Služby, Content providery, a Intent receivery
- Deklarovaná oprávnění aplikace

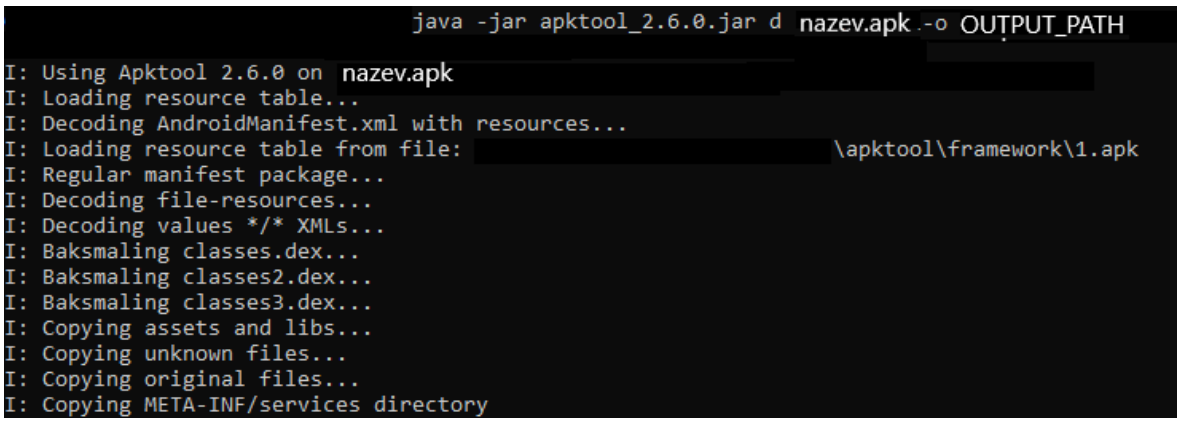
- Metadata aplikace: ikonu, používaný styl (theme), označení verze
- Další: kompatibilní verze API, statické Broadcast receivery

2.5 Dekomprimace a dekompilace

Rozbalení APK standardním způsobem - dekomprimací, jako je tomu u ZIP formátu, neposkytuje analytikům uspokojivé výsledky. [26] Kromě již zkompilovaných souborů, které je nutné dekompilovat pro další analýzu však balíček obsahuje i jiné soubory a adresáře, jejichž dekompilace není nutná. Dekomprimace balíčku by se tak mohla na první pohled jevit, jako korektní krok při jejich analýze. V případě, že bezpečnostní analytik musí zjistit pouhé aktivity aplikace, stačí mu informace uvedené v souboru AndroidManifest.xml. Tento soubor se semi-strukturovanými daty, ale patří k lidsky nečitelným souborům po dekomprimaci. Pro demonstraci byla vytvořena a sestavena aplikace do balíčku APK. Na obrázku 2.17 je otevřený soubor AndroidManifest.xml, po provedení standardní dekomprimace.

Obrázek 2.17 Nečitelný AndrodManifest.xml [zdroj vlastní]

Řešení poskytuje nástroj pro reverzní inženýrství Apktool. Nástroj má dvě základní funkcionality - první označená “d - decode” (obrázek 2.18), poskytuje řešení na uvedený problém a druhá “b - build” představuje inverzní proces - sestavení APK balíčku z dekompilovaných zdrojů.



Obrázek 2.18 Činnost nástroje Apktool [zdroj vlastní]

V rámci celé činnosti "d - decode", nástroj provádí dekompilaci a dekódování. Dekompilovány (Baksmaling, protože je využíván nástroj Baksmali) jsou zkompilované soubory, typicky: classes.dex a resources.arsc. Dekódování je pak využíváno pro lidsky nečitelné - zakódované (encoded) zdroje. Poslední prováděnou operací je kopírování a to zejména zdrojů, které byly v čitelném formátu, nebo nevyžadují žádné další úpravy.

Dekompilace tedy umožňuje obnovit zkompilované zdroje aplikace, do lidsky přívětivějšího formátu .smali, který je využíván při statické analýze. Spolu s dekódováním tak poskytuje zdroje aplikace, včetně předtím nečitelných zdrojů, jako je například AndroidManifest.xml. [26], [54]

Na obrázku 2.19 je AndroidManifest.xml zpracovaný nástrojem Apktool, do lidsky čitelného formátu.

F: > Android > DP > packages > apktooled > AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
android:compileSdkVersion="32"
android:compileSdkVersionCodename="12"
package="cz.dordy.malware.tool.diploma"
platformBuildVersionCode="32"
platformBuildVersionName="12">

    <application android:allowBackup="true" android:appComponentFactory="androidx.core.app.CoreComponentFactory"
    android:debuggable="true" android:extractNativeLibs="false"
    android:icon="@mipmap/ic_launcher" android:label="@string/app_name" android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true" android:theme="@style/Theme.Diploma">

        <activity android:exported="true" android:name="cz.dordy.malware.tool.diploma.MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN"/>
                <category android:name="android.intent.category.LAUNCHER"/>
            </intent-filter>
        </activity>
    </application>
</manifest>
```

Obrázek 2.19 Čitelný AndrodManifest.xml [zdroj vlastní]

Spousta výzkumníků ale preferuje pro zrychlení a zjednodušení analýzy dekompilaci přímo do zdrojového kódu Javy. To analytikům umožňuje číst téměř původní zdrojový kód v jazyce Java. [26]

3 DATOVÉ SADY MOBILNÍHO MALWARU PRO ANDROID OS

Klasická datová sada používaná umělou inteligencí má typicky charakter strukturovaných dat, která byla vytěžena z určitých subjektů (nebo jiných semi-strukturovaných a nestrukturovaných dat), příkladem může být datová sada od CIC: CIC-AAGM2017 [55]. Tato datová sada, kromě ".pcap" souborů popisujících záznam internetového provozu z činnosti aplikací, obsahuje ".csv" soubory. Tyto jsou generovány právě z pcap souborů a jsou již použitelné pro další výzkumnou činnost. Původ zachycené internetové komunikace je však opět pevně svázán s APK balíčky, které byly instalovány na zařízení a ze kterých byly generovány právě zmíněné pcap soubory. Tato dynamicky sbíraná data jsou, na rozdíl od samotných APK balíčků použitelná pro výzkum.

Datová sada mobilního malwaru pro Android OS je však v komunitě zabývající se kybernetickou bezpečností častěji chápána, jako kolekce APK balíčků. Tato datová sada může a nemusí být popsána jednotlivými popisky - labely. Důkazem mohou být různé datové sady, či služby poskytující přímo vzorky APK balíčků - AndroZoo [20], VirusShare [56], PRAGuard [57], či AndroidMalware 2018 [58].

Tento přístup je logický, jelikož umožňuje validovat a analyzovat APK soubory individuálním postupem. Lze tak například ověřit, zda je APK dekompilovatelné - což by pouze z dynamicky sbíraných atributů bylo velmi složité, pravděpodobně i neproveditelné. Umožňuje to také novátorské přístupy k získávání dat z aplikací a jejich klasifikaci. Stejně tak to umožňuje vlastní implementaci již tradičních přístupů, jako je třeba statická analýza.

Přímá práce s APK balíčky s sebou však nese i jistá rizika. Není totiž možné využít, již existující strukturovaná data například v csv formátu, naopak je nutné data nejprve získat z jednotlivých APK balíčků další analýzou.

Je tak nutné zajistit dostatečné místo pro všechny vzorky. Následně je možné vzorky nasbírat a provést vlastní analýzu, ze které jsou získána data. Tento postup byl zvolen v práci [17], jejíž výstupem je: datová sada tvořená APK balíčky; .pcap a .csv soubory. Tato data byla samozřejmě získána z korespondujících balíčků. Tato datová sada nese název CICMalDroid 2020 a svým zpracováním je výstup vlastně hybridní, kdy kombinuje obě výše uvedené varianty - tedy poskytnutí kolekce APK balíčků i datové sady analyticky získaných dat.

V rámci analýzy dostupných datových sad vzniklo rozřazení, podle přístupnosti datových sad.

1. Veřejně dostupné datové sady:
    - pro stažení, či práci není nutná autentizace - je volně přístupná všem

- AndroidMalware 2018 [58].

2. Akademické datové sady:
   - pro použití je nutné autentizace výzkumníka, případně pozvání
   - VirusShare [56].
3. Profesionální datové sady:
   - veřejně nedostupné, přístup pouze osoby s příslušným oprávněním
   - AV společnosti, výzkumné týmy, laboratoře.

3.1 Veřejně dostupné datové sady

Charakteristiky popisující datové sady jsou tvořeny na základě analýzy daných datových sad, jejich zdrojů (publikace, články, webové stránky) a vlastních zkušeností s datasety příslušných typů.

Veřejně dostupné datové sady mívají obvyklou velikost v řádech desítek až tisíců vzorků. Tyto datové sady jsou volně přístupné bez jakékoliv autentizace. Nejčastějším zdrojem využívaným k distribuci těchto kolekcí je GitHub [59]. Počet vzorků z těchto datových sad činí ideální množinu pro testování různých algoritmů klasifikace/detekce/sběru dat a dalších disciplín, což je dáno právě relativně malým počtem vzorků, takže případné úpravy vah, modelů nebo systému neznamenají takové časové náklady, jako při analýze milionu aplikací.

Veřejně dostupné datové sady však skýtají i různá úskalí, která nejsou úplně běžná u jiných typů sad. Fakt, že datová sada je veřejná, neznamená, že není kvalitní. Na druhou stranu u veřejných úložišť, kde se často mimo samotných aplikací vyskytují různé obrázky a další nevalidní soubory, není předpokládána taková výzkumná váha, jako je tomu například u profesionálních sad.

U veřejně dostupných datových sad se vyskytují následující problémy: nejednotné názvosloví, různé formáty (apk, zip, zašifrovaný zip s heslem), duplicity, různé úrovně zanoření, nevalidní vzorky, poškozené vzorky, absence APK balíčků, výskyt malwaru na jiné platformy, problémy s dekompilací. Předpoklad těchto problémů je spojený zejména s datasety 1-6 v tabulce 3.1. Specifikum pro tyto datové sady je absence benigních vzorků, ale také nutnost dalšího zpracování umožňujícího následnou výzkumnou činnost.

Veřejné datové sady			
ID	Název sady	Vydavatel	Očekávaný # záznamů
1	AndroidMalware_2018 [58]	sk3ptre	214
2	AndroidMalware_2019 [60]	sk3ptre	131
3	AndroidMalware_2020 [61]	sk3ptre	222
4	AndroidMalware_2021 [62]	sk3ptre	48
5	Android Malware Sample Library [63]	mstfknn	191
6	Contagio mini dump [64]	[64]	0-395 [6]
7	MalwareBazaar [65]	abuse.ch	1 458
8	Labeling Dataset[6]	[6]	2 470 000

Tabulka 3.1 Seznam veřejně dostupných datových sad mobilního malwaru

1. - 5. Očekávaný počet vzorků je odvozen z počtů unikátních hash otisků. Datové sady obsahují populární vzorky malwaru posledních 5 let, včetně těch spojených s onemocněním COVID-19. Pro sady platí, že nabývají různých struktur a úrovní zanoření s nejednotným názvoslovím, kdy některé vzorky jsou pojmenovány dle rodin, jiné dle otisku a některé i dle měsíce zveřejnění. Sady by měly reprezentovat moderní mobilní malware, pokud je jejich časové řazení dle výskytu správné. Časové zařazení bylo ověřeno v praktické části spolu se získáním charakteristik datových sad.

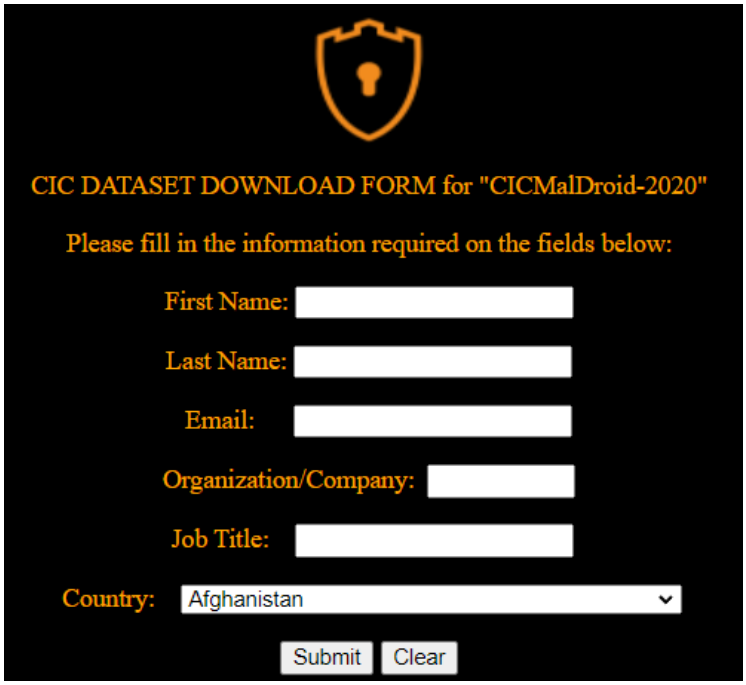
6. Contagio mini dump Velmi populární dataset, který vznikl v roce 2011, poslední vzorky byly přidány v roce 2018. Tento dataset byl hodně využíván napříč výzkumy. [6] Velikost datasetu byla odvozena z práce [6], jelikož se nejvíce blíží počtu vzorků dostupných na úložišti Contagio [64]. Uváděný počet se pak různí i v odborných pracích, kde například autoři [66] uvádí pro Contagio 189 vzorků. Tento dataset obsahuje převážně vzorky, které již dostatečně nereprezentují současný mobilní malware, jedná se o vzorky pocházející z let 2011-2018. [6] Na druhou stranu datová sada nabízí velkou diverzitu vzorků z průběhu několika let. Mezi vzorky se nachází i soubory typu HTML, RAR a jiné. Navíc zde není jednotný formát názvosloví, vzorky jsou ukládány tak, jak byly analytiky pojmenovány před přidáním do sady.

7. MalwareBazaar Tato webová služba umožňuje sdílení vzorků malwaru. Pro využití služby není nutná registrace, nicméně je možná. Není zde předpoklad nevalidních vzorků, ani nejednotného názvosloví, jelikož služba podporuje několik formátů hash otisků pro identifikaci vzorků - včetně SHA256. Jistý problém zde představuje formát platformy, kdy se analytici setkávají s malwarem na různé operační systémy, je proto nutná filtrace vzorků. Přesný počet vzorků na jednotlivé platformy se v rámci služby nepodařilo dohledat. Služba však poskytuje csv

soubor obsahující jednotlivé vzorky na různé platformy a jejich základní charakteristiky. Počet vzorků byl zjištěn vlastní analýzou nad csv souborem poskytnutým službou MalwareBazaar ke dni 27.4.2022. Analýza byla inspirována zdroji [67] a [68], jednalo se o filtraci APK souborů mezi ostatními formáty s využitím knihoven usnadňujících práci s daty.

8. Labeling dataset Výstupem výzkumu [6] je rozsáhlá a unikátní datová sada vztažená k labelingu malwaru, tato obsahuje přes 2 miliony otisků aplikací a k nim příslušné názvy rodin, získané různými AV řešeními - jednotlivé labely. Jedná se o zajímavý i užitečný dataset, hlavní problém zde ale představuje absence APK balíčků. Datová sada je, na rozdíl od výše zmíněných, ve formátu csv. Ukázala se velká nekonzistence v labelech - popiscích jednotlivých vzorků a to jak napříč akademickými řešeními, tak i mezi anti-malwarovými službami. Práce shrnuje poslední dekádu na poli výzkumů zaměřených na Android OS a znamená velký přínos, jelikož se jako jedna z mála zabývá i původem a datem vzniku datových sad/vzorků. Výzkumný dopad samotné datové sady však není tak velký ve smyslu analýz jednotlivých aplikací a jejich dalšího posuzování. Na druhou stranu ale nabízí velmi ojedinělý pohled na jednotlivá AV řešení, kdy je možné zkoumat úspěšnosti detekcí a další faktory, jako třeba překrývání názvů malwarů mezi danými anti-malwarovými službami.

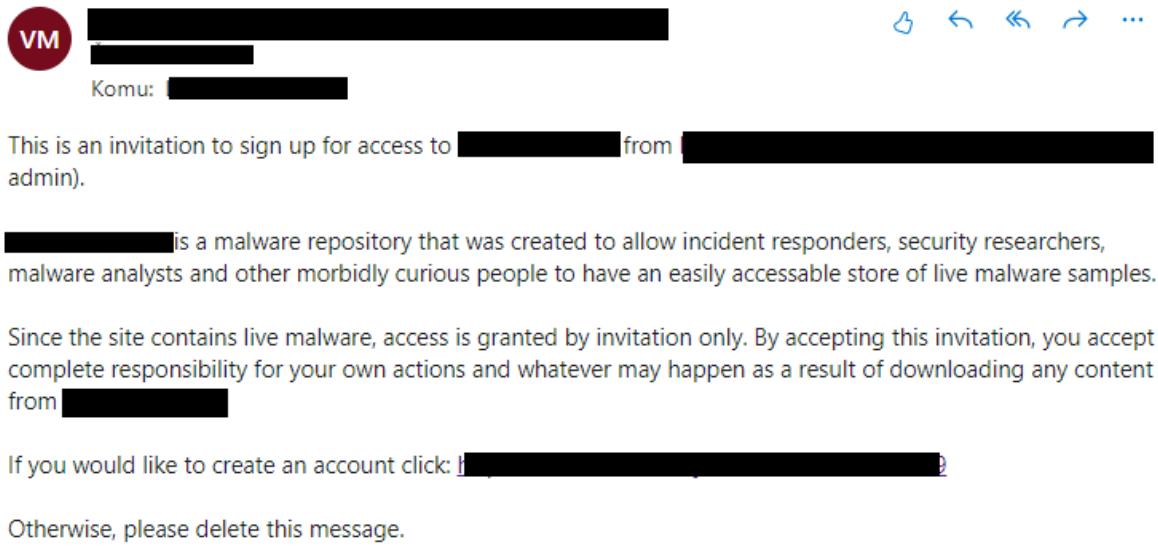
3.2 Akademické datové sady



Obrázek 3.1 Formulář CIC [zdroj vlastní]

Tento typ datové sady předpokládá využití zejména v dalších akademických pracích, či výzkumných týmech. Tyto kolekce mají nepopiratelný vliv na vývoj výzkumů na poli mobilní bezpečnosti. To je dáno zejména tím, že mají vysokou kvalitu, bývají dílem několikačlenných týmů a zároveň jsou přístupné dalším vědeckým subjektům. Velmi často jsou tak tyto sady znovupoužívány, přičemž u aktivně spravovaných služeb, jako AndroZoo [20] to při korektním pracovním postupu nepředstavuje problém. U statických datových sad, které se v čase nemění je situace jiná - vzorky nemusí být dostatečně reprezentativní (aktuální), což má zásadní vliv na konečný výzkum.

Akademické datové sady vyžadují určitý způsob autentizace. Toto probíhá buď vyplněním formuláře (viz obrázek 3.1) - CIC datasety, či zasláním e-mailu, ve kterém jsou uvedeny potřebné informace pro vydavatele, jako například název fakulty, ústavu, výzkumný tým. Reakcí na žádost pak typicky může být e-mail - pozvánka k vytvoření účtu, s vygenerovanou adresou pro vytvoření přístupu ke službě (obrázek 3.2), případně vůbec žádná reakce.



Obrázek 3.2 Pozvánka k vytvoření přístupu [zdroj vlastní]

Často je preferováno, aby o tyto vzorky žádali vedoucí příslušných týmů. Po zaslání žádosti na danou adresu, je již pouze na poskytovateli datové sady, zda ji výzkumnému subjektu poskytne, či nikoliv. Podmínky jednotlivých vydavatelů se různí a jsou přesně definovány v rámci podmínek užití jednotlivých datových sad. Jejich citování při použití je samozřejmostí.

Akademické datové sady jsou často součástí výzkumů a obsahují obvykle stovky, až miliony vzorků mobilního malwaru. Není zde předpoklad nastrčeného malwaru na jiné platformy. Datové sady mají často jiné formáty a to jak v rámci jedné sady - CICMalDroid 2020 [17], tak napříč sadami. Různí se také práce s nimi, webové služby většinou využívají API přístupu - AndroZoo [20]. Zatímco datové sady, které se v čase

nemění jsou k dispozici ke stažení jako celek - CICMalDroid 2020 [17]. Zpracování velkých datových sad může způsobit problémy, jedná se o časově i výpočetně náročný proces. [69]

Vzhledem k velikosti akademických datových sad je spíše nutné ověřit pravý záměr aplikací, jak ukázal výzkum [6], kdy v datové sadě AndroZoo zpětnou kontrolou malignity objevili, že více než 300 tisíc vzorků bylo falešně pozitivních. Druhým důležitým faktorem těchto datových sad je dekompilovatelnost jednotlivých balíčků, která je nutná pro získání zdrojového kódu a zároveň u dynamicky tvořených datových sad nemusela být verifikována. V tabulce 3.2 jsou významné datové sady mobilního malwaru s akademickým přístupem. Speciální jsou v tomto ohledu datové sady vydávané CIC [70] - zde většinou platí, že datové sady jsou sice veřejně dostupné, ale pro výzkumníky, s podmínkou citace a vyplnění zmíněného formuláře. Právě formulář představuje komplikaci při zařazení do kategorie dle dostupnosti. Formulář pravděpodobně slouží pouze pro sběr dat, to znamená, že výzkumník není přímo autentizován. Tyto datové sady by tak, do jisté míry, mohly být označeny i jako veřejné, pro výzkumné účely. Vznikem na akademické půdě, svou kvalitou a nutností uvedení informací se však řadí mezi sady akademické.

Akademické datové sady			
ID	Název sady	Vydavatel	Očekávaný # záznamů
1	AndroZoo	[20]	2 240 000 [6]
2	VirusShare	[56]	11 000 - 35 000 [6]
3	ApkLab.io (Avast software)	[71]	24 477 264
4	Android Malware Genome Project	[7]	1 260
5	PRAGuard	[57]	10 479
6	AMD	[8]	24 650
7	Android Botnet dataset	CIC [19]	1 929
8	CIC-AAGM2017	CIC [55]	400
9	CIC-AndMal2017	CIC [72]	4 354
10	CICInvesAndMal2019	CIC [73]	426
11	CICMalDroid 2020	CIC [17], [74]	*17 341
12	CCCS-CIC-AndMal-2020	CIC [75], [76]	200 000

Tabulka 3.2 Seznam akademických datových sad mobilního malwaru

1. **AndroZoo:** Internetová služba je aktivně udržovaným projektem, jedná se o jednu z nejobsáhlejších služeb. Ke dni 28. 4. 2022 eviduje celkem 19 025 897 aplikací. Již bylo zmíněno, že akademické sady mobilního malwaru často obsahují i legitimní aplikace, tak je tomu i u této kolekce. S datovou sadou pracuje například [6], pomocí API byly vyfiltrovány aplikace označené jako škodlivé, celkem jich bylo přibližně 2,5 milionu. Po ověření službou VirusTotal však bylo zjištěno, že celkový počet vzorků mobilního malwaru na službě AndroZoo je okolo 2,24 milionu

vzorků. Výhodou kolekce je zejména její aktuálnost, což umožňuje výzkumníkům sbírat čerstvé vzorky.

2. **VirusShare:** Prvotní ohledání služby VirusShare může připomínat předešlou platformu AndroZoo. Tyto sady mají dva zásadní rozdíly. Jednak VirusShare obsahuje i malware pro jiné platformy, než Android OS a jednak také neobsahuje benigní vzorky. Absence legitimních aplikací je mezi akademickými sadami neobvyklá. Uvedené skutečnosti naznačují, že platforma je zaměřena na malware obecně, nejedná se o sadu specializovanou na mobilní malware a lze tak předpokládat nižší kvantita vzorků, než je tomu například u AndroZoo.

3. **Apklab.io:** Platforma vytvořená antivirovou společností Avast. Jedná se pravděpodobně o nejobsáhlejší platformu zaměřenou na maligní APK soubory. Platforma má velmi úzkou komunitu - používá ji více, než 510 výzkumníků. Poskytuje statickou i dynamickou analýzu provedenou nástroji společnosti Avast, což umožňuje nahlédnout na analýzy profesionálních AV řešení. [71]

4. **Malgenome:** Pravděpodobně vůbec první datovou sadou svého typu. Labeling byl tvořen manuálně, což umožnil zejména nižší celkový počet aplikací. Datová sada obsahuje vzorky z let 2010 a 2011, což znamená, že již dostatečně nereprezentují moderní mobilní malware. Historicky patří k sadám, které byly opakovaně využívány a to i v roce 2018 a později. [6] Datová sada od 21. 12. 2015 již není sdílena. [10]

5. **PRAGuard:** Datová sada vznikla v roce 2015, autoři se rozhodli sadu sdílet zejména kvůli tomu, že sběr malwaru pro výzkumné účely je poměrně složitý úkol. Původní vzorky pochází z Contagio mini dump a Malgenome datasetů. Tyto vzorky byly obfuskovány sedmi různými technikami, výsledkem je datová sada PRAGuard, která již však od dubna roku 2021 není dostupná. [57], [77]

6. **AMD:** Android Malware Dataset vzniknul v roce 2017, jeho podpora je pravděpodobně ukončena, jelikož webová stránka s datovou sadou je nedostupná. [9] Kolekce disponovala vzorky mobilního malwaru pocházejícími z let 2010-2016. Řadí se také mezi sady, které byly znovupoužívány. [6]

7. **Android Botnet dataset:** Dataset specializovaný na botnety zneužívající Android OS který vznikl v roce 2015 a obsahuje 1 929 vzorků malwaru. Původ vzorků je z Malgenome, Contagio mini dump, VT a jednoho anti-malware řešení. Vzorky pochází z období 2010-2014. Dataset tak obsahuje nereprezentativní vzorky, navíc velmi specificky zaměřené. [19]

8. **CIC-AAGM2017:** Datová sada obsahuje celkem 1 900 vzorků, 1 500 je však benigních. Malware je rozdělen do dvou kategorií - Adware (250 vzorků) a Obecný malware (150 vzorků). Dataset byl analyzován dynamicky na reálném zařízení. Zaměření analytické části datové sady je sběr zachycené internetové komunikace. Reálné zařízení bylo zvoleno kvůli tzv. anti-analysis technikám, což jsou techniky, kterými se tvůrci malwaru mohou vyhnout případné detekci. Typicky upravením svého chování při zjištění, že aplikace byla spuštěna v emulátoru. [55] Sada není tak specificky zaměřená, jako třeba Android Botnet dataset. Problém datové sady je, že 250 škodlivých vzorků je adware, který je v rámci detekcí problematický svým charakterem. Je tedy možné tvrdit, že datová sada obsahuje pouze 150 vzorků "plnohodnotného" malwaru.

9. **CIC-AndMal2017:** Dataset vznikl v roce 2017, obsahuje vzorky z Contagio mini dump, VT a dřívějších výzkumů. Celkem sada obsahuje 10 854 vzorků, z nichž 4 354 je škodlivých a 6 500 je legitimních. Z 6 500 aplikací pocházejících z Google Play Store bylo 1 435 odstraněno, protože byly pomocí nástroje VirusTotal označeny, dvěma a více AV řešeními jako podezřelé, nebo Adware. Na dataset byl aplikován labeling s využitím VirusTotal. Zmíněný problém s nesouladem v pojmenování rodin malwaru byl vyřešen vybráním názvu, jenž se napříč AV řešeními vyskytoval nejčastěji. Analyzováno pak bylo přes 5 tisíc aplikací, které se podařily nainstalovat, jedná se o 426 vzorků malwaru a 5,065 legitimních aplikací. Analýza probíhala ve třech definovaných stavech: při instalaci, před a po restartu zařízení. Datová sada obsahuje jak vzorky APK balíčků, tak sesbíraná data. [72] Množství poskytnutých balíčků mobilního malwaru je relativně vysoké, avšak dynamický sběr dat ovlivnil analytickou část práce, kdy se povedlo analyzovat spíše legitmní, než škodlivé aplikace. Pokud by aplikace nebyly analyzovány v emulátoru, pravděpodobně by se podařilo analyzovat větší množství mobilního malwaru.

10. **CICInvesAndMal2019:** Jedná se o druhou část CIC-AndMal2017, na rozdíl od první zmíněné sady však navíc obsahuje statické atributy: oprávnění a intent. Dynamická analýza poté zahrnuje například API volání a vygenerované logy ve výše zmíněných tří fázích: při instalaci, před a po restartu. Počty zpracovaných vzorků jsou stejné jako u předchozí datové sady: 426 vzorků malwaru a 5 065 legitmních aplikací. [73]

11. **CICMalDroid 2020:** Celkový počet vzorků v této sadě je 17 341. Aplikace pochází z několika zdrojů, jedná se například o VirusTotal, Contagio, AMD a další včetně již vytvořených datových sad ve výzkumech CIC. Datová sada je rozdělena do následujících kategorií: adware, bankovní malware, SMS malware, riskware a

benigní. Kromě vzorků v podobě aplikací, obsahuje datová sada i analytickou část. Data byla sbírána dynamicky, s využitím emulátoru. Tímto způsobem se podařilo vytěžit 11 598 aplikací z nichž 9 803 je škodlivých. Mezi sbíraná data patří staticky a dynamicky extrahované informace včetně kompletně zachyceného síťového provozu v průběhu analýzy. [17]

12. **CCCS-CIC-AndMal-2020:** Datová sada čítající 400 tisíc vzorků, sada je vyvážená, takže obsahuje přibližně 200 tisíc vzorku legitimních i škodlivých vzorků. Legitmních 200 tisíc aplikací pochází z AndroZoo. Škodlivé aplikace pak byly sbírány ve spolupráci s CCCS (Canadian Centre for Cyber Security) [78]. Záměrem bylo vytvoření takové kolekce, která bude mít vzorky s podobnými charakteristikami, podařilo se vytvořit 14 kategorií škodlivých aplikací s tímto specifikem. Nevýhodou této datové sady je pouze absence APK balíčků, je tedy možné pracovat s analytickými daty poskytnutými v rámci datové sady, nikoliv však s aplikacemi. Data extrahovaná z aplikací jsou jak statická (název balíčku, komponenty aplikací, oprávnění a základní informace o aplikaci - počty souborů a další), tak dynamická (API, síťový provoz, baterie a další). [75]

3.3 Profesionální datové sady

Datové sady ke kterým není volný přístup, často s nimi smí pracovat pouze zaměstnanci společnosti, která datové sady vlastní - typicky se jedná o antivirové společnosti či organizace zabývající se kybernetickou bezpečností. Pravděpodobně nejblíže k profesionální datové sadě bude mít kolekce Apklab.io [71], kde společnost Avast umožňuje nahlédnout na data zpracovaná jejich specializovanými nástroji pro statickou a dynamickou analýzu.

Co se týče velikosti profesionálních datových sad, tak například společnost Nokia ve své zprávě o hrozbách z roku 2021 uvedla, že disponuje téměř 33,4 miliony unikátních vzorků na platformu Android OS. [79]

3.4 Tvorba datových sad

Rashed a další, v publikaci [6] analyzuje dostupné datové sady mobilního malwaru. Dle autorů probíhá typicky tvorba velkých datasetů ve dvou krocích:

- Skenování vzorků/otisků pomocí VirusTotal (VT)
- Labeling, tedy označení vzorků dle rodin malwaru

Druhá část představuje problém - služba VirusTotal skenuje vstup za užití několika desítek předních antivirových (AV) řešení. Jak správně Rashed upozorňuje, interní

názvy rodin nejsou shodné napříč AV službami. Proto není bezproblémové použití VirusTotal při labelingu, nicméně pro ověření záměru aplikace je tato služba ideální. [6]

V praktické části se ukázalo, že vzorky obsažené v datové sadě nemusí naplňovat původní představu o aplikaci. V praxi se tak může stát, že datová sada obsahující mobilní malware obsahuje i benigní vzorky. Toto potvrzuje i výzkum [6]. Autoři ověřovali škodlivost u více, než 2,5 milionu vzorků a ukázalo se, že přes 300 tisíc z nich, již nebylo ohodoceno žádným AV řešením jako maligní. Tyto vzorky pocházely ze služby AndroZoo [20].

Ačkoliv autoři [6] uvádí že data získali z již existujících datových sad, sběr není přímo zahrnut do tvorby datové sady. Autoři [81] však zdůrazňují, že sběr / tvorba / kurátorství výzkumných datových sad, představuje významnou složku celého výzkumu. Tvorba datové sady může trvat i několik let, zároveň se však může stát, že výzkum pracující s datovou sadou může mít lepší hodnocení, než samotný výzkum, který dataset publikoval. Zároveň je přínos kvalitních datových sad nepopíratelný. [81]

3.5 Sběr malwaru

Z výše uvedených datových sad vyplývá, že sběr malwaru probíhá většinou s využitím otevřených zdrojů, případně zdrojů akademických a nebo je založen na komunitním sdílení vzorků. Kvůli tomu jsou datové sady a jednotlivé vzorky v akademické sféře často opakovaně využívány. Jinak je tomu u datových sad profesionálních, které využívají vzorky vlastní a unikátní. Na základě analýzy datových sad uvedených výše, jsou předpokládány následující metody sběru mobilního malwaru:

- Sběr v infrastruktuře - analýza síťového provozu, tvorba honeypotů a další
- Sběr na zařízení - AV společnosti pravděpodobně monitorují aplikace, které ještě v databázi nemají
- Open Source Intelligence - využití otevřených zdrojů, může vést k veřejně a akademicky dostupným datovým sadám, ale i k jednotlivým vzorkům
- Dark Web - malware lze najít na dark webu, i ke koupi [80]

3.6 Analýza APK

Na platformě Android OS jsou rozlišovány: statická, dynamická a hybridní analýza

1. Dynamická analýza - zajímá se o chování aplikace/systému, vykonává kód, typicky vyžaduje emulátor, nebo reálné zařízení [26]

2. Statická analýza - je zaměřena na zdrojový kód aplikace, nevykonává kód [26]

3. Hybridní analýza - například výzkum [75], sbírá statické i dynamické informace, po spuštění v emulátoru

V praktické části je využita analýza statická a nástroje VirusTotal [5] a Androguard [82] s rozhraním implementovaným v jazyce Python [83].

II. PRAKTICKÁ ČÁST

4 NÁVRH SYSTÉMU

V rámci diplomové práce byl navržen automatizovaný systém pro analýzu datových sad mobilního malwaru, který běží v infrastruktuře laboratoře PT LAB [84]. Systém je interně nazýván jako AndroBank a přesněji se jedná o kooperativní multiagentní systém s cílem, vykazující známky racionálního chování. To v praxi znamená, že systém disponuje několika agenty s centrálním řízením. Centrální řízení je zodpovědné za jejich rutinní chod a spouštění, ale jednotliví agenti pracují dle své režie. Jednotliví agenti jsou schopni plnit požadavky (analyzovat datové sady, informovat výzkumníka a další) a komunikovat s okolními systémy. Agenti spolu kooperují, kdy například agent pro statickou analýzu čeká, než detekční agent vyhodnotí APK balíček. Agenti mají různé dílčí cíle, ale jeden společný cíl - tímto cílem je plné zpracování všech APK balíčků zavedených do systému. Tito agenti spolu komunikují ve sdíleném prostředí, což umožňuje koordinaci jejich akcí. Známky racionálního chování pak systém vykazuje například v situacích, kdy se chová jiným způsobem k aplikacím, které byly někdy v historii skenovány a jinak k těm, které jsou skenovány poprvé. Toto zase ovlivňuje maximální množství proveditelných požadavků na API, což vyžaduje další úpravu chování v podobě omezení počtu vzorků a další.

Pro analýzu datových sad bylo nutné do jednotlivých agentů implementovat, případně integrovat několik nástrojů. Integrované nástroje: VirusTotal a Androguard. Naopak kompletně implementovány byly například tyto nástroje: A Perfect Knife, automatizované spouštění, tvorba logů, databázový systém a další. Všechny nástroje byly nejdříve testovány v odděleném testovacím prostředí, aby se předešlo případnému poškození infrastruktury. Účelem implementace nástrojů je, kromě samotné analýzy datových sad, i zjednodušení práce s mobilním malwarem. Dílčí úlohy systému jsou následující:

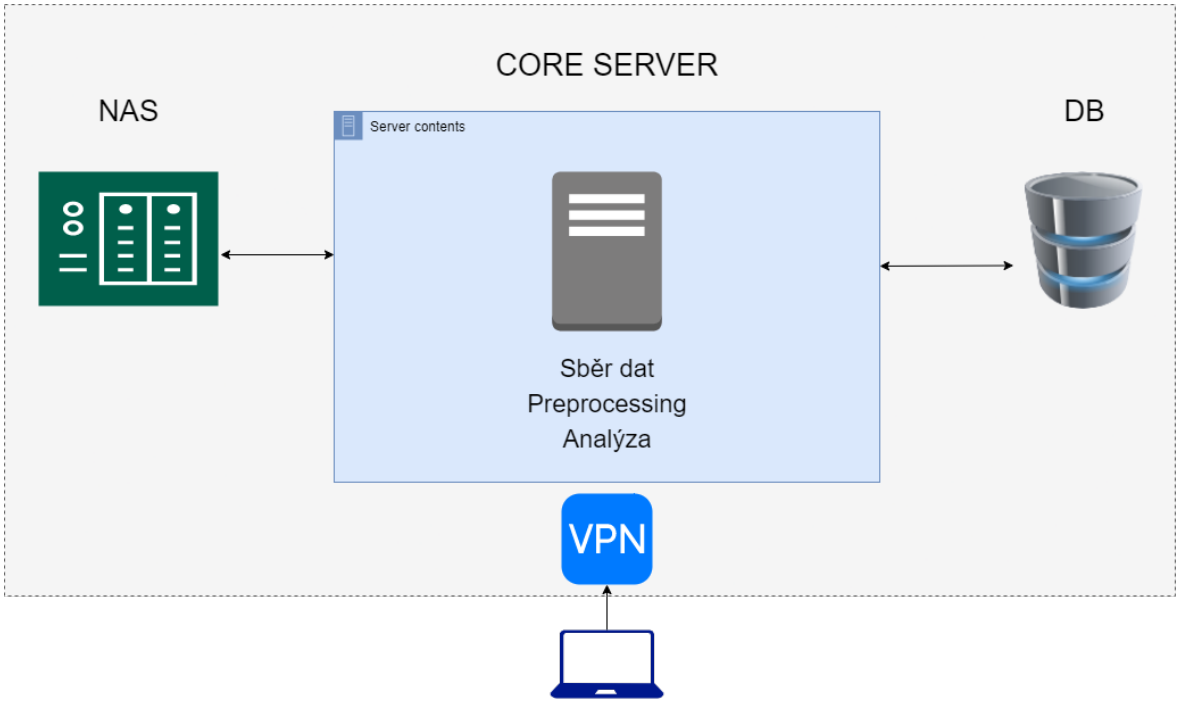
1. Sběr dat
2. Zpracování datové sady do podoby umožňující následnou výzkumnou činnost
    - Preprocessing datové sady
    - Ověření dekompilovatelnosti
    - Ověření malignity - nástroj VirusTotal [5]
3. Extrakce významných charakteristik mobilního malwaru
    - Statická analýza - nástroj Androguard [82]

4. Automatizované spouštění
    - Tvorba logů
5. Propojení s DB systémem
6. Integrace nástrojů do produkční infrastruktury

4.1 Využívaná infrastruktura

Datové sady mobilního malwaru čítají tisíce, někdy i miliony, unikátních vzorků. Proto bylo nutné zajistit dostatečný výpočetní výkon i datové úložiště. Dalším požadavkem na prostředí, tentokráte bezpečnostním, bylo nutné vytvořit prostředí ve kterém bude možné bezpečně a zároveň efektivně pracovat s datovými sadami mobilního malwaru.

Infrastruktura poskytnutá laboratoří PTLAB je izolovaným prostředím, které zajišťuje jak dostatek zdrojů, tak bezpečný prostor pro manipulaci s mobilním malwarem. Z bezpečnostních důvodů nejsou poskytnuty informace o použitých HW a SW komponent. Jsou také vynechány některé názvy a přesné popisy těchto komponent, aby nedošlo k jejich kompromitaci.



Obrázek 4.1 Zjednodušený diagram infrastruktury [zdroj vlastní]

Izolované prostředí, zobrazené na diagramu (4.1), je složeno ze tří základních částí: NAS (Network Access Storage), Core server a databáze. Server je jádrem infrastruktury, kdy jeho úlohou je sběr dat, preprocessing, analýza, celková režie korektního provozu a propojení jednotlivých částí systému a infrastruktury. Všechny tři prvky jsou spolu

pevně svázány, avšak při výpadku jednoho z řízených prvků mohou ostatní nadále pracovat v omezeném režimu. K serveru, jako centrálnímu řídícímu prvku je možné vzdálené připojení s využitím VPN.

V první fázi jsou data sbírána z prostředí serveru a ukládána na síťové úložiště. Zároveň s tímto procesem jsou data zavedena do databáze. Při preprocessingu datových sad se k datům přistupuje na síťovém úložišti, s využitím DB - ta poskytuje informace o balíčcích, které jsou předzpracovány nástroji integrovanými na serveru. Jednotlivé prvky spolu kooperují, přičemž podléhají příslušným agentům. Například agent pro analýzu reaguje na situaci, kdy jsou v databázi ještě neanalyzované balíčky vykonáním analýzy. To je provedeno tak, že z databáze zjistí o které balíčky se jedná a kde jsou uloženy. Následně jsou jednotlivé balíčky zpracovány s využitím přístupu k nim a zjištěná data jsou opět uloženy do databázových záznamů. Tímto způsobem spolu jednotlivé části infrastruktury kooperují a to dle režie korespondujících agentů.

Všechny výpočetně náročné operace probíhají na serveru. Surová data, v tomto případě APK balíčky nad kterými se provádí operace, jsou pak uloženy na síťovém úložišti. Všechny analyticky získané informace o balíčcích, včetně informací o datových sadách jsou pak uloženy v databázi, ze které jsou dále přístupné.

5 SBĚR DAT - VYBRANÉ DATOVÉ SADY KE ZPRACOVÁNÍ

Vybrané datové sady ke zpracování		
ID	Název sady	Přístup
1	AndroidMalware_2018	Public
2	AndroidMalware_2019	Public
3	AndroidMalware_2020	Public
4	AndroidMalware_2021	Public
5	Android Malware Sample Library	Public
6	CICMalDroid 2020	Academic

Tabulka 5.1 Seznam vybraných datasetů se základními informacemi

Sběr dat je jediná část systému, která vyžaduje přímou uživatelskou interakci. Toto chování bylo záměrem a zajistí tak výzkumníkům kontrolu nad celým prostředím. Bylo totiž možné zavést systém, který by sběr dat zajišťoval sám, jen na základě aktuálních souborů. Tento přístup však omezuje kontrolu nad prostředím, to by mohlo vést ke zpracování vzorků v pořadí, jenž by pro výstupy práce nebylo vhodné. Pořadí hodnocení vzorků není důležité pro celkový systém, nýbrž pro zpracování práce, která je zaměřena na určité datové sady a právě manuální vkládání dat do systému toto umožňuje. Je nutné upozornit, že uživatelská činnost v tomto směru zahrnuje nalezení datové sady, vytvoření jejího otisku a následné spuštění automatizovaného nástroje A Perfect Knife. Od chvíle, kdy je tento nástroj spuštěn probíhá již zpracování jednotlivých vzorků z datových sad automatizovaně. Výzkumník tak řídí datové sady, které vstupují do prostředí. Vybrané datové sady a jejich přístupy jsou zobrazeny v tabulce 5.1.

Datasety veřejného charakteru byly zvoleny pro ověření kvality veřejně dostupných datových sad. Vzhledem k jejich velikosti bylo možné těchto sad vybrat pět. Jelikož aktuálnost vzorků je při dalším výzkumu zásadní, je vhodné pracovat s co nejčerstvějšími vzorky, proto byly vybrány sady AndroidMalware 2018-2021. Ze stejného důvodu byla vybrána i datová sada Android Malware Sample Library. Vzorky v sadě by měly být aktuální, všechny byly přidány v roce 2020 a velká část sady je orientována na malware, spojený s onemocněním Covid-19, což je opět využito k reflektování reálných a aktuálních vzorků malwaru do výsledné datové sady. Zároveň tyto datasety vykazují známky nekonzistence, to znamená, že jsou umístěny v komplikovaných strukturách, různých formátech a je tak možné na nich prezentovat sílu preprocessingu, kdy všechny zmíněné sady se podařilo bez komplikací zpracovat a vzorky dále analyzovat. Stěžejní pro tyto datové sady je zejména jejich reálné zařazení do časového intervalu působnosti, pakliže se podaří prokázat jejich zacílení na dané časové úseky, pak jsou tyto vzorky velmi zajímavým vědeckým subjektem, jelikož malware v nich obsažený dostatečně reprezentuje moderní mobilní malware. Kromě časového zařazení je u vzorků také nutné

ověřit jejich dekompilovatelnost, která je klíčová pro provedení statické analýzy. Dalším rizikovým faktorem u těchto sad je integrita malwaru, tedy jestli se opravdu jedná o škodlivé aplikace.

Šestou zvolenou sadou je CIC-MALDroid 2020, tato sada byla vybrána kvůli své vysoké kvalitě a velkému množství vzorků. Dalším důvodem ke zvolení sady byla nonekvivalence celkového počtu vzorků a počtu analyzovaných vzorků v rozřazených kategoriích: adware, banking, benign, riskware a SMS. To znamená, že v jednotlivých kategoriích byl uveden počet analyzovaných vzorků, jenž je nižší než celkový počet. K této situaci došlo kvůli dynamickému sběru charakteristik mobilního malwaru. Již bylo zmíněno, že využití pouhé statické analýzy by mohlo mít za následek větší množství analyzovaných vzorků, než je tomu u dynamické analýzy, to zejména proto, že není nutné používat emulátor, nebo reálné zařízení, čímž je možné vyhnout se různým problémům, které jsou s tímto spojeny. Toto tvrzení potvrdila i analýza datové sady, kde bylo pouhou statickou analýzou vytěženo větší množství aplikací, než je udáváno v jednotlivých kategoriích.

U datové sady CIC-MALDroid 2020 je nutné poznamenat, že obsahuje, na rozdíl od předešlých, i vzorky legitimních aplikací. Všechny statistické údaje, jako např. celkový počet nasbíraného mobilního malwaru, jsou počítány bez těchto legitimních aplikací. Práce se totiž zaměřuje na datové sady mobilního malwaru. I když benigní aplikace byly dále zpracovány, nejsou obsaženy ani ve výsledné datové sadě. Záměr byl vytvořit datovou sadu mobilního malwaru garantované kvality, ke které je možné z různých zdrojů (např. AndroZoo), vytvořit balancovanou datovou sadu legitimních aplikací.

Ačkoliv zde nebyl předpoklad problémů spojených zejména s veřejnými datovými sadami, například výskyt malwaru na jinou platformu, nebo jinak nevalidních vzorků. Stále byl nutný preprocessing, jednotlivé balíčky byly pojmenovány dle různých konvencí i v rámci této datové sady. Preprocessing byl nutný i kvůli velkému množství balíčků, kdy nebylo možné tyto verifikovat manuálně. K této činnosti byl využit nástroj A Perfect Knife, protože zvládá předzpracování dat v různých formátech a zároveň dokáže spolehlivě vyloučit nevalidní soubory, navíc byl optimalizován pro běh v serverovém prostředí. U této datové sady bylo nutné, kromě zařazení do časového intervalu působnosti, ověřit integritu dat ve smyslu jejich škodlivosti. Při tak velkém množství vzorků se mohlo stát, že některé vzorky nebyly správně zařazeny do příslušné kategorie. Zařazení vzorků do časového intervalu dle působnosti však hraje opět klíčovou roli při tvorbě datových sad. Ačkoliv je u datasetu uveden sběr mezi lety 2017-2018, je to pouze časový horizont ve kterém byly vzorky sbírány, nikoliv období jejich působnosti. U sady existuje potenciální riziko nedostatečné reprezentace moderního mobilního malwaru.

Posledním důvodem volby uvedených datasetů je porovnání kvalit veřejných a aka-

demických datových sad.

5.1 Fyzické zpracování - Sběr vzorků

Sběr infikovaných aplikací probíhal v rámci získávání dat z veřejně dostupných datových sad a ze sad s akademickým přístupem, které byly vybrány ke zpracování. Benigní aplikace jsou zpracovávány laboratoří PTLAB, tato část práce však není veřejná, proto je praktická část zaměřena převážně na vzorky mobilního malwaru.

Celkem bylo nasbíráno 13 411 unikátních vzorků malwaru z 6 datových sad, včetně jednoho testovacího vzorku, ten není přímo začleněn do datových sad, jelikož sloužil primárně pro testování v produkčním prostředí. Datové sady vybrané ke zpracování obsahují jednotlivé vzorky, fyzický sběr je tak možné rozdělit na sběr datových sad a sběr jednotlivých vzorků, rozdíly ve zpracování prezentuje obrázek 5.1. Sběr jednotlivých vzorků ze sady již probíhá automatizovaně a bude popsán v sekci A Perfect Knife.

Fyzický sběr datové sady	Fyzický sběr vzorků
1. Nalezení sady 2. Fyzické uložení 3. Vytvoření SHA256 otisku 4. Zavedení sady do prostředí	1. Preprocessing datové sady 2. Nalezení vzorků v datové sadě 3. Vytvoření SHA256 otisku 4. Standardizace názvosloví 5. Ověření dekompilovatelnosti 6. Zavedení vzorku do prostředí

Obrázek 5.1 Rozdíl ve zpracování sady a vzorku [zdroj vlastní]

Nalezení sad probíhalo s využitím otevřených zdrojů. Jedná se o datové sady popsané výše, tyto byly získány z primárních zdrojů. U každé sady byl před zahájením výzkumné činnosti zaznamenán SHA256 otisk datové sady, pro jednoznačnou identifikaci. Zároveň slouží tyto otisky jako verifikace autenticity datové sady. Zejména u veřejně dostupných datových sad se často stává, že vzorky jsou přidávány do kolekce postupně, SHA256 je pro kontrolu změny obsahu datové sady ideální.

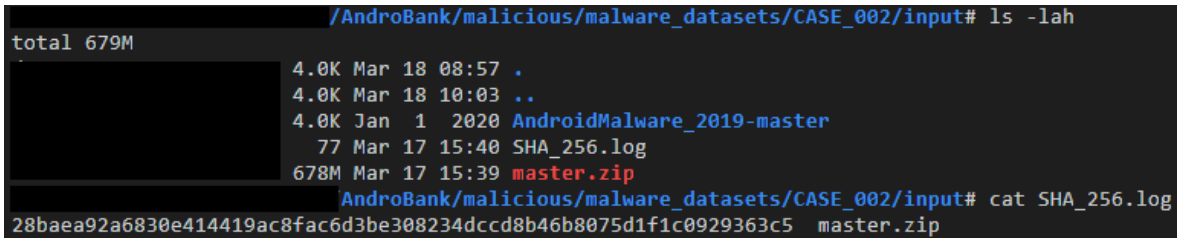
Všechny sady byly nejprve uloženy na síťové úložiště s využitím serveru. Následně byly vytvořeny jejich SHA256 otisky. Takto nasbírané datové sady, které byly nalezeny, fyzicky uloženy a otisknuty bylo možné zavést do prostředí systému a uložit do databáze. Veškerá činnost spojená se sběrem a uskladněním datových sad, potažmo vzorků, byla prováděna na serveru za využití zabudovaných Linuxových nástrojů. Základní in-

formace ukládané v databázi jsou například: název sady, SHA256 otisk, zdroj, cesta a obecné informace (datum přidání záznamu a další). Tímto způsobem byly sbírány a zpracovány všechny sady. Na obrázku 5.2 je ukázka korektně uskladněné datové sady, obsahuje následující části:

master.zip: Jedná se o originální soubor obsahující jednotlivé vzorky z datové sady.

SHA_256.log: Otisk originální sady, vytvořený zabudovaným nástrojem sha256sum.

AndroidMalware_2019-master: Datová sada zavedená do prostředí a připravená ke zpracování.



Obrázek 5.2 Uskladněná datová sada [zdroj vlastní]

6 NÁSTROJE PRO ZPRACOVÁNÍ SAD A JEJICH INTEGRACE

Takto uložené datové sady slouží i jako záloha při destrukci vstupního souboru, což se může stát za předpokladu nevalidních, či poškozených dat. Datové sady v tomto stavu však nemají velký výzkumný dopad. Je možné zkoumat jejich charakteristiky black-box metodou například zjištění, zda je sada balancovaná, nebo získání obecných charakteristik, jako velikost a další, nicméně k umožnění následující výzkumné činnosti je potřeba tyto datové sady dále zpracovat.

Zpracování do podoby umožňující další výzkum je zaručeno následující sadou nástrojů. Implementace a integrace všech nástrojů, byla realizována v jazyce Python (3.0+). Tyto nástroje spolu přímo souvisí, avšak všechny na sobě nejsou přímo závislé. To při potenciálním problému zaručí chod systému, i když v omezeném režimu. Systém má určité limity, jedním z nich je například maximální počet požadavků odeslaný službě VirusTotal. Z toho logicky plyne omezení na maximální počet vstupních souborů, které jsou denně vyhodnoceny AV testy. Jelikož systém má ambici být aktivním i v následujících letech v laboratoři PTLAB, není možné každý den manuálně spouštět dílčí úlohy. Systém proto funguje autonomně, avšak je možné do jeho chodu zasahovat, například kvůli údržbě, nebo při zjištění problémů v logu. Logy jsou další funkcionalitou plynoucí z omezení navrženého systému, kdy z důvodu automatizace celého procesu, není možné, ani praktické provozovat takový systém bez logování provozu.

Základním kamenem, kolem kterého byl sestaven celý systém je A Perfect Knife, který zaručí, že se do systému dostanou jen validní APK soubory, se kterými je možné dále pracovat v rámci statické analýzy. Navazuje na něj nástroj VT_observer, který denně vyhodnocuje všechny soubory, připravené k ověření integrity vzorku, ve smyslu jeho malignity. Z vyhodnocených souborů jsou získávány staticky extrahovaná data nástrojem Androguard. Nástroje Androguard a VT_observer jsou spolu pevně svázány a jsou integrovány do systému, jelikož se jedná o již existující službu poskytující API. Veškerý provoz celého systému je zaznamenán do rutinních logů a o správném běhu systém informuje i pomocí zpráv zasílaných botem přes Telegram [85]. Všechna data, která se podaří získat z datových sad, potažmo jednotlivých vzorků, jsou pak ukládána do databáze.

6.1 Databáze

Vzhledem k využití Pythonu, ale i kvůli vyhovujícím limitům databáze byla zvolena databáze PostgreSQL [86]. Tato databáze je hostována na serveru a je důležitou součástí celého systému. Všechna analyticky zjištěná data, jsou ukládána právě do databáze.

Databázi bylo nutné navrhnout, vytvořit, otestovat a zabezpečit hesly. Kromě procesu tvorby databáze musely být patřičně upraveny nástroje, aby umožňovaly integraci databáze do systému. Veškeré operace prováděné nad databází jsou zaznamenávány do rutinních logů.

Samotná databáze poté existuje ve dvou verzích - produkční (server) a testovací (lokální virtuální prostředí). Obecný popis ukládaných informací:

- datové sady mobilního malwaru
- jednotlivé vzorky mobilního malwaru
- vyhodnocení souboru jednotlivými AV řešeními
- statická analýza
  - informace poskytnuté službou VT (počet souborů, data modifikace a další)
  - informace poskytnuté službou Androguard (komponenty aplikace, oprávnění a další)

6.2 Automatizované spouštění

Automatizované spouštění v předem specifikovaném času není nikterak složitým problémem. Pro tento úkol lze nasadit již existující a využívaná řešení zabudovaná v OS Linux. Stejně tak lze implementovat spouštění (scheduling) skriptů i v jazyce Python, který disponuje již připravenou knihovnou použitelnou pro automatizaci.

První zmíněný postup - využití zabudovaných nástrojů, je ideální při vykonávání akce v určitou dobu a periodicky. Požadavek laboratoře PTLAB však byl simulovat chování člověka v práci. Ačkoliv lidé do práce chodí většinou na danou pracovní dobu, tento proces nemusí být přesný. V praxi se tak stane, že zaměstnanec začíná pracovní aktivitu v různých časových intervalech. Tyto rozmezí pak bývají ovlivněny různými vnějšími vlivy, jako například hustota dopravy při přesunu na pracoviště, počasí a další. Bylo nutné vytvořit algoritmus, který bude svou pracovní činnost vykonávat v určitém časovém rámci, nikoliv však ve stejný čas. To je do jisté míry proveditelné v jazyce Python, avšak zmíněný způsob schedulingu se projevoval tak, že ve vlákně čekal na určitý čas. Zmíněné postupy nebyly úplně vyhovující a proto byl vytvořen vlastní způsob plánování spuštění.

Tento způsob pracuje se skripty psanými v Pythonu, využívá však terminál a díky tomu spolupracuje s nástrojem "at", který je zabudovaný v operačním systému Linux a jeho využití je vhodné zejména při plánování úloh, které jsou spuštěny pouze jednou. Díky kombinaci obou zmíněných přístupů se podařilo vytvořit nástroj pro automatizované plánování úloh, který je interně označen jako Dealer, protože řídí každodenní zásobování systému infikovanými aplikacemi a s tím spojené rutiny.

Nástroj pro automatizované spuštění má dvě základní funkcionality: generování času spuštění, naplánování spuštění

Pro simulaci začátku pracovní aktivity je nutné vygenerovat čas, ve který se bude rutina spouštět. K tomuto byl využit jazyk Python, implementující náhodné generování čísel s využitím Gaussova (normálního) rozdělení. Vzhledem k tomu, že čísla generovaná pomocí Gaussova náhodného generátoru čísel jsou desetinná, probíhá také korekce hodnoty, aby korespondovala s jednotkami času, které jsou celočíselné. Proces generování je poté vcelku jednoduchý, kdy jsou generována dvě čísla. První číslo reprezentuje hodinu, střední hodnota byla zvolena na 7, sigma je rovna 1. Pro minuty jsou parametry 30 a 10 ve stejném pořadí. Toto má za následek generování v časech kolem sedmé hodiny ranní a přibližně kolem třicáté minuty ve vygenerované hodině, ukázka generování časů je demonstrována na obrázku 6.1.

```
Internet connected: True
VPN connected: True
Database connected: True

Next start time: 7:18
Next start time: 7:27
Next start time: 8:28
Next start time: 5:10
Next start time: 6:20
Next start time: 7:10
Next start time: 8:27
Next start time: 8:18
Next start time: 6:27
Next start time: 8:32
```

Obrázek 6.1 Generátor času spuštění [zdroj vlastní]

Takto vygenerovaný čas je používán jako parametr pro příkaz at. Dalším krokem je složení příkazu pro spuštění a jeho vykonání v terminálu. To je proveditelné v rámci Python skriptu, jak je patrné z obrázku 6.2.

```
cmd = f"echo \"{run_path}\" | at {hh}:{mm}"

proc = subprocess.Popen(cmd, shell=True, stdin=subprocess.PIPE,
    stdout=subprocess.PIPE, stderr=subprocess.PIPE)
```

Obrázek 6.2 Naplánování úlohy [zdroj vlastní]

6.2.1 Tvorba logů

Na obrázku 6.2 je ukázka plánování spuštění, za povšimnutí na obrázku stojí absence klasické konvence spouštění skriptů v Pythonu, dle které by příkaz byl upraven následovně: *"... python {run_path} ..."*.

Důvodem složení příkazu je fakt, že není spouštěn přímo zmíněný skript, nýbrž bash skript. Tento bash skript, kromě samotného spuštění programu v Pythonu, vytváří adresářovou strukturu pro zaznamenávání logů a zároveň umožňuje samotné logování. Propojení automatizovaného spouštění a tvorby logů v jeden funkční celek by mohl být časově velmi náročný proces. Je však opět možné využít zabudovaných funkcí v Linux OS, tentokrát je používáno přesměrování výstupů.

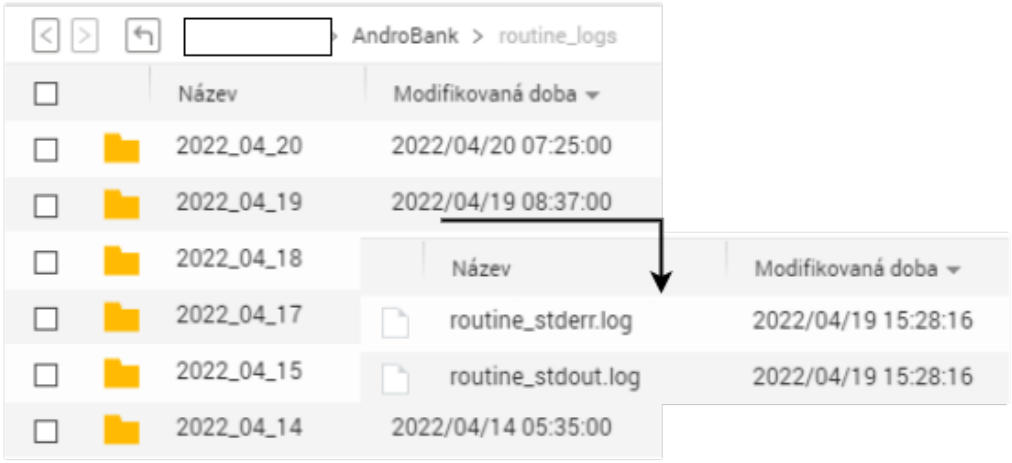
Všechny nástroje, ať implementované, nebo integrované zaznamenávají svou činnost do terminálu. Přesměrování jejich terminálových výstupů umožnilo logování důležitých informací do rutinních zpráv. Naprostou většinu logování je tak možné vyřešit v rámci bash skriptu, který je zobrazen na obrázku 6.3.

```
#!/usr/bin/bash


date=$(date '+%Y_%m_%d')
scriptPath="[redacted]
workDir="[redacted]/AndroBank/"

routineDir="routine_logs/"

if [ -d "$workDir" ]; then
  echo "WorkDir: ${workDir} - Exists, starting Dealer script..."
  routinePath=$workDir$routineDir
  logPath=$routinePath$date
  echo "Daily logs can be found in: ${logPath}"
  mkdir -p "$logPath"
  /usr/bin/python3 $scriptPath > >(tee -a "$logPath/routine_stdout.log")
   2> >(tee -a "$logPath/routine_stderr.log" >&2)
else
  echo "Work Dir: ${workDir} can not be found, script can not continue"
fi
```

Obrázek 6.3 Bash script "run.sh" [zdroj vlastní]

Z obrázku je patrné, že je nejprve zjištěno aktuální datum, které je upraveno do požadovaného formátu. Následně jsou inicializovány proměnné, které byly anonymizovány, protože se jedná o cesty. Dalším krokem, je kontrola existence "workDir" - tento adresář je umístěn na síťovém úložišti a jedná se o kořenový adresář pro navržený systém. Tento adresář je pojmenován: "AndroBank". Tato kontrola slouží zejména pro ověření připojení k NAS úložišti, pokud by nebyl adresář nalezen je vysoká pravděpodobnost, že nastal problém se síťovým úložištěm.

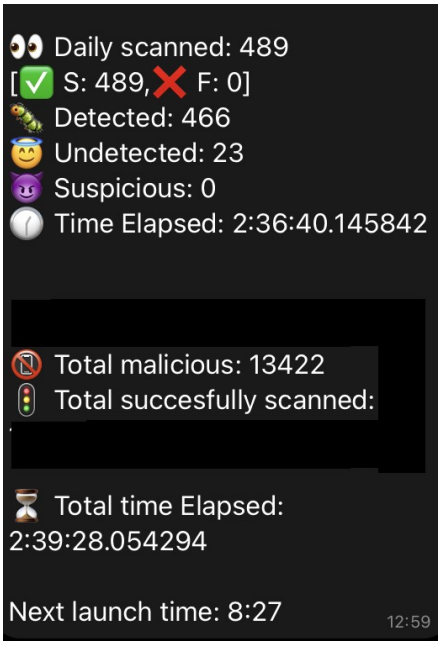
Pakliže je síťový adresář AndroBank dosažitelný, provede se vytvoření adresáře ve formátu "YYYY-MM-DD", ve kterém jsou umístěny rutinní logy. Posledním krokem je spuštění samotného Python skriptu, jak již bylo uvedeno výše, tedy za pomocí klasické konvence. Příkaz využívá nástroje tee, který zaručí přesměrování výstupu, jak do konzole, tak do souboru. Výsledkem je struktura, ve které jsou adresáře pro každý den, kdy systém pracuje. V jednotlivých adresářích jsou pak umístěny rutinní logy, jak je zobrazeno na obrázku 6.4, který byl pořízen z prostředí NAS.



Obrázek 6.4 Rutinní logy [zdroj vlastní]

6.2.2 Telegram Bot

Pro vylepšení uživatelského zážitku a také pro zjednodušení celého procesu byl vytvořen bot, který informuje o stavu systému. Díky tomu je možné sledovat pouze zprávy zasílané tímto botem, který informuje o případném selhání. Toto značně zjednodušuje práci, není totiž nutné sledovat celý běh všech agentů a případné problémy v souborech logů. Na místo toho jsou zasílány zprávy v určitých fázích systému, které informují o spuštění rutinních služeb a jejich průběhu. Po dosažení denních limitů na API, bot s využitím databáze zjistí denní přírůstky a také celkové statistiky. Přístup k botovi je platformně nezávislý, to umožňuje přihlášení k odběru bota z mobilního zařízení, nebo třeba PC. Ačkoliv je bot soukromý, umožňuje i pozvání dalších členů týmu.



Obrázek 6.5 Bot report [zdroj vlastní]

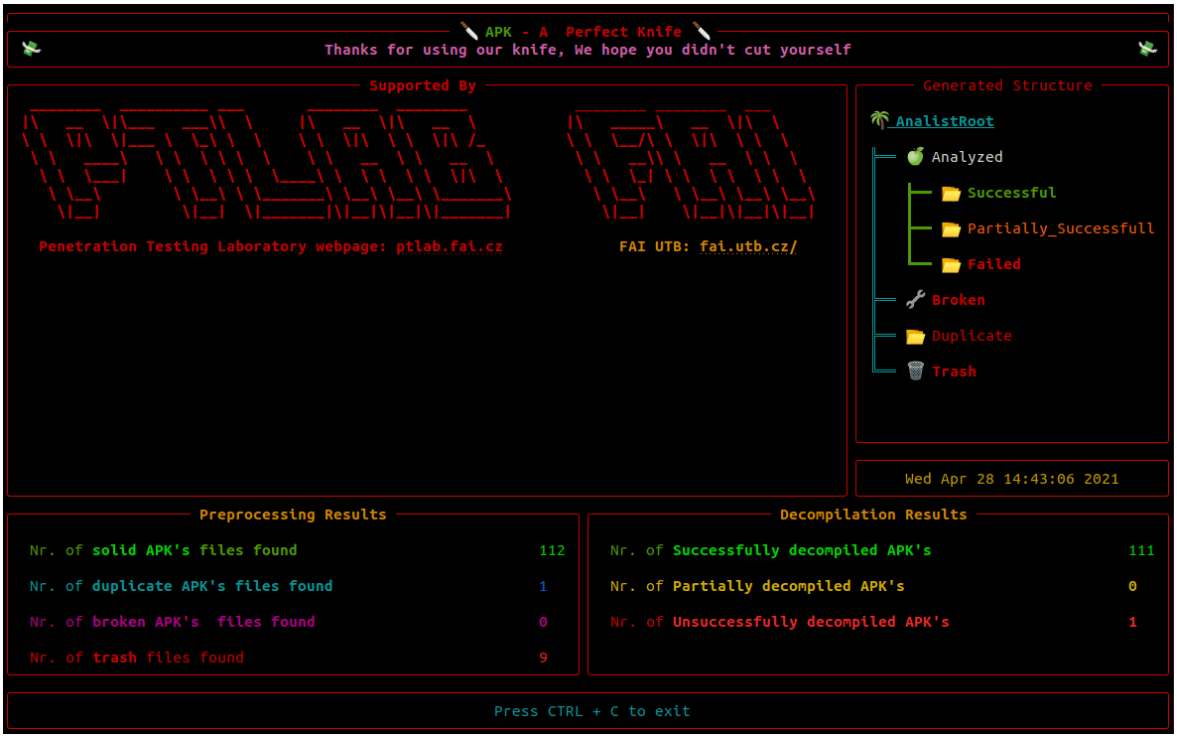
Bot tak nemá praktické využití pouze za scénáře, kdy nastane problém. Jeho hlavní funkcí je kontrolovat celý proces a informovat o aktuálních stavech v systému. V praxi bot každé ráno potvrdí, že začal pracovat a na konci celé činnosti informuje o zpracovaných aplikacích a času dalšího spuštění. Pokud se z nějakého důvodu nepodaří naplánovat spuštění, bot místo hodin uvádí hodnotu "FAILED". Částečný a anonymizovaný report je pak zobrazen na obrázku 6.5. Interní název tohoto bota je AndroSnitch, což má naznačovat, činnost nástroje, kterou je pravidelné informování o systému. Bot má soukromý přístup udělený na základě pozvánky autorem, což představuje formu zabezpečení proti odposlechu.

6.3 APK - A Pefrect Knife

Automatizovaný nástroj pro preprocessing APK balíčků a hromadné dekompilace. Inspirací k tvorbě bylo problematické zpracování balíčků a výskyty nevalidních dekompilací, zejména kvůli nepředzpracování vstupů. Výstupem nástroje je datová sada garantované kvality, připravená pro další výzkumnou činnost. Původní verze nástroje vznikla v roce 2021 pro potřeby penetrační laboratoře PTLAB. Nástroj byl vyvíjen ve spolupráci s Bc. Janem Kinclem. Práce byla jednoznačně rozdělena, kdy Bc. Jan Kincl implementoval část pro hromadné dekompilace. Autor této práce vyvíjel část, týkající se předzpracování dat, standardizace balíčků, údržby a integrace do systému. Nástroj byl prezentován v roce 2021 na soutěži STOČ 2021 [87], kde A Perfect Knife získal 1. místo v kategorii S5 - Aplikace pro 3D modelování, zpracování obrazu a dat. V roce 2021 byl také A Perfect Knife prezentován na konferenci Kybernetická bezpeč-

nost 2021: Řízení procesů a aplikace moderních technologií [88], kde kromě samotného nástroje byly diskutovány také aktuální hrozby na poli mobilní bezpečnosti.

V dalším roce probíhala údržba a úpravy nástroje, které již byly samostatnou prací a vycházely z nabytých praktických zkušeností.



Obrázek 6.6 A perfect Knife [zdroj vlastní]

Součástí nástroje je také GUI, jehož design je inspirován retro stylem. Důvod existence GUI je poměrně jednoduchý. Celá činnost nástroje je logována a uživatel je informován o jednotlivých krocích, tyto jsou popsány níže. Za běhu softwaru je však vše vztaženo k jednotlivým zpracovávaným souborům, nikoliv celé sadě. Proto je na konci běhu programu zobrazen informační layout, který je na obrázku 6.6. Layout informuje výzkumníka o výsledku celého procesu. Zároveň je celý nástroj optimalizován pro serverové prostředí, to znamená, že běží v terminálu, proto nemá k dispozici moderní nástroje pro tvorbu komplexních uživatelských rozhraní.

Motivací k využití nástroje v rámci této diplomové práce byla problematika získávání kvalitních datasetů ke zkoumání, kdy výzkumníci čelí mnohým problémům. Datové sady většinou není možné jednoduše stáhnout a okamžitě zkoumat, jelikož sady jsou často nekonzistentní, obsahují nevalidní vzorky, duplicity, poškozené archivy, archivy s hesly a skýtají další potenciální problémy. To vše následně může vést ke znehodnocení výzkumu. Reakcí na tyto problémy je část orientovaná na preprocessing. Nástroj také reaguje na problém časově náročného procesu dekompilace, který řeší druhá část nástroje pro hromadné dekompilace.

A Perfect Knife z předem neznámé vstupní datové sady, bez zásahu výzkumníka, vyfiltruje všechny validní APK soubory a následně ověří, pomocí dekompilace, jejich analyzovatelnost. Nástroj tak zásadně usnadňuje práci s datovými sadami a umožňuje výzkumníkům soustředit se na řešenou problematiku.

6.3.1 Preprocessing - Standardizace a čištění datových sad

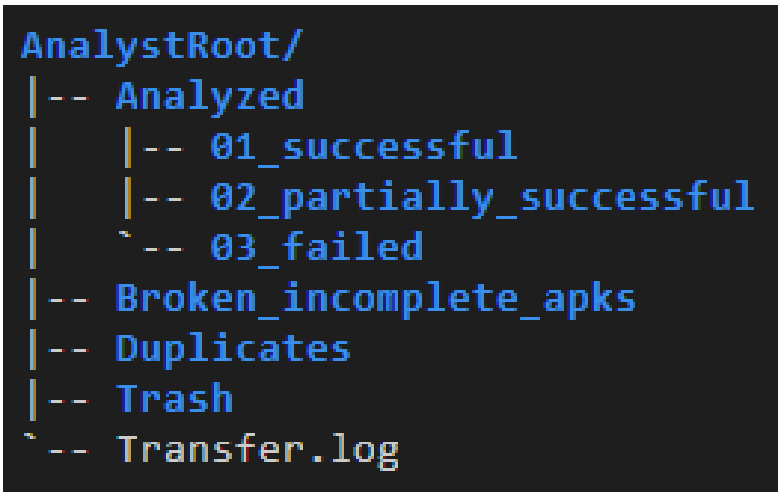
Výzkumníci při práci s datovými sadami řeší různá úskalí, která není nutné u analýzy jednotlivých balíčků brát v potaz. Například při manuální analýze aplikace je provedena dekompilace a následně jsou pomocí statické analýzy zjišťovány důležité informace o balíčku. Pokud by analyzovaný balíček byl jiného formátu, než APK, pak na to analytik patřičně reaguje, třeba jeho vyřazením, nebo dalším prozkoumáním. Při zpracování tisíců APK balíčků to však není možné, celý proces je časově poměrně náročný a samotná dekompilace balíčků by vyžadovala desítky hodin. Proto A Perfect Knife nejprve předzpracovává data, aby maximálně zamezil jakýmkoliv problémům při další analýze.

Preprocessing datové sady zahrnuje:

- tvorbu workspace.
- Dekomprimace archivů.
- Dekomprimace archivů s heslem (i AES).
- Rekurzivní vyhledávání APK balíčků v komplexních strukturách.
- Ověření integrity a standardizace vzorků.
- Odstranění duplicit, poškozených, nevalidních.
- Ochrana před spuštěním nastrčených souborů.

Základní ohledání datových sad odkrylo různé konvence názvosloví, ty se liší napříč sadami. Mezi zaznamenané formáty patří: <MD5>.apk, <SHA256>.apk, <SHA256>, <rodina_malwaru>.apk, <SHA256>.<MD5> a další. Rovněž vyšlo najevo, že v datových sadách veřejného charakteru se mohou vyskytovat nejen nevalidní soubory (readme.md, různé obrázky a jiné), ale také malware na jiné platformy. Zda se jedná o záměr je diskutabilní, avšak mohlo se jednat i o chybu. Je ale fakt, že se podařilo najít spustitelné soubory na operační systémy Windows a Linux, které byly službou VirusTotal vyhodnoceny jako škodlivé. V rámci preprocessingu byly tyto soubory odhaleny a vyřazeny z datové sady mobilního malwaru. To znamená, že A Perfect Knife do jisté míry chrání výzkumníka, před vykonáním škodlivého kódu pro jiné platformy a zamezí zavedení těchto nechtěných souborů do systému.

Preprocessing v první fázi inicializuje pracovní prostředí - workspace. Jsou vytvořeny adresáře, do kterých budou rozřazeny všechny nalezené soubory. Na obrázku 6.7 je zobrazena výsledná struktura, která se od pracovního prostředí liší zejména absencí adresáře Apks_tmp. Tento adresář slouží jako dočasný pro nalezené balíčky a také jako vstup do dalšího kroku. Všechny nevalidní soubory, jiných formátů, než APK jsou přesunuty do adresáře Trash, to umožní jejich případné ohledání a zároveň zamezí další práci s nimi. Do adresáře Trash byl přesunut i již zmíněný malware pro jiné platformy.



Obrázek 6.7 Finální struktura [zdroj vlastní]

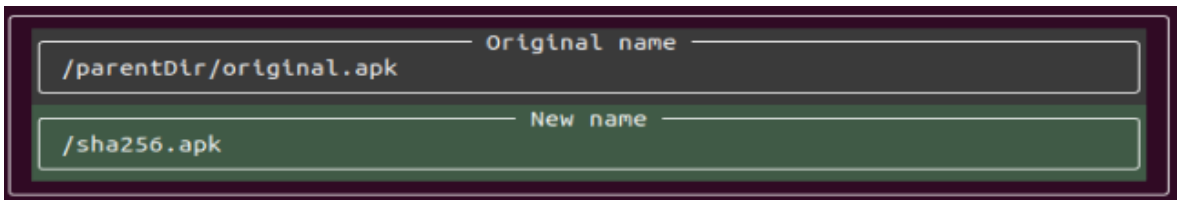
APK balíček může mít však i jiné přípony, než ".apk", jak bylo zmíněno. Není tak při identifikaci možné se na suffix spolehnout. Zároveň bývají problematické i tradiční nástroje na zjištění MIME typu, protože APK balíček je často označen jako ZIP archiv, méně často poté jako APK. Struktura souborů a adresářů uvnitř APK balíčku umožňuje jeho rozpoznání právě na základě jeho struktury. V teoretické části byla popsána struktura APK balíčku a jeho povinné části. Detekce APK balíčků je tak založena na principu, kdy probíhá kontrola na přítomnost povinných souborů v kořenovém adresáři APK balíčku. Tento postup umožňuje rozpoznání aplikace pro Android s mnohem větší přesností, než tomu bylo při testování jiných postupů. Pakliže chybí některá povinná část balíčku, je tento přesunut do adresáře pro nekompletní, či jinak poškozené APK balíčky.

U mobilního malwaru pro operační systém Android je, při legitimní distribuci pro výzkumné účely, často využíván ZIP archiv s heslem, ať už kvůli zamezení nechtěné manipulace s malwarem a nebo také pro předcházení smazání vzorku AV programem. Manipulace s mobilním malwarem na zařízení osazeným AV řešením může být velmi problematická, ačkoliv není typické pracovat s tímto typem softwaru na osobním počítači, ale spíše ve virtuálních prostředích, která jsou k tomu patřičně přizpůsobena - tento přístup může vést ke smazání vzorku, případně ztráty jeho částí (dex soubory a další). Z těchto důvodů bezpečnostní analytici preferují komprimaci APK balíčků

do zašifrovaných ZIP archivů. To s sebou přináší další problém, který je nutné vyřešit v rámci preprocessingu. Starší soubory mohou být zašifrovány standardním ZIP 2.0 šifrováním, kdežto při šifrování nových souborů bývá ve výchozím nastavení používán bezpečnější AES. Nástroj proto implementuje řešení, které je schopno odemknout a následně dekomprimovat jak starší, tak novější formát ZIPu. Přičemž heslo je definováno dle užívané konvence pro mobilní malware a sice "infected", proto jsou tyto zašifrované ZIP archivy někdy označovány jako "infected ZIP". Tato funkcionalita výrazně přispívá k vytěžení maximálního počtu balíčku z datové sady.

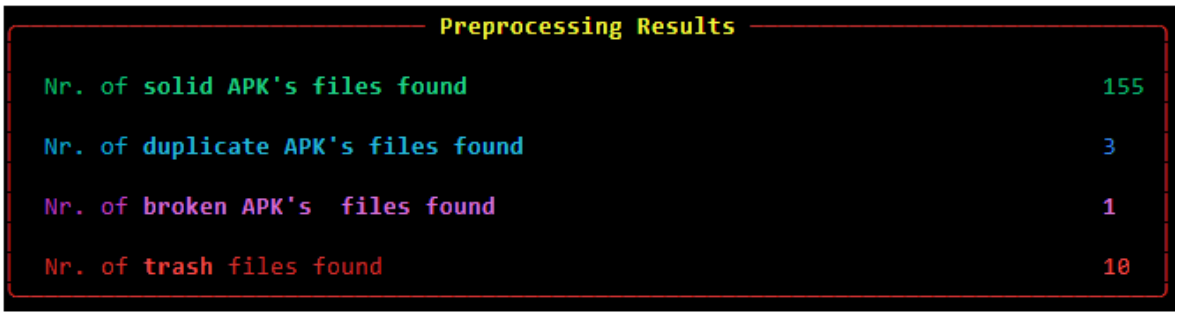
Pakliže se v takovém archivu nachází validní APK balíček splňující specifika struktury, je jeho název standardizován dle následující konvence: <SHA256>.apk. Tato konvence je standardně využívána v laboratoři PTLAB a to zejména kvůli jednoznačné identifikaci. V praxi balíčky používají i formát MD5, který je však zastaralý a tak není vyhovující, na rozdíl od SHA256. Každý validní a standardizovaný balíček je poté přesunut do adresáře Apks_tmp. Ten slouží jako vstupní adresář pro další krok, kterým je dekompilace. Tímto způsobem je zpracována celá surová datová sada, ze které jsou vyloučeny jevy mající negativní vliv na výzkum.

V rámci standardizace názvů vzorků může dojít ke ztrátě informace, pokud byl vzorek pojmenován například podle rodiny. Jeho zpětné dohledání může být poměrně problematické. Proto je součástí standardizace i tvorba transfer logu, který ukládá starý a nový název balíčku do přiloženého Transfer logu. Do transfer logu je ukládán nový název a starý název s rodičovským adresářem, to je graficky zobrazeno na obrázku 6.8.



Obrázek 6.8 Princip transfer logu [zdroj vlastní]

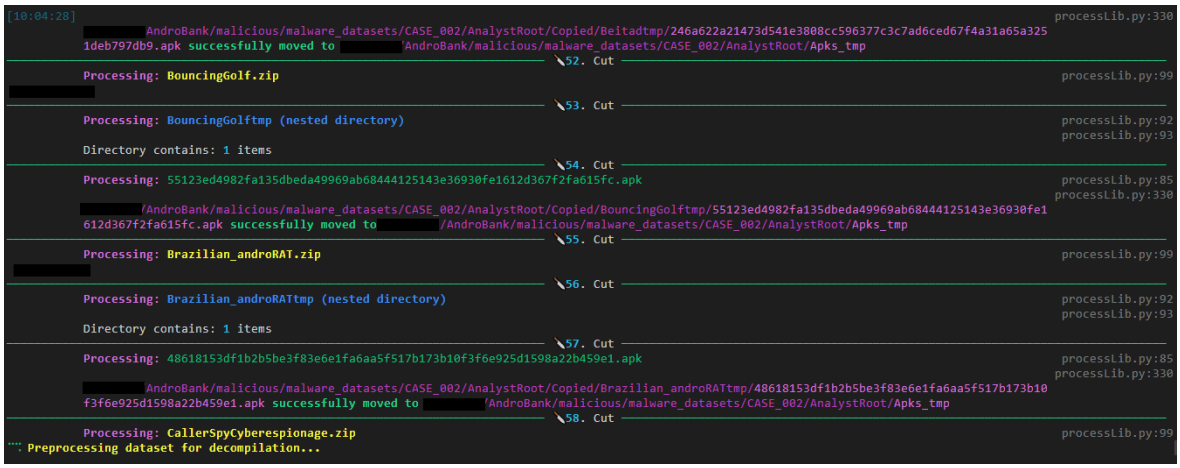
Výstupem preprocessingu je adresář obsahující pouze standardizované a validní APK balíčky připravené pro ověření jejich dekompilovatelnosti. Kromě tohoto se za výstupy nástroje dají považovat i roztříděné nevalidní soubory. U těchto souborů probíhá manuální ohledání, čímž byly eliminovány chyby v třídění, ale také je díky tomu možné analyzovat nastrčené soubory aniž by ovlivnily celý systém. Informace o celém preprocessingu jsou na konci běhu zobrazeny uživateli v rámci výsledného layoutu, ukázka je zobrazena na obrázku 6.9.



Obrázek 6.9 Preprocessing - výstup [zdroj vlastní]

Jelikož celý běh nástroje může trvat i několik hodin - je důležité výzkumníka informovat o aktuálním průběhu. Bylo možné použít klasické metody pro výpis textu do konzole, avšak pro zpříjemnění uživatelského zážitku jsou tyto provozní logy formátovány, využívají barevná schémata a ukazují aktuální stav průběhu v přehlednější formě, než je tomu u černobílého textu.

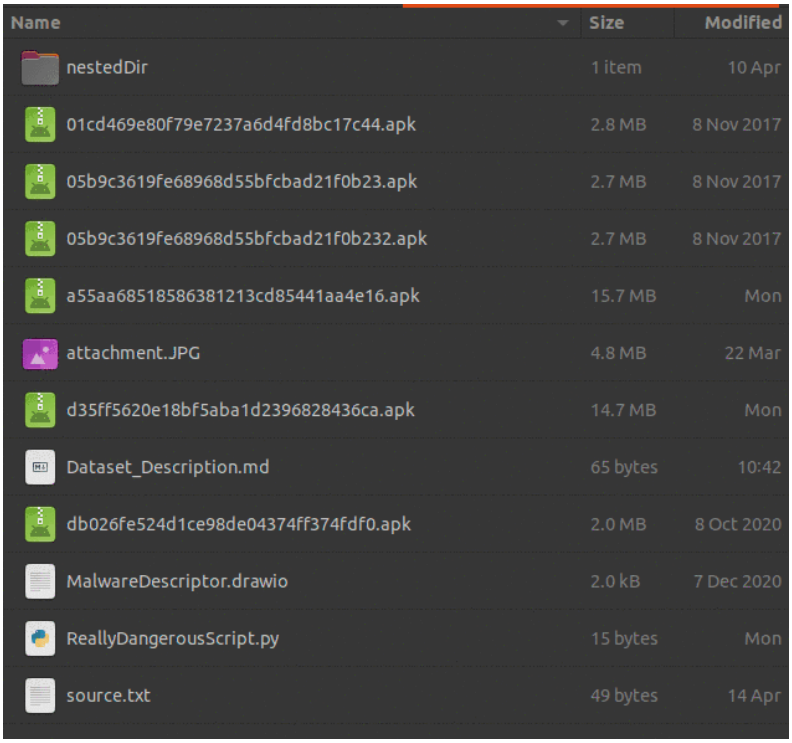
Průběh preprocessingu je demonstrován obrázkem 6.10. Preprocessing provádí "řezy" struktury, což umožňuje projít rekurzivně veškerý obsah datové sady. Zejména u veřejných datových sad bývá řezů i několikanásobně více než APK balíčků, kvůli komplexním strukturám. U akademických datových sad, které jsou distribuovány v kvalitním stavu je obvyklý počet řezů roven celkovému počtu APK balíčků plus jedna. Součástí ukázky průběhu nástroje je také zpracování šifrovaného ZIP archivu. Při zjištění, že se v archivu nachází APK soubor, je ZIP dekomprimován do dočasné složky, ze které je extrahován APK balíček, případně balíčky.



Obrázek 6.10 Preprocessing - průběh [zdroj vlastní]

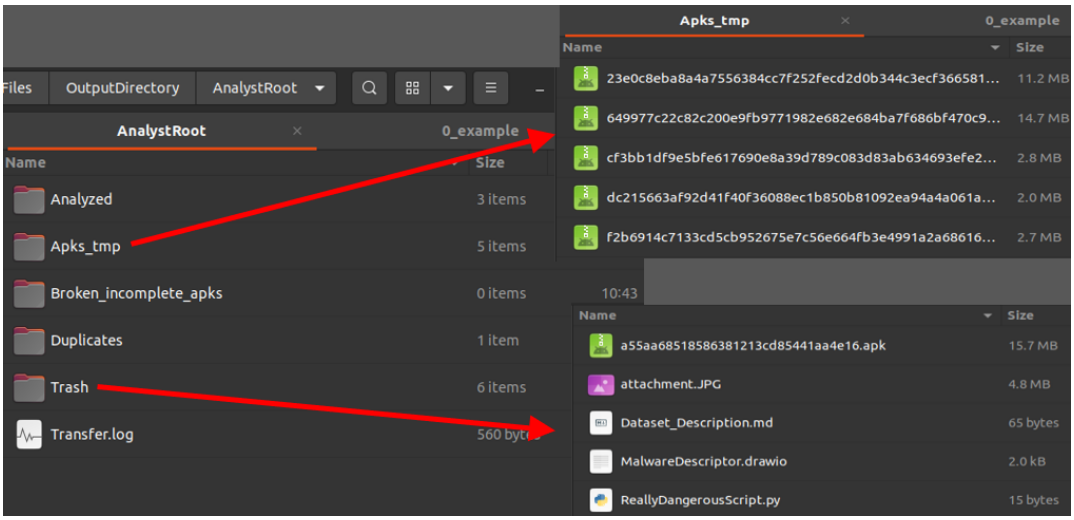
V rámci testování byla vytvořena prototypová datová sada obsahující problémy, které se mohou při zpracování vyskytnout, jako například: zanoření, duplicity, nastrčený malware zabalený jako APK balíček a další nevalidní soubory. Dále je v sadě obsaženo několik validních APK souborů, kdy jsou zašifrovány v ZIP archivu. Kom-

pletní surová vstupní datová sada je zobrazena na obrázku 6.11, APK balíčky jsou pojmenovány stejnou konvencí: <MD5>.apk. Již bylo zmíněno, že MD5 hash není vhodná pro danou problematiku.



Obrázek 6.11 Surová datová sada [zdroj vlastní]

Předzpracovaná - očištěná a standardizovaná datová sada je prezentována na obrázku 6.12 - zde je v adresáři Trash i zmíněný nastrčený malware, jedná se o spustitelný soubor pro Windows (PE32), zabalený do APK. V adresáři Apks_tmp jsou poté standardizované APK soubory.

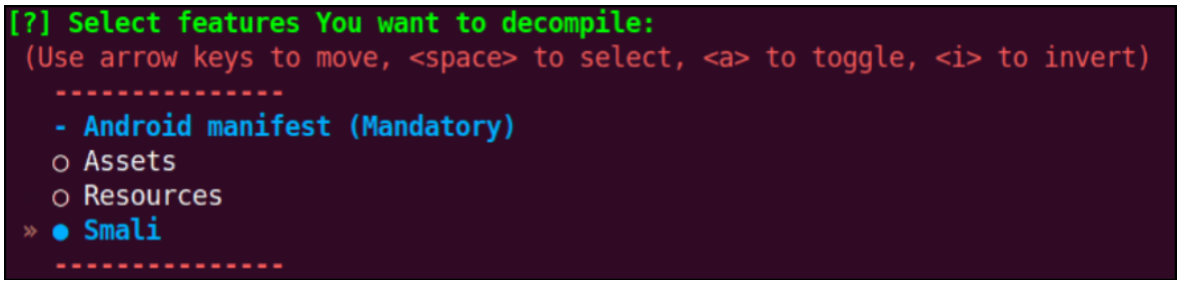


Obrázek 6.12 Předzpracovaná datová sada [zdroj vlastní]

6.3.2 Hromadná dekompilace

Dekompilační část vytvořil Bc. Jan Kincl. Dočasný adresář Apks_tmp je zásadní v celém procesu. Již zvalidované APK soubory jsou obsaženy právě v tomto adresáři, který slouží jako vstupní parametr pro hromadnou dekompilaci. Ukázalo se, že preprocessing byl velmi důležitý, protože naprostá většina balíčků, které se dostanou až do této fáze je zpracována bez větších problémů a chyb. Pro případné selhání procesu dekompilace jsou vytvořeny adresáře 02_partially_successful a 03_failed. Tyto je pak nutné manuálně prozkoumat a zjistit důvod selhání.

Ještě před zahájením preprocessingu je výzkumníkovi používajícímu A Perfect Knife zobrazeno uživatelské menu zobrazené na obrázku 6.13. Tato nabídka umožní analytikovi přizpůsobit výstup přesným výzkumným záměrům. Je díky tomu možné ověřit dekompilace (získání Smali, z dex souboru v APK balíčku), nebo pouze dekódovat soubory AndroidManifest.xml do čitelné podoby. Tento krok je nejen optimalizační, jelikož může ušetřit velké množství času, ale je také prakticky využitelný a usnadňuje i manuální analýzu. V situaci, kdy je potřeba analyzovat manifest několika aplikací, není nutné každou zvláště zpracovávat, ale je možné použít nástroj A Perfect Knife a následně manuálně analyzovat jeho výstupy. Jediným povinným souborem pro zpracování je AndroidManifest.xml, to vychází z nástroje Apktool. Dále může analytik zvolit zpracování adresáře Assets, dekompilaci resources.arsc a Smali.

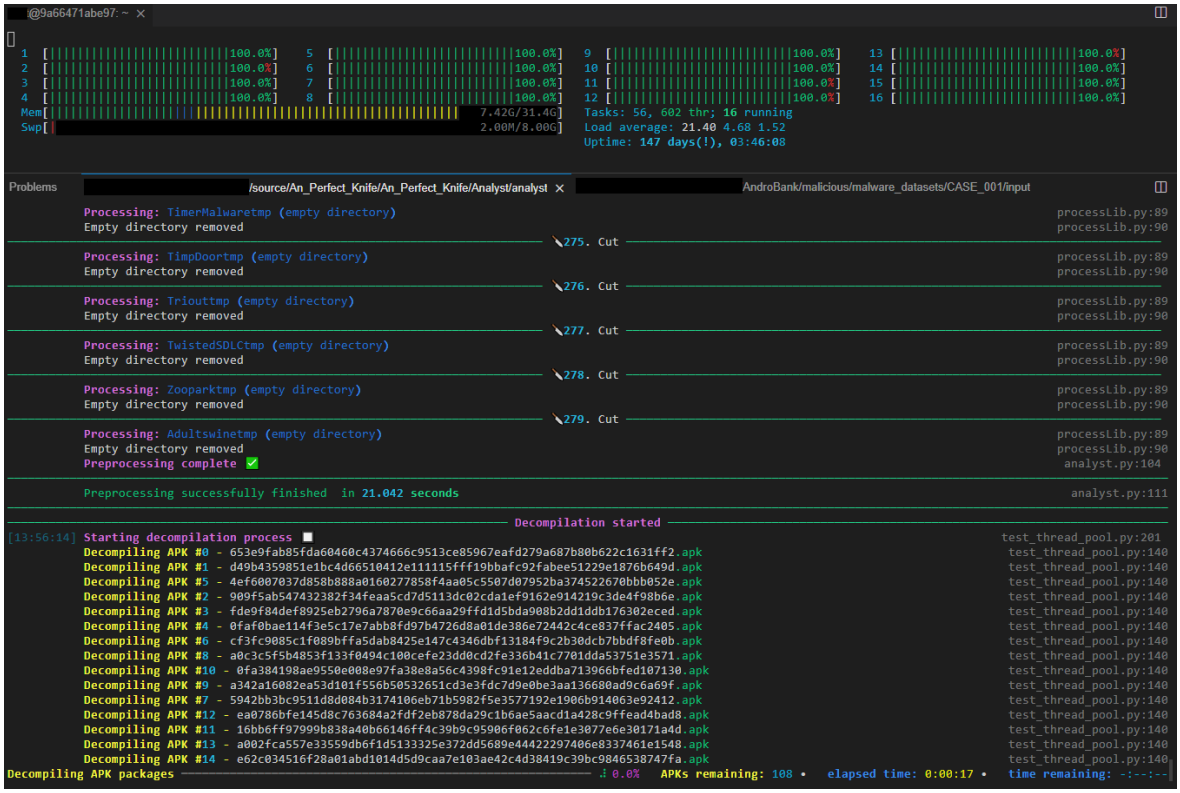


Obrázek 6.13 User menu [zdroj vlastní]

Na základě výběru uživatele je pak vytvořen vektor logických hodnot, které přesně specifikují, jaké výstupní soubory, nebo adresáře jsou požadovány, případně jaké jsou nároky na dekompilaci. Jádrem nástroje je Apktool, který byl popsán v teoretické části. Poté co je každá aplikace zpracována nástrojem Apktool, jsou validovány výsledky dle definovaného vektoru. Kontrola probíhá testováním Hammingovy vzdálenosti, na základě výsledku kontroly, jsou pak všechny zpracované soubory přesunuty do korespondujících podadresářů vytvořených v rámci tvorby pracovního prostředí.

Celý tento proces je navíc optimalizován pro paralelní běh v serverovém prostředí, což znamená, že využívá všech dostupných vláken, kromě čtyř, které jsou ponechány pro korektní běh systému. Počet nevyužitých vláken bylo nutné upravit na základě

praktických zkušeností v rámci údržby, jelikož původní vynechání dvou vláken způsobovalo nedostupnost služby vzdáleného přístupu.



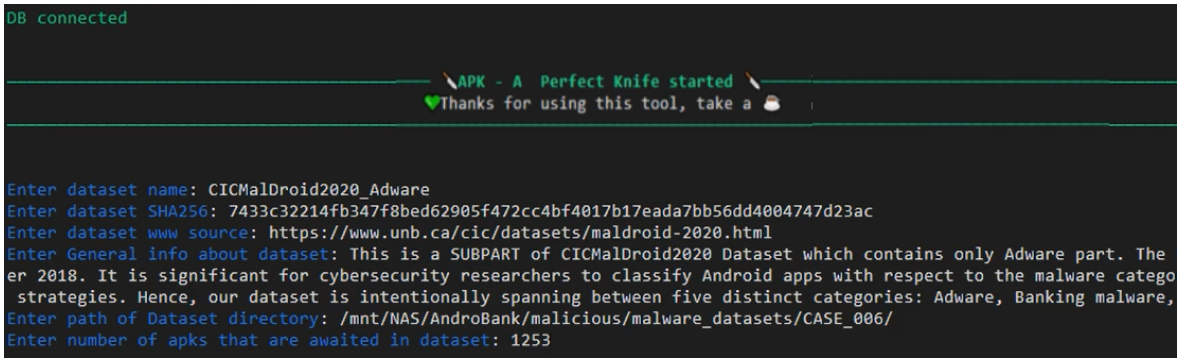
Obrázek 6.14 Ukázka běhu [zdroj vlastní]

Díky paralelizaci je pak celý dekompilační proces velmi zrychlen, ukázka využití prostředků serveru a běhu programu je zobrazena na obrázku 6.14, kde je v horní části ukázáno využití vláken serveru. Zobrazení využití prostředků serveru je realizováno zabudovaným nástrojem htop. V prostřední části je patrné dokončování preprocessingu, kde jsou mazány prázdné adresáře. V poslední, dolní, části obrázku je informace o celkovém průběhu dekompilace a aktuálně zpracovávaných balíčcích.

6.3.3 Integrace v systému AndroBank

Pro integraci do navrženého systému bylo potřeba udělat několik úprav a také určité funkcionality přidat. Úpravy prováděné v programu měly zejména funkční vliv, na příklad se jednalo o opravu bugu při zpracovávání assetů.

Přidané funkcionality jsou databázového charakteru. Nejprve byla potřeba zpětná analýza počtu zpracovaných aplikací tak, aby korespondovala s celkovým počtem validních APK balíčků, to je kontrola integrity dat v datové sadě, před zavedením do databáze. Dále byly přidány výzvy k doplnění obecných dat o datové sadě, které není možné z uložené datové sady zjistit, typicky počet očekávaných APK balíčků (uváděný vydavatelem), URL adresa zdroje a další (viz obrázek 6.15).



Obrázek 6.15 Integrace APKnife [zdroj vlastní]

Pokud celý běh nástroje A Perfect Knife proběhne korektně a kontroly jsou úspěšné, jsou přidány do databáze jednotlivé balíčky mobilního malwaru s jejich základními informacemi, jako: původ vzorku, název, cesta, velikost a další. Tyto vzorky mají garantovanou kvalitu, jedná se o validní APK soubory na nichž lze provádět další analýzu, to znamená, že takto zpracované vzorky umožňují následnou výzkumnou činnost.

6.4 Integrace VirusTotal

Nástroj VirusTotal má v systému dvě využití, jednak je to získaní analytických informací o balíčku, což bude popsáno v další sekci, jednak umožňuje ověřit integritu vzorku ve smyslu jeho malignity. Tato služba na základě API dotazu umožňuje oskenovat vstupní soubor předními AV společnostmi. To znamená, že je možné u vzorků malwaru ověřit, zda se opravdu jedná o malware a to automatizovanou formou. Tato metoda má svá omezení. Zejména nově zachycené vzorky malwaru nemusí být vyhodnoceny jako maligní, jelikož nejsou zavedeny v databázích společností. Pokud je takový nový vzorek zachycen, může se stát, že bude vyhodnocen pouze jedním, nebo dokonce žádným AV řešením. Na tyto vzorky je nutné reagovat i v rámci tvorby datových sad. V případě, že by se jednalo o falešně pozitivní vzorek, mělo by to za následek zkreslení výsledků navazujících výzkumu. Proto je důležité vzorky jednotlivě validovat i v tomto ohledu. Každý analyzovaný vzorek službou VirusTotal je zařazen do třídy příslušnosti, které jsou definovány tři: undetected, suspicious, detected.

Kategorie undetected nezaznamenala ani jednu detekci, pakliže je takový vzorek v datové sadě malwaru, je to špatně. I pokud by se teoreticky jednalo o malware, který však není detekován žádnou AV společností, je na zvážení vyřazení vzorku ze sady, alespoň do doby, než bude vzorek detekován, či manuálně analyzován.

Detekovaný vzorek - detected, znamená vyhodnocení souboru jako malware a to více, než 3 AV řešeními. Tyto vzorky jsou v datové sadě mobilního malwaru považovány za správné a žádoucí.

Poslední třídou příslušnosti je suspicious, u těchto balíčků je vysoký předpoklad, že se jedná o malware, avšak může se jednat o vzorek falešně pozitivní. Proto jsou suspicious vzorky označeny také jako "významná detekce", která není úplně jednoznačná. Byla vytvořena speciální třída příslušnosti, což umožňuje brát v potaz v dalších výzkumech i tyto hraniční stavy, které mohou mít potenciální vliv na kvalitu výzkumu.

Jedním ze zmíněných limitů služby VT je u veřejné API omezení na 500 požadavků denně. Skript, jenž bývá denně spouštěn má na starosti i režii skenování. Vzorky zpracované nástrojem A Perfect Knife jsou uloženy na NAS úložišti a jejich informace v databázi. Každý den je načteno 500 vzorků, které jsou v databázi, ale ještě nebyly oskenovány. Tyto vyhodnocené vzorky a veškeré informace jsou opět logovány do terminálu (obrázek 6.16), což umožňuje přesměrování výstupů do rutinních logů. Kromě zaznamenávání provozu do logů, jsou veškeré zjištěné informace ukládány i do databáze.

```
1. Package: com_apm2studio_chatgame.apk
SHA256: 9697d84016a32dcb2fea5c27c8b2754afaa8f19be8a801ad48ffc9748dc3b53b
VT scan success: True, Own decision: undetected
Record added:
ID: 256; SHA: 9697d84016a32dcb2fea5c27c8b2754afaa8f19be8a801ad48ffc9748dc3b53b; Scan Result: True

2. Package: com_zhangyue_read_storyaholic.apk
SHA256: 469293d5ae60da636b67867053fa98123fbd0d917e435492266e15a41d750cf0
FILE WASNT CHECKED BY VT PREVIUSLY
SHA 256: 469293d5ae60da636b67867053fa98123fbd0d917e435492266e15a41d750cf0
Path: /[redacted]/AndroBank/db_integrate_test/01_apks/com_zhangyue_read_storyaholic.apk
Scanning file.. This action may take some time
```

Obrázek 6.16 VirusTotal Log [zdroj vlastní]

Z obrázku 6.16 je evidentní, že se liší implementace pro vzorky, které jsou skenovány poprvé. Pakliže již v historii byl vzorek oskenován je možné dotazovat se na vzorky na základě jejich SHA256 otisku. Toto nelze u prvně skenovaných vzorků. Jednak by sken nemohl proběhnout, protože z otisku není možné žádným způsobem získat původní APK a jednak služba sbírá neznámé soubory pro další dotazy výzkumných subjektů. To je patrné, když se podaří nalézt vzorek, jenž nebyl službou dříve zaznamenán - výzkumník je vyzván k vložení kompletního vstupního souboru, místo pouhého otisku. Profesionální datové sady nejsou běžně dostupné a pokud by byly poskytnuty výzkumnému týmu s velkou pravděpodobností by ze strany poskytovatele existoval požadavek na to, aby vzorky z datové sady nebyly dále distribuovány. Uvedená možnost skenování pouhých otisků může být řešením, jak soubory ověřit a zároveň zabránit redistribuci jedinečných APK balíčků.

6.5 Extrakce významných charakteristik

Na datové sadě, která byla zpracována nástrojem A Perfect Knife je možné provádět další výzkumnou činnost. Jako demonstrace byla provedena extrakce významných charakteristik mobilního malwaru. Tyto charakteristiky byly zjišťovány nástroji VirusTotal a Androguard, kdy VT zprostředkovává analytická data, jako datum první modifikace a další, na proti tomu Androguard získává informace automatizovanou statickou analýzou. Interně je tento nástroj nazýván APEx (Android Package Extractor) a zjednodušeně se jedná o nástroj, který logickým zpracováním extrahuje analytická data.

6.5.1 VirusTotal

První část extrakce charakteristik se neprovádí statickou analýzu - nezískává informace ze zdrojového kódu, nýbrž kontextuální data, jako například počet různých souborů v balíčku a další. Tento krok je prováděn v rámci aplikační logiky zároveň s detekcí vzorku, to znamená, že každý vzorek, jenž je oskenován je zároveň analyzován. Informace se mohou jevit jako méně důležité, než třeba oprávnění aplikace. To je do jisté míry pravda. Pokud výzkum pracuje s již hotovou datovou sadou garantované kvality, která byla korektně zpracována, tak tyto informace nemají příliš velký dopad. Nicméně při tvorbě datových sad tyto informace hrají klíčovou roli. Zejména pak zařazení vzorku do časového intervalu působnosti zajišťuje, že datová sada dostatečně reprezentuje moderní mobilní malware.

Například datum první a poslední modifikace může naznačit datum vzniku vzorku. Uvedené informace mohou být modifikovány i tvůrci malwaru, takže nemusí být relevantní. Ve výzkumech používajících tuto metriku k odhalení časového intervalu působnosti se mohou objevit data, která již z principu nedávají smysl. Jedná se zejména o vzorky modifikované před vznikem Android OS a nebo takové, jenž mají datum modifikace vyšší, než je datum aktuální. Toto tvrzení bylo potvrzeno při sestavení APK a AAB balíčků v teoretické části. V potaz se tak berou data od roku 2008, kdy je datován vznik Android OS. Pokud by měly být vzorky zároveň dostatečně reprezentativní, pak by časový horizont měl být alespoň od roku 2015, kdy byl uveden Android 6.0, který již disponuje Android Runtime a také runtime oprávněními. Například aplikace využitá v teoretické části při popisu aktivit - ESET Mobile Security & Antivirus, vyžaduje alespoň Android 5.0 pro instalaci na zařízení. To odpovídá minimálnímu datu v roce 2014 a API 21. Tyto informace však slouží spíše jako doplňkové vzhledem k možnosti jejich úpravy. Ne všechna data získaná pomocí VirusTotal jsou ale modifikovatelná a tak nabízí praktické využití i ve výzkumech. Důležité informace vytěžené službou VirusTotal jsou popsány níže.

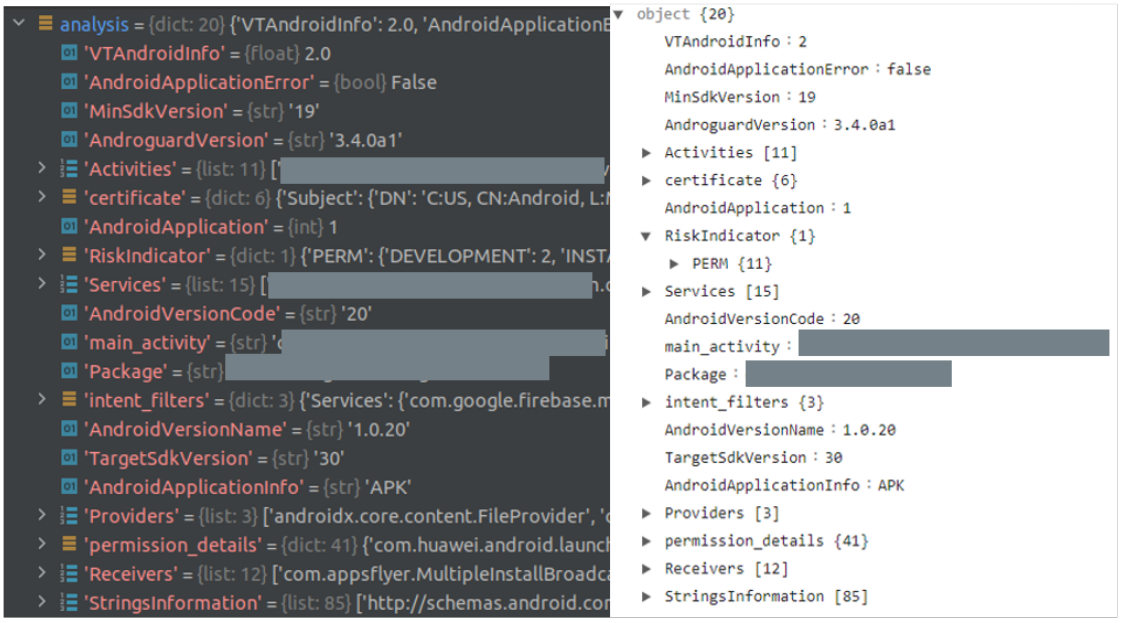
- Yara rules
  - Jedná se o proměnné definující pravidla, jenž obsahují vzory malwaru.
- Bundle info
  - Obsahuje informace o balíčku, jako: první a poslední datum modifikace, počet potomků v balíčku, nalezené typy souborů a jejich frekvenci výskytů, typ balíčku a jeho velikost.
- Popular threat classification
  - Často hodnota null, pokud je však k dispozici, nabízí doporučený label pro malware a také počet výskytů daného názvu napříč AV řešeními. Toto do jisté míry reaguje na problém zmiňovaný v teoretické části a sice nevhodnost nástroje pro tvorbu labelingu.
- Počet provedených skenů
  - Tato informace nemá velký dopad na samotnou detekci, avšak může poskytnout přehled o tom, jak frekventované vzorky jsou v datové sadě obsaženy. Zejména u znovupoužívaných datových sad je předpoklad vysokého počtu provedených skenů.
- TrID
  - Jedná se o speciální nástroj, který na základě binárních signatur souborů, identifikuje typ aktuálně zpracovaného souboru a seřadí všechny možnosti dle jejich pravděpodobností, což je prezentováno na obrázku 6.17. Tento nástroj je jeden z mála postupů, který dokáže relativně přesně odhalit APK soubor. Ani tento přístup však negarantuje jistotu správného výsledku. APK balíčky bývají často označovány jako Java Archive, což není špatně, nicméně může to být matoucí při zpracování malwaru pro více platforem.


```
[
    {'file_type': 'Java Archive', 'probability': 30.6},
    {'file_type': 'Sweet Home 3D design (generic)', 'probability': 23.8},
    {'file_type': 'Mozilla Firefox browser extension', 'probability': 18.1},
    {'file_type': 'Mozilla Archive Format (gen)', 'probability': 15.9},
    {'file_type': 'ZIP compressed archive', 'probability': 9.0}
]
```


Obrázek 6.17 VirusTotal TrID [zdroj vlastní]

- Tags
  - Tagy slouží ve službě VT zejména pro vyhledávání, aplikace může mít tagů několik, například “apk”, “reflection” a nebo třeba “obfuscated”.
- Magic
  - Unixový nástroj pro odhadnutí typu souboru z tzv. magic numbers, velmi často jsou APK balíčky označovány následujícími typy: “Zip archive data, at least v2.0 to extract”, “Zip archive data”. Toto nelze považovat za chybu, jelikož APK balíček je v podstatě ZIP, nicméně pro analýzu to není příliš směrodatné, zejména ve chvíli, kdy je problém s automatizovaným zpracováním ZIP archivu vyřešen v předešlém kroku v rámci nástroje A Perfect Knife.
- Další
  - MD5,
  - popis typu (velmi spolehlivé, u APK balíčků označeno jako “Android”),
  - meaningful name - jména, pod kterými byla aplikace původně distribuována.

6.5.2 Androguard



Obrázek 6.18 Androguard: Python object vs. JSON [zdroj vlastní]

Nástroj Androguard je zodpovědný za vytěžení staticky extrahovaných informací. Podobně jako předešlá extrakce analytických dat, je tento proces prováděn s aplikací,

jenž byla oskenována. Androguard je integrován v nástroji VirusTotal a do databáze je kompletní analýza vložena ve formátu Python objektu. Tento formát je praktický zejména při analýze v jazyce Python, zároveň jeho konverze do formátu JSON je velmi jednoduchá operace, která zajistí nezávislost na zvolené technologii pro analýzu. Porovnání těchto formátu je zobrazeno na obrázku 6.18, kde v levé části je původní Python object a v pravé části je ve formátu JSON, oba pro stejnou aplikaci.

Celý objekt zároveň má relativně komplexní, ale hlavně měnící se strukturou, což znamená, že by muselo být ošetřeno až několik desítek vstupů. Samo ošetření vstupů nepředstavuje významný problém v implementaci, nýbrž v následné výzkumné činnosti. Pakliže je zpracována aplikace nemající žádnou službu, pak tento atribut v objektu chybí. Výzkumník je tak schopen chybějící pole nahradit hodnotou NULL, nebo doplnit na základě průměrných hodnot. Způsoby korekce se liší podle typu dat a výzkumu. Těmto korekcím se přezdívá čištění dat a výzkumníci se mohou setkat s následujícími typy chyb: chybná data, chybějící data a nekonzistentní data. Při práci s datovými sadami jsou tyto zásahy do sady poměrně zásadní. Je žádoucí aby korekce proběhla až na straně subjektu, který sadu zkoumá a to pokud možno co nejšetrněji. Pokud by byla provedena korekce dat například doplněním dle průměrných hodnot, výzkumný subjekt pracující s výslednou datovou sadu by tuto informaci mohl ztratit.

Proto je extrahován kompletní Androguard report, který je jednoduše zpracovatelný a zároveň poskytuje dalším subjektům pracujícím se sadou originální výstup. Před provedením extrakce významných charakteristik mobilního malwaru byla k dispozici datová sada mobilního malwaru garantované kvality, s ověřenou integritou dat. Po provedení extrakcí je navíc k sadě dostupná množina významných charakteristik mobilního malwaru. Tato výsledná sada spolu s množinou charakteristik umožňuje navazující výzkumnou činnost a aplikaci metod umělé inteligence pro získání nových, nezjištěných poznatků ze zpracovaných vzorků vybraných datových sad.

7 VYHODNOCENÍ ZPRACOVANÝCH DATOVÝCH SAD

Výstupem jsou zpracované datové sady v rámci diplomové práce. Z bezpečnostních důvodů nejsou sdíleny přímo APK balíčky obsahující živý mobilní malware, místo celého balíčku bude zahrnut SHA256 otisk jednotlivých zpracovaných vzorků, který představuje jednoznačný identifikátor balíčku. Ke každému vzorku z výsledné datové sady je přiřazen výsledek skenování. Toto zahrnuje jednak detailní informace o detekci antivirovými společnostmi, jednak zmíněné třídy příslušnosti: detected, suspicious, undetected. Zároveň s tímto jsou ke každému balíčku přiřazeny informace získané statickou analýzou s využitím nástroje Androguard.

Součástí výstupů práce je také analýza datových sad vybraných ke zpracování, které se věnuje následující část práce. Případné problémy v datových sadách nemají za cíl snížit jejich důvěryhodnost, ale spíše poukázat na jednotlivé problémy. Je nutné upozornit, že vytvoření všech zmíněných sad je značně problematický úkol a může se stát, že do sady jsou zařazeny nevalidní soubory. Není také vyloučena přítomnost malwaru na jiné platformy a to zejména v situaci, kdy se tvůrce kolekce nezabývá pouze analýzou mobilního malwaru. Zmíněná tvrzení jsou klíčová, jelikož cílem této práce není diskreditovat tvůrce jednotlivých sad, naopak tvůrcům datových sad patří uznání a poděkování, jelikož pro komunitu zabývající se kybernetickou bezpečností představují tyto kolekce důležitý pramen informací.

7.1 Analýza datových sad

Analýza datových sad se zaměřuje na zjištění, zda se v sadách vyskytují nežádoucí jevy a jaké to jsou. Dále je zkoumán reálný počet balíčků vůči očekávanému a s tím také spojené statistiky dekompilací, jež jsou úzce spjaty s preprocessingem datových sad. Vyšlo najevo, že naprostou většinu aplikací, které byly vyhodnoceny jako validní APK balíčky, bylo bez problémů dekompilovat. Součástí analýzy je také ověření integrity datové sady ve smyslu její malignity, tedy zda se v datových sadách mobilního malwaru vyskytuje opravdu jen malware, nebo je tomu jinak. Poslední významnou oblastí podléhající analýze je dostatečná reprezentativnost vzorků v kontextu moderního mobilního malwaru.

V rámci ověření časového intervalu, ve kterém malware působí je možné využít informace získané pomocí VirusTotal a sice první a poslední datum modifikace. Již bylo zmíněno, že tento postup může být problematický a nemusí umožnit informační zisk ze všech zkoumaných subjektů. Proto byl využit druhý přístup, který je podložen daty získanými statickou analýzou, přesněji daty z AndroidManifest.xml. V rámci popisu manifestu bylo zmíněno, že obsahuje: MinSdkVersion a TargetSdkVersion. První zmíněný atribut, tedy minimální verze SDK (software development kit), je při tvorbě

aplikací pro Android automaticky vyplněn, na základě verze OS, která byla zvolena při tvorbě projektu. Nástroj Androguard poté ze staticky získaných informací určí minimální verzi sdk, které je přímo odvozena z manifestu, ale také TargetSdkVersion - tedy verzi na které byla aplikace testována a nevykazuje žádné problémy. V roce 2022 je minimální zvolitelná verze API 16, což odpovídá Androidu verze 4.1, takové vzorky nejsou dostatečně reprezentativní.

Z verzí SDK není sice možné určit den, nebo měsíc vzniku aplikace. To je možné z dat modifikací, nicméně cílová verze sdk je mnohem více vypovídající o tom, na jaká zařízení aplikace cílí. Z data modifikace, které bylo automaticky po sestavení aplikace nastaveno na dobu dávno před vznikem Android OS není vůbec jasné na jaká zařízení cílí. Je tak možné, že data modifikací na kterých některé výzkumy staví nejsou až tak relevantní jako verze SDK, navzdory nižší přesnosti mají potenciál zařazení aplikace do časového intervalu působnosti. Kdy minimální verze SDK je považována za začátek intervalu, včetně, a cílová verze představuje konec intervalu, také včetně.

Pakliže by se mělo teoretické minimum pro dostatečně reprezentativní vzorek určit, je vhodná inspirace aplikací ESET, která jak již bylo zmíněno podporuje zařízení s Android OS 5.0+ a API 21 a vyšší. Teoreticky nejreprezentativnější by však byly vzorky pro Android 6.0+ a API 23 a vyšší.

7.2 AndroidMalware - 2018

První zpracovanou datovou sadou je veřejná kolekce AndroidMalware - 2018. Vzorky v této datové sadě byly pozorovány v roce 2018, jak uvádí její autor. Důležité charakteristiky jsou shrnuty v tabulce 7.1. Všechny validní APK balíčky nalezené v sadě, byly kompletně a úspěšně zpracovány. To znamená, že se podařilo všechny vzorky staticky analyzovat, dekompilovat a ověřit jejich integritu ve smyslu škodlivosti.

Bylo odhaleno pět nevalidních souborů a vytěženo 108 validních APK balíčků. Všechny vzorky v datové sadě mobilního malwaru byly zařazeny do třídy příslušnosti "detected". Data o cílových verzích naznačují, že obsahuje vzorky aktuální. Nejnižší MinSdkVersion je sice pouze osm, avšak maximální cílová verze SDK je rovna 27. To znamená, že sada obsahuje vzorky, jenž splňují podmínku na minimální SDK rovno 23. V rámci výsledné datové sady jsou zahrnuty všechny vzorky, pro budoucí výzkum se však nabízí filtrace vzorků, splňujících podmínku pro dostatečnou reprezentativnost.

Celkový počet APK balíčků neodpovídá očekávanému počtu, což nepředstavuje zásadní problém. Na rozdíl od nevalidních souborů, kterých bylo nalezeno rovných pět, jak je vidět z obrázku 7.1. Právě defekty v podobě nevalidních souborů představují hlavní problém na jinak velmi kvalitní sadě. Je nutné zdůraznit, že validní vzorky vykazují vysokou kvalitu. Například maximální cílová verze SDK silně převyšuje maxi-

AndroidMalware - 2018			
SHA256: 9d4c1246d71a62aac518b568b7d9dec472fe16f0756a088deaf03d0a11d5e606			
Očekávaný počet APK		134	
Reálný počet APK		108	
Z toho dekompilace		Z toho skenování	
Úspěšné	108	Detected	108
Částečně úspěšné	0	Suspicious	0
Selhání	0	Undetected	0
Z toho SDK Info			
Minimum: MinSdkVersion Maximum: MinSdkVersion		8 26	
Minimum: TargetSdkVersion Maximum: TargetSdkVersion		14 27	
Defekty v datové sadě			
Duplicity		0	
Nekompletní APK		0	
Nevalidní soubory		5	
Neúspěšná statická analýza		0	

Tabulka 7.1 Charakteristiky datové sady 1

mální hodnoty nalezené v rámci některých částí akademické datové sady CICMalDroid 2020. Přesněji převyšuje všechny části akademické sady až na kategorii Riskware, kde jsou hodnoty maximálních SDK stejné.

Faktem je, že v sadě byl nalezen i malware v jiném formátu, než APK. Není vyloučena spustitelnost na zařízení s Android OS. Prvotní ohledání ukázalo, že se jedná o dva shell scripty a dva soubory typu ELF (Executable and Linkable Format), což je spustitelný soubor využívaný například v Linux OS. Zda se jedná o záměr je diskutabilní, účelem práce není snížit důvěryhodnost sady, nýbrž poukázat na tyto skutečnosti. V rámci této práce slouží soubory především pro zvýraznění nástroje A Perfect Knife, jenž v rámci preprocessingu tyto soubory odhalil. Pátým nevalidním souborem je readme.md, obsahující popis sady.

Celkově datová sada vykazuje vysokou kvalitu, vhodná by byla hlubší analýza nevalidních souborů. Maximální verze SDK pro rok 2018 byla rovna 28, maximum obsažené v této sadě je 27. Takové vzorky lze považovat za reprezentativní. Analýza charakteristik mobilního malwaru navíc často potvrzuje, že datum prvního skenování službou VirusTotal proběhlo v roce 2018. Uvedené skutečnosti naznačují, že tvůrci malwaru nejsou primárně zaměřeni jen na nejnovější verze, ale často cílí na verze starší.

Obrázek 7.1 Výstup A Perfect Knife pro datovou sadu Android Malware 2018 [zdroj vlastní]

7.3 AndroidMalware - 2019

Kolekce AndroidMalware - 2019 vykazuje podobnou kvalitu, jako předchozí sada. Celkově jsou charakteristiky velmi podobné. Opět by byla vhodná filtrace vzorků, které reprezentují minima v datové sadě, pokud by měla být splněna podmínka pro dostatečnou reprezentativnost. Minima SDK verzí jsou u datové sady AndroidMalware - 2019 nižší, než u jejího předchůdce. Maximální SDK v roce 2019 se rovnalo 29, to odpovídá i maximu v cílovému SDK, které bylo zjištěno v rámci tvorby charakteristik datové sady a je prezentováno v tabulce 7.2. V rámci charakteristik mobilního malwaru v datové sadě bylo také potvrzeno, že vzorky byly poprvé skenovány službou VirusTotal právě v roce 2019. Uvedené skutečnosti opět potvrzují, že vzorky byly nejen sbírány v roce 2019, ale také byly v tomto roce pozorovány, jako aktivní. To naznačuje, že tvůrci mobilního malwaru se sice zaměřovali na nejnovější verze Android OS, stejně tak však malware cílil i na zařízení s již ukončenou oficiální podporou.

Společné rysy obou sad jsou podobné cílové SDK verze, ačkoliv sada pro rok 2019 pokrývá širší spektrum API. Další sdílené charakteristiky jsou nižší reálný počet APK souborů, než ten udávaný, ale také stoprocentní úspěšnost při dekompilacích a skenování. Výsledky skenování ověřily integritu dat ve smyslu jejich škodlivosti, kdy všechny vzorky byly opět zařazeny do třídy příslušnosti: "detected".

Odlišnosti byly pozorovány zejména při preprocessingu, kdy bylo odhaleno několik defektů v datové sadě. Přesněji se jednalo o tři duplicity, které byly z datové sady odstraněny a jeden nekompletní APK balíček. Nekompletní APK balíček byl manuálně ohledán, kdy analýza dat získaných ze služby VirusTotal naznačuje, že se jedná o spusti-

telný soubor na operační systém Windows, navíc byl soubor vyhodnocen jako škodlivý, proto by byla vhodná hlubší analýza. Není obvyklé, že soubory typu PE32 jsou zařazeny do kategorie nekompletních APK balíčků, právě hlubší analýza souboru by měla odkrýt důvod tohoto rozřazení. Předmětem práce však nebylo analyzovat všechny nevalidní soubory, nýbrž jejich vyloučení z výsledné datové sady, tak aby výsledná datová sada obsahovala pouze validní APK balíčky. Vyloučením spustitelného souboru na operační systém Windows je tak zamezeno v dalším nechtěném šíření tohoto souboru. Vyloučení souboru také zamezuje případnému zkreslení výsledků navazujících výzkumů. Hlubší zkoumání tohoto souboru vyžaduje sadu dovedností, která se liší od tradiční sady dovedností analytiků mobilního malwaru, proto by v ideálním scénáři měla proběhnout analýza těchto souborů za spolupráce s odborníky na malware zacílený na operační systém Windows.

AndroidMalware - 2019			
SHA256: 28baea92a6830e414419ac8fac6d3be308234dccd8b46b8075d1f1c0929363c5			
Očekávaný počet APK		186	
Reálný počet APK		155	
Z toho dekompilace		Z toho skenování	
Úspěšné	155	Detected	155
Částečně úspěšné	0	Suspicious	0
Selhání	0	Undetected	0
Z toho SDK Info			
Minimum: MinSdkVersion		5	
Maximum: MinSdkVersion		24	
Minimum: TargetSdkVersion		11	
Maximum: TargetSdkVersion		29	
Defekty v datové sadě			
Duplicity		3	
Nekompletní APK		1	
Nevalidní soubory		10	
Neúspěšná statická analýza		1	

Tabulka 7.2 Charakteristiky datové sady 2

Další diferencí mezi sadami je také počet nevalidních souborů, kterých bylo nalezeno rovných deset. Kromě tradičního souboru readme.md (popisující datovou sadu), byly v sadě objeveny i další nevalidní soubory. Ostatních devět souborů bylo ohledáno s využitím služby VirusTotal, ačkoliv se často jedná o malware zacílený na operační systém Windows, vzorky jsou úzce spjaty s rodinami, které cílí na Android OS. Rodiny Tripoli

a Farseer totiž dle komunity na platformě VirusTotal, která čerpala informace zejména z analýz organizací zabývajících se touto problematikou, cílili jak na operační systém Android, tak na Windows. Celkem bylo nalezeno pět souborů typu PE32 (spustitelný soubor pro operační systém Windows), se třemi zástupci rodiny Farseer a dvěma zástupci rodiny Tripoli. Třetí zástupce rodiny Tripoli je pak ve formátu RAR a dle služby VirusTotal obsahuje jednu, nebo více spustitelných souborů pro Windows. Poslední tři nevalidní soubory jsou zástupci malwaru CryptoMining Botnet. Dva vzorky byly ve formátu textového dokumentu a třetí ve formátu shell skriptu. Jako u všech nevalidních souborů by zde byla vhodná hlubší analýza, nicméně důležité pro výslednou datovou sadu je vyloučení právě těchto formátů.

Kromě preprocessingu se vyskytl jeden problém i při statické analýze, kdy jeden ze vzorků nebyl úspěšně analyzován nástrojem VirusTotal. Neúspěch této statické analýzy byl podrobně prozkoumán a bude reflektován do systému v rámci budoucí údržby, jež je plánována po dokončení diplomové práce laboratoří PTLAB. Účelem práce totiž bylo, kromě tvorby výsledné datové sady také nalezení problematických míst v systému tak, aby byly tyto části identifikovány a mohly být zavedeny do dalších verzí systému.

7.4 AndroidMalware - 2020

Android Malware - 2020 je datová sada velmi podobná těm výše zmíněným. Tabulka 7.3 prezentuje charakteristiky kolekce, které se nejvíce liší nalezením podezřelých vzorků. Kromě 190 APK balíčků, jenž jsou zařazeny do třídy “detected”, byly zaznamenány dvě významné detekce, které jsou přiřazeny k třídě “suspicoius”. Statická analýza byla úspěšně provedena u 191 ze 192 vzorků.

Maximální verze SDK pro rok 2020 byla rovna 30, což odpovídá i maximu nalezenému v datové sadě. Stejně jako předchozí datové sady i tato obsahuje vzorky s nízkou minimální cílovou verzí SDK. Tyto vzorky jsou ve výsledné datové sadě ponechány, avšak do navazujících výzkumů se nabízí jejich další filtrace pro splnění reprezentativní podmínky.

V sadě bylo nalezeno sedm defektů, jmenovitě se jedná o odstraněnou duplicitu, dva nekompletní balíčky a čtyři nevalidní soubory.

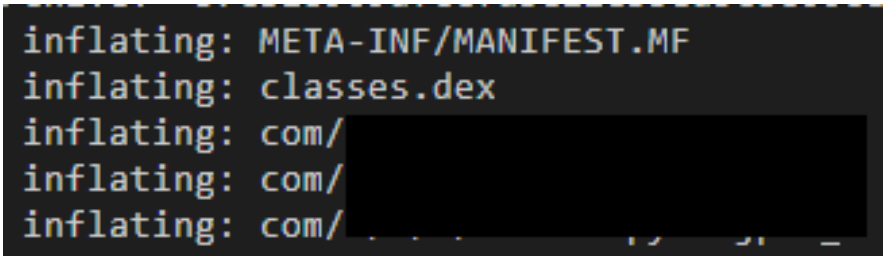
Oba nekompletní balíčky byly správně zařazeny, jelikož vzorky obsahovaly pouze některé z povinných souborů (viz. obrázek 7.2), chybějící soubor je například “AndroidManifest.xml”, formát obou souborů pak byl službou VirusTotal označen jako Java Archive a oba byly vyhodnoceny jako škodlivé. Ačkoliv je jejich praktická využitelnost teoreticky možná, cílem bylo eliminovat nekompletní balíčky, což se podařilo.

Nevalidní soubory jsou tvořeny jedním readme souborem, jedním souborem označeným službou VirusTotal jako spyware pro operační systém Windows a dvěma ELF

AndroidMalware - 2020			
SHA256: 2375a35831ba9781f88d474e3f5cb6bca23f7f15232179a9652fc464a401c91b			
Očekávaný počet APK		218	
Reálný počet APK		192	
Z toho dekompilace		Z toho skenování	
Úspěšné	192	Detected	190
Částečně úspěšné	0	Suspicious	2
Selhání	0	Undetected	0
Z toho SDK Info			
Minimum: MinSdkVersion Maximum: MinSdkVersion		4 24	
Minimum: TargetSdkVersion Maximum: TargetSdkVersion		14 30	
Defekty v datové sadě			
Duplicity		1	
Nekompletní APK		2	
Nevalidní soubory		4	
Neúspěšná statická analýza		1	

Tabulka 7.3 Charakteristiky datové sady 3

soubory, jenž jsou prokazatelně zaměřeny na Android OS a zastupují rodinu, nejčastěji označovanou antivirovými programy jako "Android . Xiny", což potvrzuje i komunita na platformě VirusTotal. Správně tak byly ze systému tyto soubory vyřazeny, ačkoliv v rámci navazujících výzkumů představuje malware na Android OS ve formátu ELF zajímavý subjekt ke zkoumání. Je však nutné upozornit, že automatizované zpracování těchto souborů na operačním systému Linux přináší řadu dalších úskalí, například napadení výzkumného systému. V rámci autonomního zpracování datových sad je bezpečnější tyto soubory vyloučit a následně manuálně analyzovat.



Obrázek 7.2 Nekompletní balíček [zdroj vlastní]

7.5 AndroidMalware - 2021

Společným jmenovatelem všech Android Malware datasetů je nejednotné názvosloví, které bylo v rámci standardizace vzorků vyřešeno automatizovaně a v systému jsou všechny validní vzorky zavedeny pod názvy <SHA256>.apk, dalším společným faktorem pro všechny výše uvedené sady včetně té pro rok 2021 je nižší reálný počet APK balíčků, než bylo očekáváno.

Dataset pro rok 2021 by teoreticky mohl obsahovat maximální cílovou verzi SDK rovnu 31, nebo 32. Ve srovnání s předešlými sadami je tak překvapením, že obsahuje maximální cílovou verzi 30. Toto je demonstrováno daty v tabulce 7.4. Nepředstavuje to problém, avšak očekávání v tomto směru nebylo naplněno. Na druhou stranu minimum cílové verze SDK, jenž bylo nalezeno činí 21. Tato hodnota znamená, že celá sada je dostatečně reprezentativní a není tak nutná filtrace vzorků dle podmínky pro dostatečnou reprezentativnost. Jedná se o druhý nejlepší výsledek v rámci všech zpracovaných datových sad, kdy jediná sada, které převyšuje toto cílové SDK je Android Malware Sample Library, která je popsána níže.

AndroidMalware - 2021			
SHA256: d163c18bb15aff412b059031430b923120424b1feed17fa9a7604b9cbdcc2b1e			
Očekávaný počet APK	137		
Reálný počet APK	115		
Z toho dekompilace		Z toho skenování	
Úspěšné	115	Detected	115
Částečně úspěšné	0	Suspicious	0
Selhání	0	Undetected	0
Z toho SDK Info			
Minimum: MinSdkVersion	5		
Maximum: MinSdkVersion	27		
Minimum: TargetSdkVersion	21		
Maximum: TargetSdkVersion	30		
Defekty v datové sadě			
Duplicity	1		
Nekompletní APK	1		
Nevalidní soubory	2		
Neúspěšná statická analýza	1		

Tabulka 7.4 Charakteristiky datové sady 4

Další pozitivum na této sadě je fakt, že všechny zpracované APK balíčky byly úspěšně dekompilovány i skenovány. Skenování potvrdilo, že všechny vzorky obsažené v sadě je možné označit jako malware.

Celkový počet defektů v datové sadě je navíc velmi nízký, kdy byly vyloučeny pouze 4 soubory. Kromě jednoho duplicitního balíčku byly vyřazeny tři další soubory. Nekompletní APK balíček má stejnou charakteristiku, jako Java Archiv balíčky probírané v předchozí datové sadě. Opět se jedná o malware na Android OS, ale tento archiv byl vyloučen ze stejných důvodů, jako tomu bylo v předchozí analýze.

V rámci nevalidních souborů byl odhalen opět soubor readme.md, ale také mobilní malware pro Android OS. Malware však není součástí APK balíčku, ale jedná se o samostatný dex soubor. S nejvyšší pravděpodobností se jedná o dex, jenž je dynamicky načten legitimní aplikací, jak bylo popsáno v teoretické části. Soubor byl z datové sady odstraněn, jelikož se nejedná o validní APK balíček, jedná se však o analyzovatelný soubor implementovaným systémem, proto je v rámci plánované údržby systému naplánováno i přidání funkcionality ke zpracování právě těchto dex souborů. Do výsledné datové sady nebyl tento soubor zařazen, jelikož byl cílem zpracovat validní APK balíčky. Provedení statické analýzy pak selhalo pouze u jednoho validního APK balíčku, ostatní balíčky úspěšně prošly celým procesem zpracování a podařilo se získat charakteristiky mobilního malwaru. Na obrázku 7.3 je ukázka sady po zpracování a ověření dekompilací pomocí nástroje A Perfect Knife, který potřeboval celkem 294 řezů sadou pro kompletní zpracování.

Obrázek 7.3 Výstup A Perfect Knife pro datovou sadu Android Malware 2021 [zdroj vlastní]

7.6 Android Malware Sample Library

První datovou sadou od jiného vydavatele a zároveň poslední zpracovanou veřejnou datovou sadou je Android Malware Sample Library. Z analýzy vyplynulo, že kolekce obsahovala přesný počet APK balíčků, jenž byl specifikován vydavatelem. Žádná aplikace nebyla označena jako benigní, ale u dvou validních APK balíčků byla zaznamenána významná detekce a tedy zařazení do třídy "suspicoius". Zbylých 162 balíčků bylo vyhodnoceno jako "detected". V rámci zpracování datové sady bylo odhaleno přesně 164 APK balíčků, 27 duplicit a tři nevalidní soubory. Statickou analýzu se pak podařilo kompletně provést u 161 APK balíčků, takže pouze tři aplikace byly analyzovány s problémy. Kompletní charakteristiky datové sady jsou prezentovány tabulkou 7.5.

Android Malware Sample Library			
SHA256: a4d250ce2c67cf5aa47ad6c464363050dc11cdc40036da5dadf15d9b6ff2c4e6			
Očekávaný počet APK	164		
Reálný počet APK	164		
Z toho dekompilace		Z toho skenování	
Úspěšné	164	Detected	162
Částečně úspěšné	0	Suspicious	2
Selhání	0	Undetected	0
Z toho SDK Info			
Minimum: MinSdkVersion	5		
Maximum: MinSdkVersion	21		
Minimum: TargetSdkVersion	22		
Maximum: TargetSdkVersion	29		
Defekty v datové sadě			
Duplicity	27		
Nekompletní APK	0		
Nevalidní soubory	3		
Neúspěšná statická analýza	3		

Tabulka 7.5 Charakteristiky datové sady 5

Zajímavé se může zdát, že očekávaný počet APK je roven reálnému, když bylo nalezeno několik defektů. Balíčky byly v původní sadě umístěny ve dvou adresářích, v rámci těchto dvou adresářů se 27 balíčků překrývalo a byly označeny jako duplicitní. Duplicity byly vyloučeny z výstupní datové sady. Ohledání nevalidních souborů odkrylo dva textové dokumenty obsahující identifikátory vzorků a readme.md soubor popisující kolekci.

Celkově datová sada vykazuje vysokou kvalitu, kdy nevalidní soubory byly čistě

popisné, takže nejsou považovány za problematické, i když musely být vyloučeny z výsledné datové sady. Ke kvalitě uvedené datové sady přispívají hodnoty SDK verzí, kdy minimum v cílových verzích je rovno 22, to znamená, že celá datová sada je dostatečně reprezentativní a není nutná další filtrace vzorků. Zároveň je to nejvyšší minimální hodnota dosažená mezi všemi datovými sadami, které byly zpracovány.

7.7 CICMalDroid 2020

Datová sada je vydavatelem rozdělena do pěti kategorií, kde je možné každou zpracovat zvlášt. Tohoto bylo využito pro poskytnutí detailnějších informací o jednotlivcýh kategoriích v sadě. Vydavatel, jak bylo zmíněno v teoretické části, sbíral vzorky mezi lety 2017-2018 (SDK: 26-29). V rámci zařazení vzorků do intervalu časové působnosti se tak nabízí přímé srovnání s datovou sadou Android Malware 2018. Je samozřejmě nutné brát v potaz velikosti obou datových sad, kdy tvorba tak obrovské datové sady, jako je CICMalDroid 2020 přináší řadu problémů, jenž u menších datových sad nehrají tak významnou roli. Zejména zařazení vzorků do kategorie příslušnosti může být problematické a může se v čase měnit, to znamená že vzorky v roce 2018 mohly patřit mezi benigní, avšak v roce 2022 mohou být detekovány několika AV programy. Sami autoři k sadě uvádí, že čerpá z několika zdrojů, jenž byly používány v zejména v předešlých letech. Toto tvrzení bylo analýzou potvrzeno a je pozorovatelné zejména na nízkých minimálních hodnotách cílových SDK verzí.

Zajímavým ukazatelem v této sadě je očekávaný celkový počet APK balíčků. Očekávaný počet je přímo získán od vydavatele datové sady a jedná se o počty balíčků, jenž byly v daných kategoriích úspěšně analyzovány. Pakliže počet úspěšně staticky analyzovaných vzorků bude vyšší, než očekávaný počet APK balíčků, znamená to, že se podařilo vytěžit více aplikací samotnou statickou analýzou, než s využitím emulátoru, či reálného zařízení.

Vydavatelé datové sady uvádí, že v rámci provedené analýzy úspěšně zpracovali 9803 vzorků mobilního malwaru. V rámci této diplomové práce bylo vytěženo z datové sady CICMalDroid 2020 celkem 12687 validních APK balíčků a statická analýza, tedy získání charakteristik mobilního malwaru byla kompletně provedena u 12670 z nich, takže analýza selhala pouze u sedmnácti vzorků.

V originální datové sadě jsou zahrnuty i benigní aplikace, vzhledem k povaze této práce a jejímu cíli: tvorba datové sady mobilního malwaru, byly tyto aplikace vyloučeny z výsledné datové sady, ale také ze statistik. Porovnávaná data jsou tak získána pouze ze vzorků mobilního malwaru. Popis jednotlivých částí kolekce je dostupný níže.

7.7.1 CICMalDroid 2020: Adware

První kategorií obsaženou v kolekci je adware. Z charakteristik datové sady v tabulce 7.6 je nejvíce překvapující maximální cílová verze SDK rovna 21, což nedosahuje odpovídajících verzí 26-29, které bylo možné v letech 2017 a 2018 sbírat. Například datová Android Malware - 2018 značně převyšuje tyto hodnoty. Na této datové sadě tak není filtrace pouze vhodná, ale silně doporučená, jelikož vzorky s cílovou verzí SDK 4 neplní podmínku pro minimální reprezentativnost. Celková kvalita datové sady je tímto faktem velmi snížena. Ačkoliv všechny vzorky byly úspěšně dekompilovány i zařazeny do třídy příslušnosti, filtrace je zde nevyhnutelná. Obsoletní vzorky mohou mít značně negativní vliv na navazující výzkumnou činnost.

CICMalDroid 2020: Adware			
SHA256: 7433c32214fb347f8bed62905f472cc4bf4017b17eada7bb56dd4004747d23ac			
Očekávaný počet APK		1253	
Reálný počet APK		1513	
Z toho dekompilace		Z toho skenování	
Úspěšné	1513	Detected	1513
Částečně úspěšné	0	Suspicious	0
Selhání	0	Undetected	0
Z toho SDK Info			
Minimum: MinSdkVersion		1	
Maximum: MinSdkVersion		15	
Minimum: TargetSdkVersion		4	
Maximum: TargetSdkVersion		21	
Defekty v datové sadě			
Duplicity		0	
Nekompletní APK		0	
Nevalidní soubory		1	
Neúspěšná statická analýza		2	

Tabulka 7.6 Charakteristiky datové sady 6.1

Všechny vzorky byly zařazeny do výsledné datové sady, primárním důvodem však je zachování dat v databázi, jelikož analýza dostupných datových sad ukázala, že kolekce mohou být časem znepřístupněny. Ačkoliv část kolekce není dostatečně reprezentativní, celková kvalita zpracovaných balíčků je velmi vysoká. Důkazem je velmi nízký počet defektů v kolekci, jelikož statická analýza nebyla úspěšně provedena u dvou z 1513 balíčků a nevalidní soubor byl nalezen pouze jeden. Tímto souborem je shell skript, který je distribuován spolu s datovou sadou a slouží k přejmenování souborů, tento

není potřebný vzhledem k činnosti nástroje A Perfect Knife, který řeší jednotnost názvosloví v rámci celého systému automatizovaně.

7.7.2 CICMalDroid 2020: Banking

Druhá část datové sady je zaměřená na bankovní malware, na rozdíl od adwaru zde analýza odhalila větší množství problematických faktorů. Co se týče reprezentativnosti vzorků v sadě, tak je tato část o něco aktuálnější, dokonce obsahuje vzorky s maximální cílovou verzí SDK 25 (viz tabulka 7.7). To znamená, že vzorky v době sběru byly téměř maximálně aktuální. Na druhou stranu zde byl registrován i výskyt velmi starých a nedostatečně reprezentativních vzorků, opět je tedy žádoucí filtrace pro navazující výzkumnou činnost. Celkem statická analýza proběhla úspěšně u 2441 vzorků, takže pouze 10 vzorků nemá přiřazeny kompletní charakteristiky mobilního malwaru. Tvůrci datové sady získali charakteristiky k 2100 aplikacím, takže se tento počet analyzovaných APK balíčků dá považovat za úspěch.

CICMalDroid 2020: Banking			
SHA256: 353c9d800ad73b0048fe6d7f8649008c30b6d776c68137041c0be9fc1258ffa7			
Očekávaný počet APK	2100		
Reálný počet APK	2451		
Z toho dekompilace		Z toho skenování	
Úspěšné	2451	Detected	2419
Částečně úspěšné	0	Suspicious	7
Selhání	0	Undetected	25
Z toho SDK Info			
Minimum: MinSdkVersion	3		
Maximum: MinSdkVersion	18		
Minimum: TargetSdkVersion	4		
Maximum: TargetSdkVersion	25		
Defekty v datové sadě			
Duplicity	54		
Nekompletní APK	0		
Nevalidní soubory	1		
Neúspěšná statická analýza	10		

Tabulka 7.7 Charakteristiky datové sady 6.2

V rámci skenování souborů však byly odhaleny zásadní problémy. Celkem 25 aplikací, jež jsou v sadě přítomny, není v roce 2022 označeno žádným AV programem, jako malware. Tyto vzorky jsou zařazeny do třídy příslušnosti "undetected". Tento ob-

jev přímo vycházející z analýzy mobilního malwaru v datové sadě je zásadní. Výskyt benigních aplikací v kategorii Banking malware má negativní dopad na výsledky navazujících výzkumů. V sadě bylo také nalezeno sedm významných detekcí, které byly zařazeny do třídy příslušnosti "suspicious", zbylých 2419 aplikací pak bylo označeno správně jako mobilní malware. Zde je třeba upozornit, že při tak velkém množství aplikací lze problémy tohoto typu očekávat. Proto také byla provedena tato analýza. Výzkumy pracující s datovými sadami neprovádí tyto kontroly vždy, je ale evidentní, že je to nutné.

Defekty v datové sadě jsou v podobě duplicit, což nepředstavuje problém, jelikož byly v rámci preprocessingu z datové sady vyloučeny. Jediný nevalidní soubor v sadě pak plní stejný účel jako v minulé části a jedná se o bash skript ke změně názvů.

Datová sada vykazuje průměrnou kvalitu, kdy zmíněné problémy spojené s rozsáhlými datovými kolekcemi kompenzuje právě vysoký počet validních vzorků. Dále je již na zvážení výzkumných subjektů, zda zahrnou všechny aplikace, nebo jen ty dostatečně reprezentativní. Na obrázku 7.4 je ukázka po zpracování sady. Kompletní zpracování této části pak pro zajímavost trvalo přibližně 2 hodiny. Ekvivalentní manuální zpracování by trvalo i desítky hodin.

Obrázek 7.4 Výstup A Perfect Knife pro datovou sadu CICMalDroid 2020: Banking [zdroj vlastní]

7.7.3 CICMalDroid 2020: Riskware

Předposlední zpracovanou částí kolekce CICMalDroid 2020 je Riskware. Defekty v datové sadě představují čtyři nekompletní APK balíčky, kdy je postrádán zejména dex soubor, který je klíčový pro statickou analýzu. Tyto vzorky byly vyloučeny z výsledné datové sady.

Jak již bylo zmíněno výše, jedním z cílů práce bylo najít oblasti, kde existuje prostor pro zlepšení. V tabulce 7.8 je patrný velký počet nevalidních souborů. Lepší zařazení pro tyto soubory by pravděpodobně byla kategorie pro nekompletní APK balíčky, protože se jedná o soubory se zkompilovaným zdrojovým kódem, avšak nejsou zabaleny do APK balíčku. Vzniklá situace pramení z povahy nevalidních dat. Již bylo zmíněno, že by bylo dobré analyzovat i samotné dex soubory, protože je to proveditelné. Soubory označeny jako nevalidní v této části sady jsou zejména dex formátu. Tyto byly správně z datové sady APK balíčků vyloučeny, je ale na zvážení jejich analýza, protože se dle VirusTotal prokazatelně jedná o mobilní malware. Toto je tedy jedno z míst ve vytvořeném multiagentním systému, kde je prostor ke zlepšení. V celkovém pojetí to však opět není možné brát jako chybu, jelikož cílem tvorby datové sady byla prezence kompletních a validních APK balíčků.

CICMalDroid 2020: Riskware			
SHA256: 9b72f4f54dcf3ce3d505d828acca0fd453a0bbf795a6245c713e4f7d065e9f64			
Očekávaný počet APK		2546	
Reálný počet APK		3906	
Z toho dekompilace		Z toho skenování	
Úspěšné	3906	Detected	3906
Částečně úspěšné	0	Suspicious	0
Selhání	0	Undetected	0
Z toho SDK Info			
Minimum: MinSdkVersion Maximum: MinSdkVersion		1 21	
Minimum: TargetSdkVersion Maximum: TargetSdkVersion		4 27	
Defekty v datové sadě			
Duplicity		0	
Nekompletní APK		4	
Nevalidní soubory		452	
Neúspěšná statická analýza		3	

Tabulka 7.8 Charakteristiky datové sady 6.3

Všechny nalezené balíčky v sadě pak byly správně zařazeny do třídy "detected" a z celkových 3906 aplikací se podařilo kompletně analyzovat 3903 z nich, pouze tři aplikace tak vyžadovaly další analýzu rutinních logů.

Vzorky v této části sady pak vykazují nejvyšší hodnotu maximálních cílových verzí SDK, kdy bylo dosaženo cílové verze 27, která byla v roce 2018 jednou z nejaktuálnějších. Stále je však u sady doporučena filtrace nejméně reprezentativních vzorků a to zejména u aplikací nepřesahujících cílovou verzi SDK 20.

7.7.4 CICMalDroid 2020: SMS

Poslední a zároveň největší část datové sady CICMalDroid 2020 je zaměřena na SMS malware. Jen v této části sady se podařilo vytěžit o téměř 1000 aplikací více, než by tomu bylo při použití emulátoru. Stejně jako u předchozích částí vykazuje datová sada CICMalDroid vysokou kvalitu zpracovaných soubrů a to zejména proto, že všechny nalezené balíčky byly dekompilovatelné, patří do třídy "detected" a pouze dva z nich nebyly úspěšně staticky analyzovány, jak prezentuje tabulka 7.9.

CICMalDroid 2020: SMS			
SHA256: 765fa6e2abc0f84bf3b939f881854c21d16bb8b54afbf22dfc08ce5ada4f294f			
Očekávaný počet APK		3904	
Reálný počet APK		4817	
Z toho dekompilace		Z toho skenování	
Úspěšné	4817	Detected	4817
Částečně úspěšné	0	Suspicious	0
Selhání	0	Undetected	0
Z toho SDK Info			
Minimum: MinSdkVersion		1	
Maximum: MinSdkVersion		11	
Minimum: TargetSdkVersion		4	
Maximum: TargetSdkVersion		24	
Defekty v datové sadě			
Duplicity		0	
Nekompletní APK		4	
Nevalidní soubory		1	
Neúspěšná statická analýza		2	

Tabulka 7.9 Charakteristiky datové sady 6.4

Problematické jsou opět verze SDK, kde je nutná filtrace, některé vzorky byly poprvé oskenovány službou VirusTotal v roce 2012, což v roce 2022 není dostatečně reprezentativní. Defekty v datové sadě jsou reprezentovány vzorky, jenž nemají dex soubor a jsou tedy správně vyloučeny z výsledné datové sady. Nevalidní soubor je potom opět bash skript sloužící k přejmenování souborů, takže je také vyloučen.

8 POPIS VÝSLEDNÉ DATOVÉ SADY

Do výsledné datové sady byly zahrnuty všechny nalezené validní APK balíčky, avšak deset balíčků bylo duplicitní napříč datovými sadami mobilního malwaru, takže celkový počet zpracovaných unikátních balíčků činí 13411. Shrnutí počtu vytěžených aplikací v rámci jednotlivých kolekcí je prezentováno tabulkou 8.1

Zpracované datové sady		
ID	Název sady	Počet validních balíčků
1	AndroidMalware_2018	108
2	AndroidMalware_2019	155
3	AndroidMalware_2020	192
4	AndroidMalware_2021	115
5	Android Malware Sample Library	164
6	CICMalDroid 2020	12687
Celkem		13421
7	AndroBank (unikátní v systému)	13411

Tabulka 8.1 Zpracované balíčky

Statická analýza byla úspěšně provedena u 13388 APK balíčků, čímž byly získány významné charakteristiky mobilního malwaru.

Ze skenování vyšlo najevo, že celkem 11 balíčků bylo zařazeno do třídy příslušnosti "suspicious", dalších 25 balíčků bylo označeno jako "undetected" a celkem 13375 vzorků bylo správně zařazeno do třídy "detected".

Výsledná datová sada obsahuje ke každému balíčku přidružené informace. Jedná se o obecné informace - původ vzorku a další; výsledky skenování - detailní informace o vyhodnocení jednotlivými AV programy; a množinu významných charakteristik mobilního malwaru v podobě dat ze statické analýzy, ale také informací získaných službou VirusTotal, jako data prvního skenování.

Vznikla tak rozsáhlá datová sada, která pokrývá mobilní malware přibližně od roku 2012 až do roku 2020. Všechny balíčky jsou obsaženy ve výsledné datové sadě, avšak pro navazující výzkumnou činnost je při použití sady doporučeno filtrování aplikací, jenž splňují podmínku pro dostatečnou reprezentativnost. Záměrně byly ponechány všechny vzorky, protože hranice této podmínky nejsou přesně specifikovány a neexistuje žádný standard od kterého lze odvodit právě tyto hranice. Nabízí se však již zmíněný postup, kdy je možné sledovat minimální verze, jenž jsou podporovány anti-malwarovými programy pro operační systém Android.

ZÁVĚR

V rámci analýzy dostupných datových sad bylo vybráno šest datových sad, které byly úspěšně zpracovány. Práce si kladla za cíl zlepšit situaci na poli datových sad, což se díky kompletnímu preprocessingu a standardizaci povedlo. Všechny datové sady zpracované vytvořeným multiagentním systémem byly zbaveny veškerých defektů a nežádoucích jevů. Dalším cílem práce bylo upozornit na klíčové vlastnosti datových sad mobilního malwaru, kdy jsou vyzdvihnuty zejména vlastnosti jako zařazení do časového intervalu působnosti, ale také ověření malignity. Z analýzy datových sad, jež byly zpracovány dále vyplývá, že dostupné sady vykazují vysoké kvality, ale nepředstavují hotový produkt, který je možné okamžitě použít pro navazující výzkumnou činnost. Teoreticky to možné je, ale jak vychází z praktické části práce je žádoucí další zpracování těchto kolekcí mobilního malwaru. Doporučené kroky před zahájením navazujících výzkumů jsou: předzpracování sady kvůli odstranění defektů; ověření dekompilovatelnsti pro zaručení analyzovatelnosti vzorků; ověření integrity dat ve smyslu jejich škodlivost pro zamezení zkreslení výsledků výzkumu (vyřazení legitimních vzorků ze sady mobilního malwaru); filtrace vzorků dle podmínky pro dostatečnou reprezentativnost.

Kromě zpracování datových sad do podoby umožňující navazující výzkumnou činnost, byly také extrahovány významné charakteristiky mobilního malwaru a to zejména provedením automatizované statické analýzy. Výsledná datová sada obsahuje více, než 13000 vzorků mobilního malwaru ke kterým jsou přiřazeny jejich charakteristiky a kompletní výsledky skenování nástrojem VirusTotal. Podařilo se upozornit na významné problémy, mající negativní vliv na navazující výzkumy a byly prezentovány i doporučení, jak tyto problémy eliminovat.

Součástí práce je také popis základních částí operačního systému Android a také příklady jejich zneužití. Práce také poukazuje na problematiku se zastaralými verzemi Android OS. Procentuálně se totiž může jevit počet těchto zařízení velmi nízký, avšak vezmeme li v potaz celkový počet zařízení s Android OS, pak se rozhodně nejedná o zanedbatelné množství.

OMEZENÍ PRÁCE A DALŠÍ MOŽNÝ VÝZKUM

Při zpracování datových sad mobilního malwaru se podařilo odhalit v systému několik míst, která vykazují jistý prostor pro zlepšení. Z analýzy vyplynulo, ačkoliv to není úplně obvyklé, že mobilní malware pro operační systém Android nemusí být nutně distribuován v APK balíčku, ale i ve formátu dex. Pro maximalizaci informačního zisku z datové sady by bylo vhodné přidání funkcionality ke zpracování dex souborů s jejich analýzou. Pro maximalizaci informačního zisku z jednotlivých APK balíčků je vhodné analyzovat rutinní logy a patřičně reagovat na balíčky, které se při provedení statické analýzy ukázaly, jako problematické. Dále by byla vhodná hlubší analýza souborů, které byly označeny jako defektní, jedná se o soubory: ELF, shell a Java Archive. Další analýza by bylo vhodná i u souborů, které byly označeny vydavateli datových sad jako malware, ale následným skenováním byly zařazeny do tříd "suspicious" a "undetected".

Obecně chybí definice podmínky pro dostatečnou reprezentativnost, tedy na minimální úroveň API ke zkoumání. Tato definice by bezpečnostním analytikům značně usnadnila vyhledávání relevantních vzorků s ohledem na jejich interval časové působnosti a umožnilo by to i přesnou filtraci vzorků.

Pro testování detekčního potenciálu nástrojů založených na umělé inteligenci jsou potřebné i vzorky legitimní. Touto problematikou se autor zabývá v rámci laboratoře PTLAB. Ideální datová sada, která by poskytovala dostatečný kontrast mezi legitimními a škodlivými aplikacemi by měla obsahovat přibližně stejný počet aplikací, jako má datová sada mobilního malwaru a měla by zahrnovat několik kategorií aplikací.

SEZNAM POUŽITÉ LITERATURY

[1] Desktop vs Mobile vs Tablet Market Share Worldwide. *StatCounter: Global Stats* [online]. c1999-2022 [cit. 2022-04-11]. Dostupné z: https://gs.statcounter.com/platform-market-share/desktop-mobile-tablet/worldwide

[2] Desktop vs Mobile vs Tablet Market Share in Africa - March 2022. *StatCounter: Global Stats* [online]. c1999-2022 [cit. 2022-04-05]. Dostupné z: https://gs.statcounter.com/platform-market-share/desktop-mobile-tablet/africa

[3] Mobile & Tablet Operating System Market Share Worldwide - March 2022. *StatCounter* [online]. c1999-2022 [cit. 2022-04-05]. Dostupné z: https://gs.statcounter.com/os-market-share/mobile-tablet/worldwide/

[4] SAMAT, Sameer. Android 12 Beta: Designed for you. *The Keyword* [online]. Kalifornie: Google, 18 May 2021 [cit. 2022-04-06]. Dostupné z: https://www.blog.google/products/android/android-12-beta/

[5] VirusTotal. *VirusTotal* [online]. Dublin [cit. 2022-04-05]. Dostupné z: https://www.virustotal.com/

[6] RASHED, Mohammed a Guillermo SUAREZ-TANGIL. An Analysis of Android Malware Classification Services. *Sensors* [online]. MDPI, 2021, **21**(16), 1-14 [cit. 2022-04-06]. ISSN 14248220. Dostupné z: doi:10.3390/s21165671

[7] ZHOU, Yajin a Xuxian JIANG. Dissecting Android Malware: Characterization and Evolution. In: *2012 IEEE Symposium on Security and Privacy* [online]. San Francisco: IEEE, 2012, 2012, s. 95-109 [cit. 2022-04-29]. ISBN 978-1-4673-1244-8. ISSN 2375-1207. Dostupné z: doi:10.1109/SP.2012.16

[8] WEI, Fengguo, Yuping LI, Sankardas ROY, Xinming OU a Wu ZHOU. Deep Ground Truth Analysis of Current Android Malware. POLYCHRONAKIS, Michalis a Michael MEIER, ed. *Detection of Intrusions and Malware, and Vulnerability Assessment* [online]. Cham: Springer International Publishing, 2017, 2017-06-04, s. 252-276 [cit. 2022-04-12]. Lecture Notes in Computer Science. ISBN 978-3-319-60875-4. Dostupné z: doi:10.1007/978-3-319-60876-1_12

[9] *AMD* [online]. Argus Lab, 2017 [cit. 2022-04-15]. Dostupné z: http://amd.arguslab.org/

[10] Android Malware Genome Project: Dataset Release Policy. *Malgenome* [online]. [cit. 2022-04-28]. Dostupné z: http://www.malgenomeproject.org/policy.html

[11] XIA, Haoran. *Ultimate feature extractor for Android mobile applications - Technical Features*. Groningen, 2020. Bachelor theses. University of Groningen.

[12] PICARD, S., C. CHAPDELAINE, C. CAPPI, L. GARDES, E. JENN, B. LEFEVRE a T. SOUMARMON. Ensuring Dataset Quality for Machine Learning Certification. In: *2020 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)* [online]. IEEE, 2020, 2020, s. 275-282 [cit. 2022-04-11]. ISBN 978-1-7281-7735-9. Dostupné z: doi:10.1109/ISSREW51248.2020.00085

[13] GOOGLE. Android 6.0 Changes. *Android Developer* [online]. Mountain View: Google, 2021 [cit. 2022-04-06]. Dostupné z: https://developer.android.com/about/versions/marshmallow/android-6.0-changes

[14] Endoflife.date: Android. *Endoflife.date* [online]. [cit. 2022-04-11]. Dostupné z: https://endoflife.date/android

[15] GOOGLE. 2022 Android Security Bulletins. *Android Open Source Project* [online]. Mountain View: Google, 2022 [cit. 2022-04-06]. Dostupné z: https://source.android.com/security/bulletin/2022

[16] Mobile Android operating system market share by version worldwide from January 2018 to December 2021. *Statista* [online]. c2022 [cit. 2022-04-05]. Dostupné z: https://www.statista.com/statistics/921152/mobile-andrFoid-version-share-worldwide/

[17] MAHDAVIFAR, Samaneh, Andi Fitriah ABDUL KADIR, Rasool FATEMI, Dima ALHADIDI a Ali A. GHORBANI. Dynamic Android Malware Category Classification using Semi-Supervised Deep Learning. In: *2020 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)* [online]. IEEE, 2020, 2020, s. 515-522 [cit. 2022-04-12]. ISBN 978-1-7281-6609-4. Dostupné z: doi:10.1109/DASC-PICom-CBDCom-CyberSciTech49142.2020.00094

[18] Datasets: CIC. *Canadian Institute for Cybersecurity* [online]. New Brunswick: University of New Brunswick, c2022 [cit. 2022-04-12]. Dostupné z: https://www.unb.ca/cic/datasets/index.html

[19] ABDUL KADIR, Andi Fitriah, Natalia STAKHANOVA a Ali Akbar GHORBANI. Android Botnets: What URLs are Telling Us. QIU, Meikang, Shouhuai XU, Moti YUNG a Haibo ZHANG, ed. *9th International Conference on Network and System Security (NSS)* [online]. New York: Springer International Publishing, 2015, 2015-11-06, s. 78-91 [cit. 2022-04-15]. Lecture Notes in Computer Science. ISBN 978-3-319-25644-3. Dostupné z: doi:10.1007/978-3-319-25645-0_6

[20] ALLIX, K., T. BISSYANDE, J. KLEIN a Y. TRAON. AndroZoo: Collecting Millions of Android Apps for the Research Community. In: *In Proceedings of the 2016 IEEE/ACM 13th Working Conference on Mining Software Repositories (MSR)* [online]. Austin: IEEE Computer Society, 2016, s. 468-471 [cit. 2022-04-13]. ISBN 978-1-4503-4186-8. Dostupné z: doi:10.1109/MSR.2016.056

[21] MUELLER, Bernhard, Sven SCHLEIER, Jeroen WILLEMSEN a Carlos HOLGUERA. *OWASP Mobile Security Testing Guide: The Ultimate Guide to Mobile App Security Testing and Reverse Engineering*. Maryland: OWASP, 2022. ISBN 9781257966363.

[22] KOTZIAS, Platon, Juan CABALLERO a Leyla BILGE. How Did That Get In My Phone? Unwanted App Distribution on Android Devices. In: *2021 IEEE Symposium on Security and Privacy (SP)* [online]. San Francisco: IEEE, 2021, 2021, s. 53-69 [cit. 2022-04-20]. ISBN 978-1-7281-8934-5. ISSN 2375-1207. Dostupné z: doi:10.1109/SP40001.2021.00041

[23] Preinstalled Android Apps on Samsung Devices: Threat Guidance. *Lookout Threat Lab* [online]. San Francisco: Lookout, 2022 [cit. 2022-04-20]. Dostupné z: https://www.lookout.com/documents/threat-reports/us/lookout-preinstalled-android-apps-samsung-tg-us.pdf

[24] GOOGLE. Platform Architecture. *Android for Developers* [online]. Mountain View: Google, 2021 [cit. 2022-04-06]. Dostupné z: https://developer.android.com/guide/platform

[25] VELU, Vijay Kumar. *Mobile Application Penetration Testing: Explore real-world threat scenarios, attacks on mobile applications, and ways to counter them*. Birmingham: Packt Publishing, 2016. ISBN 9781785883378.

[26] KLEYMENOV, Alexey a Amr THABET. *Mastering Malware Analysis: The complete malware analyst's guide to combating malicious software, APT, cybercrime, and IoT attacks*. Birmingham: Packt Publishing, 2019. ISBN 9781789610789.

[27] CRĂCIUNESCU, Denis. The Layers of the Android Security Model: Why do attackers exploit the users and not the operating system itself?. *Pro Android Dev* [online]. San Francisco: Medium, 2020 [cit. 2022-04-21]. Dostupné z: https://proandroiddev.com/the-layers-of-the-android-security-model-90f471015ae6

[28] GOOGLE. System and kernel security. *Android for Developers* [online]. Mountain View: Google, 2020 [cit. 2022-04-06]. Dostupné z: https://source.android.com/security/overview/kernel-security.html

[29] GOOGLE. HIDL. *Android for Developers* [online]. Mountain View: Google, 2022 [cit. 2022-04-06]. Dostupné z: https://source.android.com/devices/architecture/hidl

[30] GOOGLE. AIDL Overview. *Android for Developers* [online]. Mountain View: Google, 2022 [cit. 2022-04-06]. Dostupné z: https://source.android.com/devices/architecture/aidl/overview

[31] GOOGLE. Legacy HALs. *Android for Developers* [online]. Mountain View: Google, 2020 [cit. 2022-04-06]. Dostupné z: https://source.android.com/devices/architecture/hal

[32] JETBRAINS. FAQ: What does Kotlin compile down to?. *Kotlin Lang* [online]. Praha: Kotlin Foundation, 2022 [cit. 2022-04-06]. Dostupné z: https://kotlinlang.org/docs/faq.html#what-does-kotlin-compile-down-to

[33] GOOGLE. Android Runtime (ART) and Dalvik. *Android for Developers* [online]. Mountain View: Google, 2020 [cit. 2022-04-06]. Dostupné z: https://source.android.com/devices/tech/dalvik

[34] GOOGLE. Configuring ART: How ART works. *Android for Developers* [online]. Mountain View: Google, 2022 [cit. 2022-04-06]. Dostupné z: https://source.android.com/devices/tech/dalvik/configure#how_art_works

[35] GOOGLE. Get started with the NDK. *Android for Developers* [online]. Mountain View: Google, 2020 [cit. 2022-04-22]. Dostupné z: https://developer.android.com/ndk/guides

[36] GOOGLE. Create an Android library. *Android for Developers* [online]. Mountain View: Google, 2022 [cit. 2022-04-22]. Dostupné z: https://developer.android.com/studio/projects/android-library

[37] STONE, Maddie. Reverse Engineering Android Apps - Native Libraries. *Android App Reverse Engineering 101: Learn to reverse engineer Android applications* [online]. GitHub, 2020 [cit. 2022-04-22]. Dostupné z: https://www.ragingrock.com/AndroidAppRE/reversing_native_libs.html

[38] LU, Kai. Deep Analysis of Android Rootnik Malware Using Advanced Anti-Debug and Anti-Hook, Part II: Analysis of The Scope of Java. *Fortinet: Threat Research* [online]. Sunnyvale: Fortinet, 2022 [cit. 2022-04-22]. Dostupné z: https://www.fortinet.com/blog/threat-research/deep-analysis-of-android-rootnik-malware-using-advanced-anti-debug-and-anti-hook-part-ii-analysis-of-the-scope-of-java

[39] APVRILLE, Axelle. REVERSE ANDROID MALWARE LIKE A JEDI MASTER. In: *VB2021 localhost: Virus Bulletin International Conference* [online]. virtual event: Virus Bulletin, 2021 [cit. 2022-04-22]. Dostupné z: https://vblocalhost.com/uploads/VB2021-Apvrille.pdf

[40] GOOGLE. Permissions on Android. *Android for Developers* [online]. Mountain View: Google, 2022 [cit. 2022-04-23]. Dostupné z: https://developer.android.com/guide/topics/permissions/overview

[41] ESET. ESET Mobile Security & Antivirus: 7.3.15.0. *Google Play Store* [mobile app]. 2022 [cit. 2022-04-26]. Dostupné z: https://play.google.com/store/apps/details?id=com.eset.ems2.gp

[42] GOOGLE. Application Fundamentals. *Android for Developers* [online]. Mountain View: Google, 2021 [cit. 2022-04-23]. Dostupné z: https://developer.android.com/guide/components/fundamentals

[43] TeaBot: a new Android malware emerged in Italy, targets banks in Europe. *Cleafy: Labs* [online]. Milano: Cleafy, 2021, 2021 [cit. 2022-04-25]. Dostupné z: https://www.cleafy.com/cleafy-labs/teabot

[44] SharkBot: a new generation of Android Trojans is targeting banks in Europe. *Cleafy: Labs* [online]. Milano: Cleafy, 2021, 2021 [cit. 2022-04-25]. Dostupné z: https://www.cleafy.com/cleafy-labs/sharkbot-a-new-generation-of-android-trojan-is-targeting-banks-in-europe

[45] GOOGLE. Services overview. *Android for Developers* [online]. Mountain View: Google, 2021 [cit. 2022-04-23]. Dostupné z: https://developer.android.com/guide/components/services

[46] Event Triggered Execution: Broadcast Receivers. *MITRE ATT&CK* [online]. MITRE, 2022 [cit. 2022-04-25]. Dostupné z: https://attack.mitre.org/techniques/T1624/001/

[47] Pegasus for Android: Technical Analysis and Findings of Chrysaor. *Lookout: Threat Intelligence* [online]. 2017 [cit. 2022-04-25]. Dostupné z: https://info.lookout.com/rs/051-ESQ-475/images/lookout-pegasus-android-technical-analysis.pdf

[48] GOOGLE. <provider>. *Android for Developers* [online]. Mountain View: Google, 2021 [cit. 2022-04-23]. Dostupné z: https://developer.android.com/guide/topics/manifest/provider-element

[49] Protected User Data: Contact List. *MITRE ATT&CK* [online]. MITRE, 2022 [cit. 2022-04-26]. Dostupné z: https://attack.mitre.org/techniques/T1636/003/

[50] Protected User Data: SMS Messages. *MITRE ATT&CK* [online]. MITRE, 2022 [cit. 2022-04-26]. Dostupné z: https://attack.mitre.org/techniques/T1636/004/

[51] Protected User Data: Call Log. *MITRE ATT&CK* [online]. MITRE, 2022 [cit. 2022-04-26]. Dostupné z: https://attack.mitre.org/techniques/T1636/002/

[52] GOOGLE. Intent. *Android for Developers* [online]. Mountain View: Google, 2021 [cit. 2022-04-26]. Dostupné z: https://developer.android.com/reference/android/content/Intent

[53] GOOGLE. About Android App Bundles. *Android for Developers* [online]. Mountain View: Google, 2022 [cit. 2022-04-26]. Dostupné z: https://developer.android.com/guide/app-bundle

[54] WIŚNIEWSKI, Ryszard a Connor TUMBLESON. Apktool: 2.6.1. *Apktool: A tool for reverse engineering Android apk files* [software]. 2022 [cit. 2022-04-26]. Dostupné z: https://ibotpeaches.github.io/Apktool/install/

[55] LASHKARI, Arash Habibi, Andi Fitriah A.KADIR, Hugo GONZALEZ, Kenneth Fon MBAH a Ali A. GHORBANI. Towards a Network-Based Framework for Android Malware Detection and Characterization. In: *2017 15th Annual Conference on Privacy, Security and Trust (PST)* [online]. IEEE, 2017, 2017, s. 233-23309 [cit. 2022-04-27]. ISBN 978-1-5386-2487-6. Dostupné z: doi:10.1109/PST.2017.00035

[56] *VirusShare: Because Sharing is Caring* [online]. Corvus Forensics [cit. 2022-04-26]. Dostupné z: https://virusshare.com/

[57] MAIORCA, Davide, Davide ARIU, Igino CORONA, Marco ARESU a Giorgio GIACINTO. Stealth attacks: An extended insight into the obfuscation effects on Android malware. *Computers & Security* [online]. Elsevier, 2015, **51**, 16-31 [cit. 2022-04-27]. ISSN 01674048. Dostupné z: doi:10.1016/j.cose.2015.02.007

[58] SK3PTRE. AndroidMalware_2018. *GitHub* [online]. GitHub, c2022, 15 Feb 2019 [cit. 2022-04-26]. Dostupné z: https://github.com/sk3ptre/AndroidMalware_2018

[59] GitHub. *Where the world builds software* [software]. Github, c2022 [cit. 2022-04-26]. Dostupné z: https://github.com/

[60] SK3PTRE. *AndroidMalware_2019* [online]. GitHub, c2022, 1 Jan 2020 [cit. 2022-04-26]. Dostupné z: https://github.com/sk3ptre/AndroidMalware_2019

[61] SK3PTRE. *AndroidMalware_2020* [online]. GitHub, c2022, 6 Jan 2021 [cit. 2022-04-26]. Dostupné z: https://github.com/sk3ptre/AndroidMalware_2020

[62] SK3PTRE. AndroidMalware_2021. *GitHub* [online]. GitHub, c2022 [cit. 2022-04-26]. Dostupné z: https://github.com/sk3ptre/AndroidMalware_2021

[63] DEMIRHAN, Mustafa Kaan. *Android Malware Samples Library* [online]. GitHub, c2022, 19 Sep 2020 [cit. 2022-04-26]. Dostupné z: https://github.com/mstfknn/android-malware-sample-library

[64] *Contagio Mobile Malware Mini Dump* [online]. [cit. 2022-04-26]. Dostupné z: https://contagiominidump.blogspot.com/

[65] *MalwareBazaar: Database* [online]. abuse.ch, c2022 [cit. 2022-04-26]. Dostupné z: https://bazaar.abuse.ch/browse/

[66] KOULIARIDIS, Vasileios, Georgios KAMBOURAKIS a Tao PENG. Feature Importance in Android Malware Detection. In: *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)* [online]. IEEE, 2020, 2020, s. 1449-1454 [cit. 2022-04-28]. ISBN 978-1-6654-0392-4. Dostupné z: doi:10.1109/TrustCom50675.2020.00195

[67] VANDERPLAS, Jacob T. *Python data science handbook: essential tools for working with data*. Sebastopol, CA: O'Reilly Media, 2016. ISBN 1491912057.

[68] JOSHI, Prateek. *Artificial Intelligence with Python* [online]. Birmingham: Packt Publishing, 2017 [cit. 2021-10-20]. ISBN 9781786464392. Dostupné z: https://www.packtpub.com/product/artificial-intelligence-with-python/9781786464392

[69] Machine Learning for Malware Detection. *Kaspersky Lab* [online]. Kaspersky Lab, c2022 [cit. 2022-04-26]. Dostupné z: https://media.kaspersky.com/en/enterprise-security/Kaspersky-Lab-Whitepaper-Machine-Learning.pdf

[70] *Canadian Institute for Cybersecurity: Canada-s research leader in cybersecurity* [online]. New Brunswick: University of New Brunswick, c2022 [cit. 2022-05-01]. Dostupné z: https://www.unb.ca/cic/

[71] Apklab.io: A Mobile Threat Intelligence Platform by Avast. *Avast* [online]. Praha: Avast Software, 2022 [cit. 2022-04-28]. Dostupné z: https://www.apklab.io/

[72] LASHKARI, Arash Habibi, Andi Fitriah A. KADIR, Laya TAHERI a Ali A. GHORBANI. Toward Developing a Systematic Approach to Generate Benchmark Android Malware Datasets and Classification. In: *2018 International Carnahan Conference on Security Technology (ICCST)* [online]. Montreal: IEEE, 2018, 2018, s. 1-7 [cit. 2022-04-29]. ISBN 978-1-5386-7931-9. ISSN 2153-0742. Dostupné z: doi:10.1109/CCST.2018.8585560

[73] TAHERI, Laya, Andi Fitriah Abdul KADIR a Arash Habibi LASHKARI. Extensible Android Malware Detection and Family Classification Using Network-Flows and API-Calls. In: *2019 International Carnahan Conference on Security Technology (ICCST)* [online]. India: IEEE, 2019, 2019, s. 1-8 [cit. 2022-04-29]. ISBN 978-1-7281-1576-4. ISSN 2153-0742. Dostupné z: doi:10.1109/CCST.2019.8888430

[74] MAHDAVIFAR, Samaneh, Dima ALHADIDI a Ali. A. GHORBANI. Effective and Efficient Hybrid Android Malware Classification Using Pseudo-Label Stacked Auto-Encoder. *Journal of Network and Systems Management* [online]. 2022, **30**(1), 34 [cit. 2022-05-01]. ISSN 1064-7570. Dostupné z: doi:10.1007/s10922-021-09634-4

[75] RAHALI, Abir, Arash Habibi LASHKARI, Gurdip KAUR, Laya TAHERI, FRANCOIS GAGNON a Frédéric MASSICOTTE. DIDroid: Android Malware Classification and Characterization Using Deep Image Learning. In: *2020 the 10th International Conference on Communication and Network Security* [online]. New York: Association for Computing Machinery, 2020, 27. 11. 2020, s. 70-82 [cit. 2022-05-01]. ISBN 9781450389037. Dostupné z: doi:10.1145/3442520.3442522

[76] KEYES, David Sean, Beiqi LI, Gurdip KAUR, Arash Habibi LASHKARI, Francois GAGNON a Frederic MASSICOTTE. EntropLyzer: Android Malware Classification and Characterization Using Entropy Analysis of Dynamic Characteristics. In: *2021 Reconciling Data Analytics, Automation, Privacy, and Security: A Big Data Challenge (RDAAPS)* [online]. Hamilton: IEEE,

2021, 2021-5-18, s. 1-12 [cit. 2022-04-29]. ISBN 978-1-7281-6937-8. Dostupné z: doi:10.1109/RDAAPS48126.2021.9452002

[77] Android PRAGuard Dataset. *PRA Lab* [online]. Cagliari, 2018 [cit. 2022-04-28]. Dostupné z: https://pralab.diee.unica.it/en/AndroidPRAGuardDataset

[78] *Canadian Centre for Cyber Security* [online]. 2020 [cit. 2022-05-01]. Dostupné z: https://cyber.gc.ca/en/

[79] NOKIA. *Threat Intelligence Report 2021* [online]. NOKIA. Finland, c2022 [cit. 2022-05-01]. CID210870. Dostupné z: https://onestore.nokia.com/asset/210870

[80] ROBERTSON, John, Ahmad DIAB, Ericsson MARIN, Eric NUNES, Vivin PALIATH, Jana SHAKARIAN a Paulo SHAKARIAN. *Darkweb Cyber Threat Intelligence Mining* [online]. Cambridge: Cambridge University Press, 2017 [cit. 2022-05-01]. ISBN 9781316888513. Dostupné z: doi:10.1017/9781316888513

[81] KOYA, Kushwanth a Gobinda CHOWDHURY. A quality and popularity based ranking method for research datasets. In: *2022 4th Asia Pacific Information Technology Conference* [online]. New York, NY, USA: Association for Computing Machinery, 2022, 2022-01-14, s. 103-110 [cit. 2022-04-27]. ISBN 9781450395571. Dostupné z: doi:10.1145/3512353.3512368

[82] DESNOS, Anthony a ANDROGUARD TEAM. *Androguard* [software]. [cit. 2022-05-03]. Dostupné z: https://github.com/androguard/androguard

[83] VAN ROSSUM, Guido a Fred L. DRAKE. *Python 3 Reference Manual*. Scotts Valley: CreateSpace, 2009. ISBN 978-1-4414-1269-0. Dostupné z: doi:10.5555/1593511

[84] *PT LAB: Penetration Testing Laboratory* [online]. 2022 [cit. 2022-05-01]. Dostupné z: https://ptlab.fai.utb.cz/

[85] TELEGRAM. *Telegram* [software]. Telegram FZ LLC and Telegram Messenger [cit. 2022-05-13]. Dostupné z: https://telegram.org

[86] POSTGRESQL. *PostgreSQL* [software]. The PostgreSQL Global Development Group, c1996-2022 [cit. 2022-05-03]. Dostupné z: https://www.postgresql.org/

[87] STOČ 2021. *Studentská tvůrčí a odborná činnost: Výsledky* [online]. 2021 [cit. 2022-05-03]. Dostupné z: http://akce.fs.vsb.cz/2021/stoc/vysledky.pdf

[88] KYBERNETICKÁ BEZPEČNOST 2021. *Řízení procesů a aplikace moderních technologií: Kybernetická bezpečnost* [online]. 2021 [cit. 2022-05-03]. Dostupné z: https://e-konference.utb.cz/kyberneticka-bezpecnost-2021/

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

AOSP	Android Open Source Project
IPC	Meziprocesová komunikace
HW	Hardware
SW	Software
ÖS	Operační systém
HAL	Abstraktní hardwarová vrstva
ART	Android Runtime
JVM	Java Virtual Machine
DVM	Dalvik Virtual Machine
AOT	Ahed of time
JIT	Just in time
DEX	Dalvik Executable
API	Rozhraní pro programování aplikací
UID	Identifikátor uživatele
CCCS	Kanadský institut pro kybernetickou bezpečnost
APK	Android Application Package
GUI	Grafické uživatelské rozhraní
DB	Databáze
VPN	Virtuální privátní síť
SDK	Software development kit

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM PŘÍLOH

PŘÍLOHA P I. CSV SOUBOR SE ZPRACOVANÝMI VZORKY

Vzhledem k povaze dat je veřejně dostupný pouze seznam zpracovaných SHA256 otisků aplikací a jejich charakteristiky v kontextu ke třídě příslušnosti. Pro přístup k vyhodnocení a kompletním charakteristikám mobilního malwaru je nutné kontaktovat laboratoř PTLAB.