Educational Administration: Theory and Practice
2024,30(2), 1320-1327
ISSN:2148-2403
https://kuey.net/
Research Article

Analysis Of The Most Recent Trojans On The Android Operating System

Yukti Tyagi[1*], Dharamveer Singh[2], Ramander Singh[3], Sudhir Dawra[4]

[1*,3]Department of Computer Science and Engineering, R D Engineering College, India
[2]Department of Mechanical Engineering, R D Engineering College, India
[4]Department of Computer Science and Engineering, Inderprastha Engineering College, India

***Corresponding Author:** Yukti Tyagi
*Email:-yuktityagi5027@gmail.com

Citation: Yukti Tyagi,Et.al (2024), Analysis Of The Most Recent Trojans On The Android Operating System ,*Educational Administration: Theory and Practice*, *30*(2), 1320-1327
Doi: 10.53555/kuey.v30i2.6846

ARTICLE INFO | **ABSTRACT**

- With the rapid advancements of electronics, the mobile operating system can accommodate various applications, which greatly facilitates people's everyday life. With a user group of more than 2 billion, the Android platform provides a diverse ecosystem for developing and publishing all sorts of applications. Although Google's official application store, Google Play, contains over 2 million apps, such a huge market also attracts hackers to make profits through distributing malware.
Mobile malware has rocketed since 2009. As reported by Broadcom Inc., an industry- leading security company, 2017 witnessed an increase of new mobile malware strains, com- pared with the year of 2016. Additionally, more profit-driven malware emerged with the growth of underground markets.
Due to the fragmentation problem of the Android plat- form, Android has long been the most targeted operating system suffering from attacks. To keep pace with the cutting-edge antimalware countermeasures adopted by cyber-security businesses, malware developers have abused high-level obfuscation, virtual environment recognition, conditional execution (logic bomb), run-time payload dropping, etc., to fool their opponents (i.e., security defending products and reverse engineering tools).
These techniques are usually more obvious to trace during the evolution and diversification of a malware family. In this thesis, we take a close look into both recent Android trojans and one specific family of Android banking trojan, that infiltrates banking applications to steal credentials or trick victims to type in their usernames and passwords through displaying fake login interfaces.
The results indicate that Android Trojans evolves towards possessing more malicious capabilities and more diverse permutations without losing their core design, which would cause more limitations and ineffectiveness for modern security solutions.

Keywords- Trojans, Android, antimalware

1. Introduction

As mobile integrate more convenient applications that can tackle real life problems, people have become extremely dependent on such small devices, e.g. social networking, news reading, video & audio playing and watching, digital marketing, etc.

Google declared in 2019 [4] that more than 2 billion Android devices were active. In addi- tion, Fig. 1 shows that Android has started to become dominant in the market share against other systems since 2012. The market share of Android came to its peak between 2018 and 2019, with more than 90 percent.

The statistics of available applications in the official store, Google Play peaked at March 2018, with 3.6

million. Although the statistics fell from 3.6 to 2.6 million from March to September due to some policy, the vol- ume of applications in Google play has never decreased since then.

The wide acceptance of the Android system can be attributed to the open-sourced trait of its source code, which facilitates the API (short for Application Programming Interface) calling, testing and debugging process for developers.

However, the side effect of such property stands out due to the severe fragmentation issues. Figure 3 is the screenshot of the distribution dashboard that comes from the official Integrated Development Environment (IDE) in April 2020, Android Studio for Android developers when a new project is created and click on the "Help me choose" link under the minimum SDK (short for Software Development Kit) dropdown. The cumulative distribution data reflect the percentage of devices that your application could be run on, according to the minimum SDK version selected.

Based on these statistics, the distribution for each version is shown in Fig.4. Only less than 10 percent of Android mobile users have upgraded to the latest version, which makes modern anti-virus engines insensitive to those attacks that are aimed at older versions [5, 6, 7, 8, 9].

Although the OS framework fragmentation prevents general exploits from hackers, different vendors of Android OS customize their ecosystem by adding different native libraries, which malware authors can abuse to evade the generic detection of anti-virus engines [7, 8, 10, 11, 12, 13, 14, 15, 16].

Moreover, the speed for OEMs (Original Equipment Manufacturers) and their vendors to fix previous vulnerabilities is slow enough for attackers to

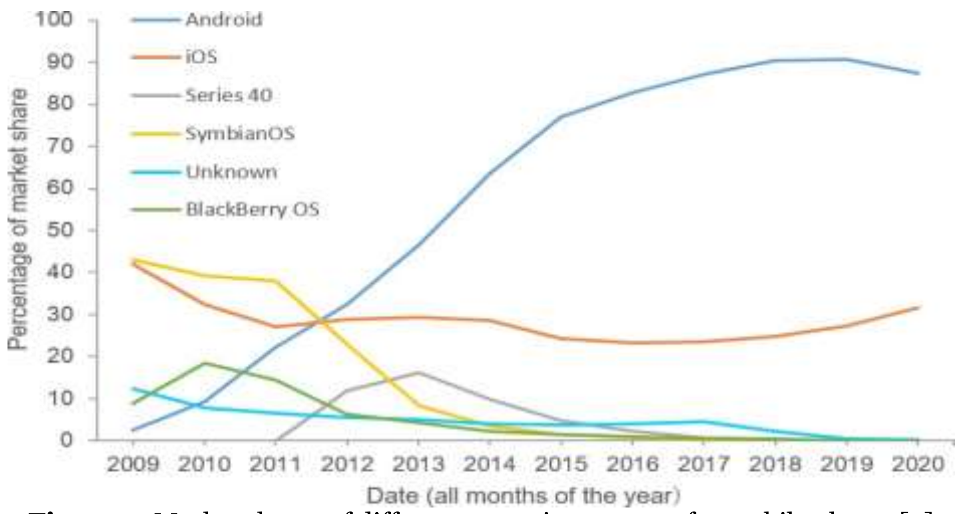


Figure 1: Market shares of different operating systems for mobile phones[2]

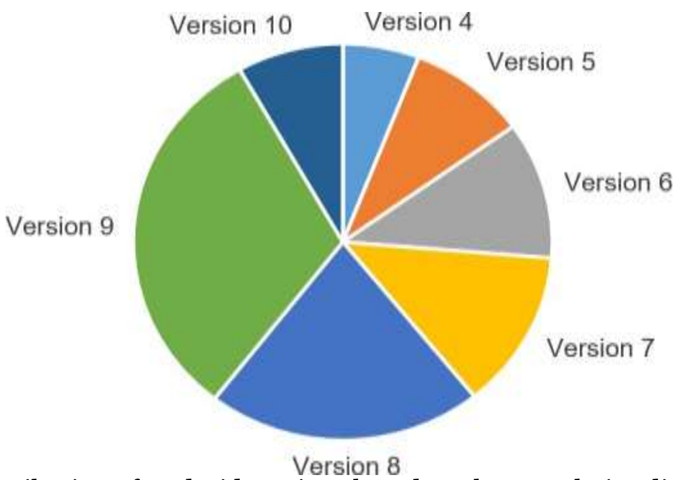


Figure 4: Distribution of Android versions based on the cumulative distribution data

2. System Description

2.1 Android malware and Android banking trojan

Malware is an umbrella word of **mal**icious soft**ware**. The term malicious refers to what could cause damage to the computer system or more intuitively, extremely hinder the user experience once being run. However, the boundary of benign and malicious software is actually not clear.

For instance, some software are programmed to stealthily collect the device information (e.g. android id, manufacturer, model, firmware version), keystroke whatever users are typing and/or record users' speeches.

Finally all of these information or just the most representative or recognizable group (through some local processing) of information is sent to the server that has already been deployed for gathering user data. Generally, Android malware can be classified into 7 categories.

As shown in Table 1, the differences between seven types of malware regarding six properties are displayed. The meanings of the each sub-property is shown below:

Table 1: Differences between different types of malware

		Virus	Worm	Trojan	Backdoor	Spyware	Rootkit	Botnet
	Parasitic	C						
Existing Form	Masquerade		C	C	C	C	C	C
	Independent identity		C					
	Repackaging	C		C	C	C	C	C
	Update attack			C	C	C	C	C
Propagation mode	Sideload	C	C	C	C	C	C	C
	Self-replicate	C	C					
	Local files	C	C	C		C		C
	Network traffic	C	C					C
Attack target	Operating system		C	C	C	C	C	C
	Data theft	C	C	C	C	C	C	C
	Network paralysis	C	C					C
Major risks	System damage	C	C					C
Spreading speed		++	+++	+		+	++	+++
Difficulty of being detected		+	+++	++	++	++	++	++

(a) Parasitic: Attached itself to another executable in order to get executed whenever the host executable is triggered by victim's interaction.

(b) Masquerade: Disguised itself with legitimate apps' meta-data, especially the exte- rior icon and app name that are displayed to users during installation and in the home screen after successful installation.

(c) Independent identity: Reproduce itself continuously and spread the copies through networks or removable media running in the background without any user interac- tion

2. Propagation mode:
(a) Repackaging:
(b) Update attack:
(c) Sideload:
(d) Self-replicate: Malware copies itself continuously without any user interaction.
3. Major risks:
(a) Data theft:
(b) Network paralysis:
(c) System damage:

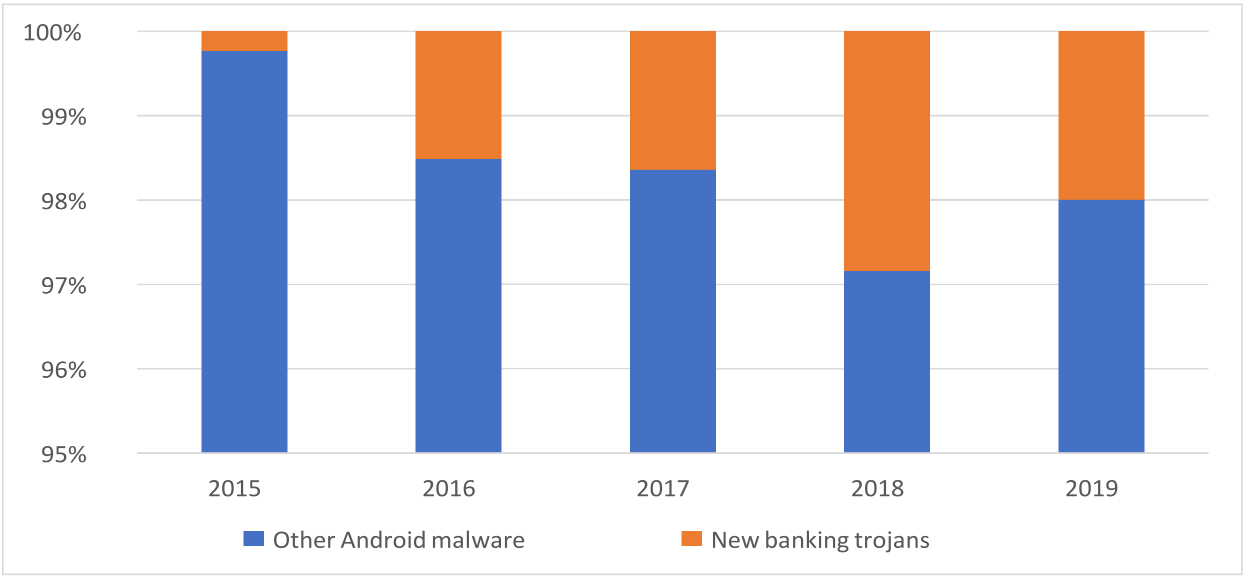


Figure 5: Banking trojans detected by Kaspersky from 2015 to 2019

2.2 Research Objectives

Due to the fact that the normal software and even the Android operating system are continuously refreshing to fix the vulnerabilities and/or add more features, the techniques abused by the malware authors are also constantly being upgraded to become compatible with both the old environments and the new environments. This paper also aims to propose an approach to identify suspicious Android applications without utilizing prior knowledge. At last, to understand the core logic of the Android trojans, a thorough analysis of an Android trojan family would be conducted. What the readers can take away from this thesis are:

i. To know the core modules that is frequently abused by recent popular Android trojans to perform dangerous behaviors.
ii. To understand how to identify suspicious Android applications without much prior knowl- edge.
iii. To understand how a real-world Android trojan works.

3. Methodology

3.1 Android Malware Families Analysis

Even though these applications themselves do not target the device itself, they are used for bad purposes. N.

I. AMINuddIN et al. detect Android trojan based on dynamically extracted system calls [46]. However, some trojans that implement environment-aware techniques are able to evade such approach.

3.1.1 Android malware detection

Other literature is more focused on detection (i.e. a binary decision system that tells whether a given sample is malicious or not), clustering (i.e. a decision system that can aggregate similar samples into a group without any label from prior knowledge).
We can partition the related work of Android malware detection in two classes: (i) run-time monitoring of the invoked events, (ii) static analysis of the code to detect known patterns of misbehaviors.

3.2 Static Approaches

In 2013, an approach named AndroSimilar was proposed by P. Faruki et al. [57]. It attains the accuracy of 60% and follows the foot-print mechanism of known malware.

Further it is used to identify the unknown malware. Fuzzy hashing is used to detect changes made in the application by repackaging. This tool is limited to a small set of malware database.

3.3 Dynamic Approaches and Hybrid Approaches

I. Burguera et al. proposed CrowDroid that uses dynamic analysis of Android applications behavior to detect malware [74].
They used unsupervised machine learning algorithms to detect malware from Android application and results were saved at the server, which achieves accuracy between 85% and 100% depending on malware.

3.4 Deep Analysis of Recent Android Trojans

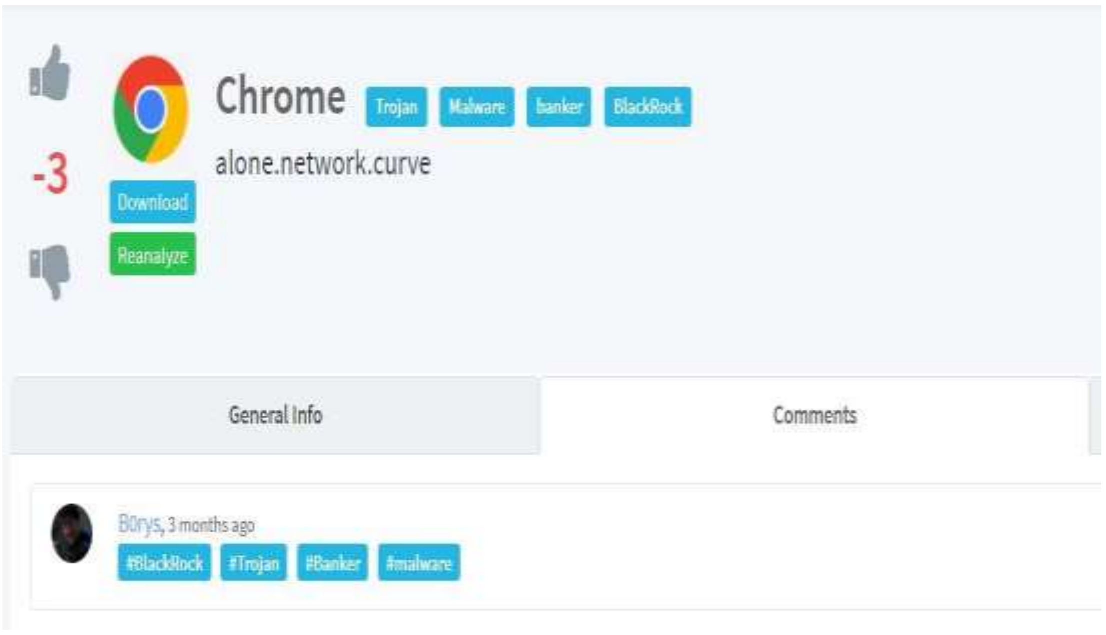
Although some literatures (like [9, 64, 91, 92, 47]) have analysed the technical details of some Android malware thoroughly, the datasets collected by them are somehow obsolete. For instance, even the most recent

research [47] utilises the dataset during 2016-2017; The dataset used by
[92] ranges from 2010 to 2016. Therefore, A newer dataset of Android trojans is a pressing need.

3.4.1 Data Collection and Extraction

To collect newer samples of Android trojans, a simple script for crawling some public security repositories has been written, especially the platform of Koodous [93] and Apkdetect [1]. To illustrate, these platforms have already output labels or classification results given a sample.

limit. track possess one more label, **banker**, they also generate the same label **BlackRock**, as the label **banker** refer to a general type of malware that target the victim's bank card information. Compared with the label **BlackRock**, it is less specific and unique in terms of the coverage.



Chrome Trojan Malware banker BlackRock

alone.network.curve

-3

Download

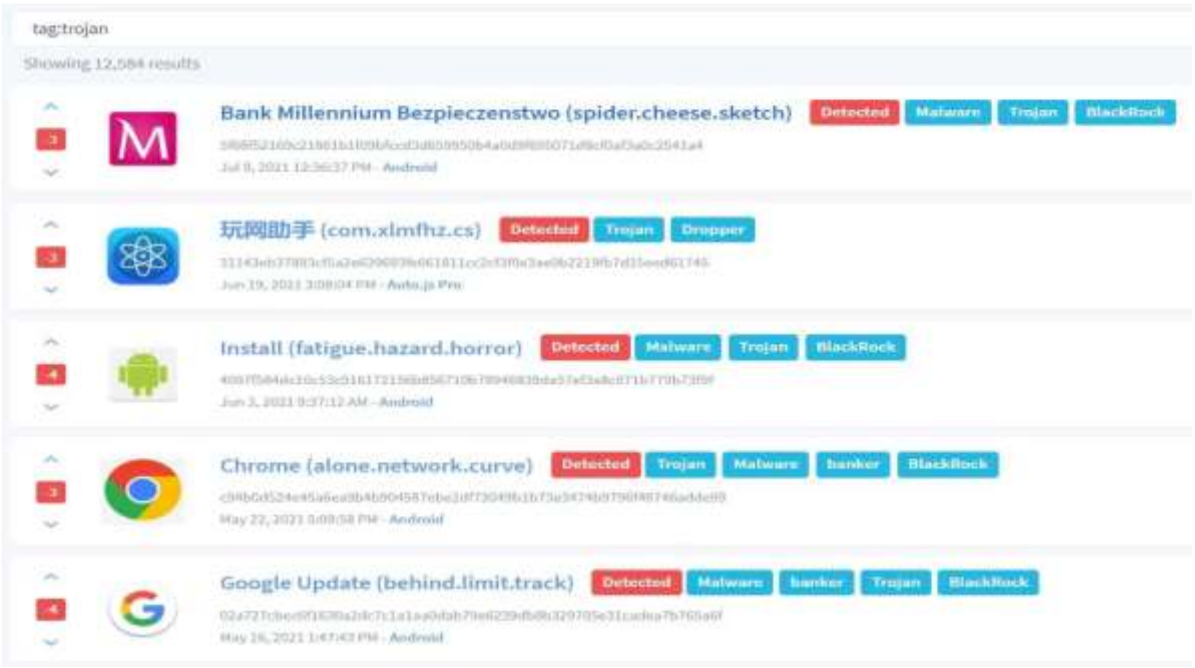
Reanalyze

General Info | Comments

B0rys, 3 months ago

#BlackRock #Trojan #Banker #malware

Figure 7: The screenshot that the hashtags added by an analyst's comment



tag:trojan

Showing 12,584 results

-3 Bank Millennium Bezpieczenstwo (spider.cheese.sketch) Detected Malware Trojan BlackRock
[illegible]
Jul 9, 2021 12:36:37 PM - Android

-3 玩网助手 (com.xlmfhz.cs) Detected Trojan Dropper
[illegible]
Jun 19, 2021 3:08:04 PM - Auto.js Pro

-4 Install (fatigue.hazard.horror) Detected Malware Trojan BlackRock
[illegible]
Jun 3, 2021 9:37:12 AM - Android

-3 Chrome (alone.network.curve) Detected Trojan Malware banker BlackRock
[illegible]
May 22, 2021 5:09:58 PM - Android

-4 Google Update (behind.limit.track) Detected Malware banker Trojan BlackRock
[illegible]
May 16, 2021 1:47:43 PM - Android

Figure 8: The screenshot of the first five samples when searching with "tag: trojan" on Koodous platform

Apkdetect is similar with Koodous. The differences are that Apkdetect does not provide any convenient API for queries and that the classification name produced by Apkdetect is based on the matching of the configurations of different malware, which are observed and submitted by security researchers.

Due to the fact that Apkdetect does not possess a large volume of samples, a manual collection method is adopted, i.e. before loading the webpage, a chrome browser where the webpage is loaded is used. Specially, the network recording function is activated, which is able to capture and preserve all requests and responses in logs.

Upon typing in the family name that is to be collected, the returned responses mainly include two parts, i.e. one with brief information including uploaded file name, file type, its MD5 hash and the most probable name produced (Figure 10); the other with more details, i.e. the configuration matched and the control & command server extracted from the sample (Figure 11). The second part is parsed to collect the MD5 hash, family name and matched configuration.

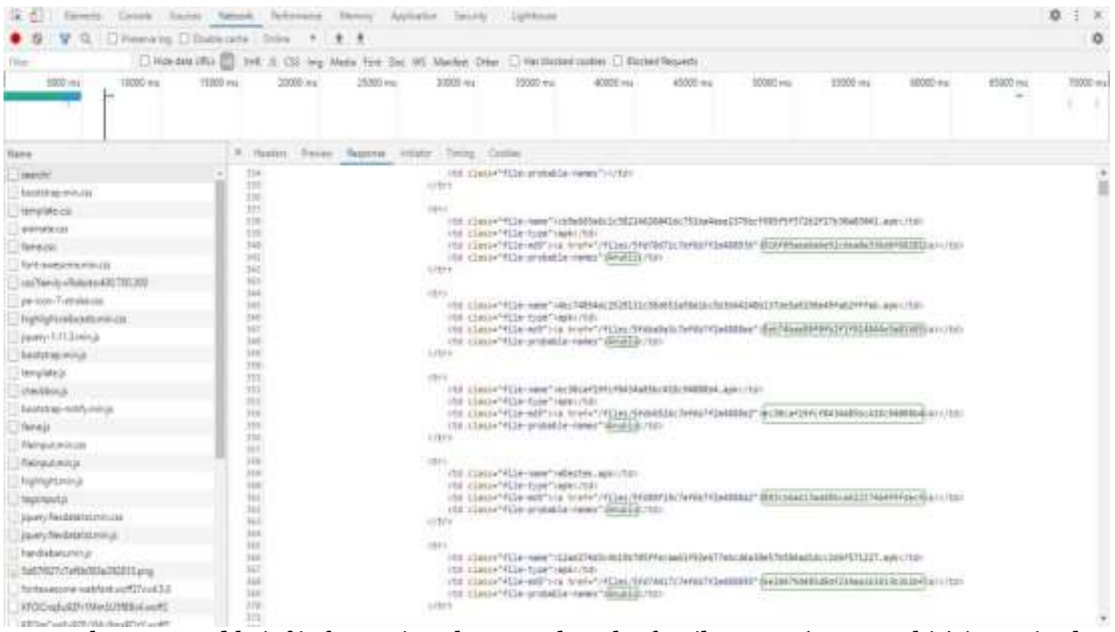


Figure 10: The captured brief information that matches the family name (e.g. Anubis) input in the search box

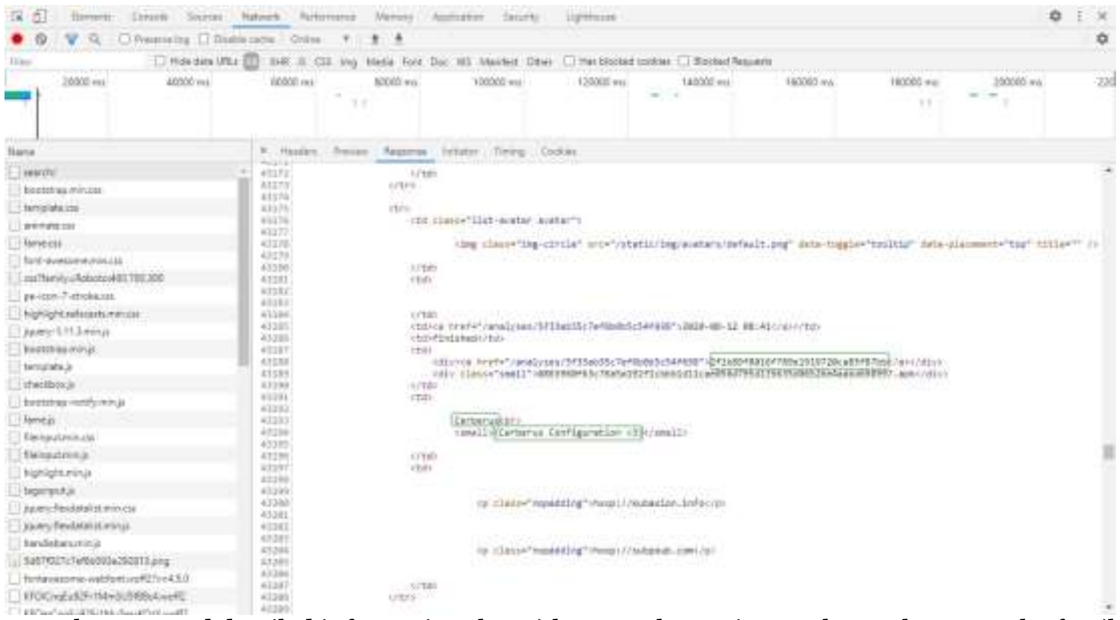


Figure 11: The captured detailed information that either matches or is somehow relevant to the family name

configuration-MD5 pairs) would be discarded from both collections. In the end, the statistics of the data collected are illustrated in Table 2 with the corresponding timeline based on online blog posts from security companies.

Table 2: The timeline of 20 (taking sub-families into account) Android trojans in the collection

Trojan	Number of Samples	Discovered Month

Flexnet	53	2017-07 [94]
RedAlert	521	2017-09 [95]
Bankbot Anubis	722	2017-11 [96]
Catelites	97	2017-12 [97]
Gustuff	25	2018-04 [98]
Hydra v1	84	2018-07 [99]
BianLian	7	2018-10 [100]
Rotexy	5	2018-11 [101]
Cerberus v1	254	2019-06 [102]
Ginp v1	1	2019-06 [103]
Hydra v2	8	2019-06
Brata	3	2019-08 [104]
Ginp v2	1	2019-08
Ginp v3	37	2019-11
Hydra v3	18	2020-02
Cerberus v2	3	2020-04
BlackRock	15	2020-05 [105]
Cerberus v3	476	2020-08
Hydra v4	27	2020-08
Hydra v5	23	2020-10
Total	2380	

4. CONCLUSION AND FUTURE WORK

This paper collects and investigates the recent Android trojans, and profiles them both statically and dynamically in order to illustrate what techniques current Android malware are exploiting to launch attacks towards users. It should be noticed that most trojans can only function well with a live C&C server. If there is no C&C server alive, even though trojans are able to seize top privilege to perform malicious actions, malware authors are not able to harvest any profit from victims. After all, what motivates most trojan authors to publish and distribute trojans is the potential revenues. Planting aggressive adwares on the infected devices is one method of gaining a small amount of profits, but doing so would also have to take the risks of immediately arouse the victim's suspicions as well as annoy victims, As a consequence, advanced countermeasures would be taken to remove the trojan. By contrast, if trojan authors focus on preparing a well- designed scheme to make victims believe the integrity, then it would be highly possible that such strategy gain a proming profit. This, from another perspective explains why the trojans need to evade both static analysis and dynamic analysis as long as possible.
Besides, this thesis also looks deeper into Anubis, a popular Android trojan family that severely infects Android platforms to understand how it has evolved, This work, finished by the writer, has been accepted by and presented at the 7th International Conference on Dependable Systems and Their Applications. The Anubis family has evolved into different versions, where technical details vary while the core code blocks of the family are nearly identical. Their phishing, overlay and code obfuscation mechanisms have been described. Such information will be useful for past, present and future victims. At the same time, how new complex versions emerge based on the old versions has also been illustrated.
As for possible directions of future work, it should be valuable to investigate how to employ hybrid (both static and dynamic) analysis more efficiently to deal with different payload loading patterns (i.e. dynamic loading and native loading), since some APK reinforcement products like Qihoo 360, are able to pack the real payload with multiple layers of encryption, which would make purely static analysis nearly ineffective. On the other hand, dynamic analysis especially in the sandbox environment, some parameters reflecting the infrastructure need to be dynamic or at least not fixed, thus making it possible for trojans that check device-info to perform their real functionalities smoothly.

References

1. apkdetect, https://app.apkdetect.com/, @pr3wtd, accessed: February 2020.
2. "Mobile operating system market share worldwide, 2020," https://gs.statcounter.com/ os-market-share/mobile/worldwide, 2020.
3. E. Alepis and C. Patsakis, "Hey doc, is this normal?: exploring android permissions in the post marshmallow era," in *International Conference on Security, Privacy, and Applied Cryptography Engineering*. Springer, 2017, pp. 53–73.
4. J. Gamba, M. Rashed, A. Razaghpanah, J. Tapiador, and N. Vallina-Rodriguez, "An analysis of pre-installed android software," *arXiv preprint arXiv:1905.02713*, 2019.
5. L. Wei, Y. Liu, and S.-C. Cheung, "Taming android fragmentation: Characterizing and detecting

compatibility issues for android apps," in *Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering*, 2016, pp. 226–237.

6. M. Zheng, P. P. Lee, and J. C. Lui, "Adam: an automatic and extensible platform to stress test android anti-virus systems," in *International conference on detection of intrusions and malware, and vulnerability assessment*. Springer, 2012, pp. 82–101.
7. V. Chebyshev, "Mobile malware evolution 2019," https://securelist.com/ mobile-malware-evolution-2019/96280/, 2020.
8. C. Bai, Q. Han, G. Mezzour, F. Pierazzi, and V. Subrahmanian, "Dbank: Predictive behavioral analysis of recent android banking trojans," *IEEE Transactions on Dependable and Secure Computing*, 2019.
9. Shiv Kumar, Dharamveer Singh, "Energy And Exergy Analysis Of Active Solar Stills Using Compound Parabolic Concentrator" International Research Journal of Engineering and Technology Vols. 6, Issue 12, Dec2019, ISSN (online) 2395-0056. https://www.irjet.net/archives/V6/i12/IRJET-V6I12327.pdf
10. Dharamveer and Samsher, Comparative analyses energy matrices and enviro-economics for active and passive solar still, materialstoday: proceedings, 2020, https://doi.org/10.1016/j.matpr.2020.10.001
11. Dharamveer, Samsher, Anil Kumar, Analytical study of Nth identical photovoltaic thermal (PVT) compound parabolic concentrator (CPC) active double slope solar distiller with helical coiled heat exchanger using C$_u$O Nanoparticles, Desalination and water treatment, 233 (2021) 30-51, https://doi. org/10.5004/dwt.2021.2 7526
12. Dharamveer,Samsher, Anil Kumar, Performance analysis of N-identical PVT-CPC collectors an active single slope solar distiller with a helically coiled heat exchanger using CuO nanoparticles, Water supply, October 2021, https://doi.org/10.2166/ws.2021.348
13. M. Kumar and D. Singh, Comparative analysis of single phase microchannel for heat flow Experimental and using CFD, International Journal of Research in Engineering and Science (IJRES), 10 (2022) 03, 44-58. https://www.ijres.org/papers/Volume-10/Issue-3/Ser-3/G10034458.pdf
14. Subrit and D. Singh, Performance and thermal analysis of coal and waste cotton oil liquid obtained by pyrolysis fuel in diesel engine, International Journal of Research in Engineering and Science (IJRES), 10 (2022) 04, 23-31. https://www.ijres.org/papers/Volume-10/Issue-4/Ser-1/E10042331.pdf
15. Dharamveer Singh, Satyaveer Singh, Ashok Kumar Yadav, Osama Khan, Ashish Dewangan, Saiful Islam, Meshel Q. Alkahtani, Saiful Islam "From Theory to Practice: A Sustainable Solution to Water Scarcity by Using Hybrid Solar Distiller with Heat Exchanger and Aluminum Oxide Nanoparticles" Journal ACS Omega, Vol. 8 (37) 33543–33553, 05 Sep 2023, , https://doi.org/10.1021/acsomega.3c03283
16. Dharamveer Singh "Economic, Enviroeconomic analysis of active solar still using Al2O3 nanoparticles" International Journal of Thermodynamics, Vol. 26 (4) 68-76, 01 Dec 2023, https://doi.org/10.5541/ijot.1295637
17. Dharamveer Singh, Satyaveer Singh, Aakersh Chauhan, Anil Kumar "Enviroeconomic analysis of hybrid active solar desalination system using nanoparticles" Journal of Environmerntal engineering and Science, Vol. 19 (1) 18-28, 08 Aug 2023, https://doi.org/10.1680/jenes.23.00045
18. Dharamveer Singh, Ashok Kumar Yadav, Anil Kumar, Samsher, "Energy matrices and life cycle conversion analysis of N-identical hybrid double slope solar distiller unit using Al_2O_3 nanoparticle". *Journal of Water and Environmental Nanotechnology*, Vol. 8 (3) 267-284, 10 Aug 2023, http://doi:10.22090/jwent.2023.03.006
19. Rajesh Kumar and Dharamveer Singh, "Hygrothermal buckling response of laminated composite plates with random material properties Micro-mechanical model," International Journal of Applied Mechanics and Materials Vols. 110-116 pp 113-119, https://doi.org/10.4028/www.scientific.net/AMM.110-116.113
20. Anubhav Kumar Anup, Dharamveer Singh "FEA Analysis of Refrigerator Compartment for Optimizing Thermal Efficiency" International Journal of Mechanical and Production Engineering Research and Development (IJMPERD) Vol. 10 (3), pp.3951-3972, 30 June 2020**.**
21. Shiv Kumar, Dharamveer Singh, "Optimizing thermal behavior of compact heat exchanger" International Journal of Mechanical and Production Engineering Research and Development (IJMPERD) Vol. 10 (3), pp. 8113-8130, 30 June 2020**.**
22. Y. Zhang, G. Xiao, Z. Zheng, T. Zhu, I. Tsang, and Y. Sui, "An empirical study of code deobfuscations on detecting obfuscated android piggybacked apps," in *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2020, pp. 914–926.
23. Y. Tang, H. Wang, X. Zhan, X. Luo, Y. Zhou, H. Zhou, Q. Yan, Y. Sui, and J. W. Keung, "A systematical study on application performance management libraries for apps," *IEEE Transactions on Software Engineering*, 2021.
24. L. Wang, R. He, H. Wang, P. Xia, Y. Li, L. Wu, Y. Zhou, X. Luo, Y. Sui, Y. Guo *et al.*, "Beyond the virus: a first look at coronavirus-themed android malware," *Empirical Software Engineering*, vol. 26, no. 4, pp. 1–38, 2021.