

Analysis of the Latest Trojans on Android Operating System

by Baodi Ning

Thesis submitted in fulfilment of the requirements for
the degree of

Master of Analytics

under the supervision of Dr. Yulei Sui
and co-supervision of Dr. Jingling Xue

University of Technology Sydney
Faculty of Engineering and Information Technology

January 2021

CERTIFICATE OF ORIGINAL AUTHORSHIP

I, Baodi Ning declare that this thesis, is submitted in fulfilment of the requirements for the award of Master of analytics, in the Faculty of Engineering and Information Technology at the University of Technology Sydney.

This thesis is wholly my own work unless otherwise referenced or acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

This document has not been submitted for qualifications at any other academic institution.

This research is supported by the Australian Government Research Training Program.

Signature: Production Note:
Signature removed prior to publication.

Date: January 6th, 2021

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, compared 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 platform, Android has long been the most targeted operating system suffering from attacks. To keep pace with the cutting-edge anti-malware 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. This thesis focuses on both statically reverse engineering the samples and dissecting the programs to understand their internal logic and find the similar features that could be used to assist security analysts, and dynamically monitor their behaviors in emulators. From public and private sources, 2380 samples of trojans from 20 (sub)families have been collected. As a result of the analysis, a lucid overview and improved apprehension of Android trojans are provided. 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.

Acknowledgement

Throughout my research life pursuing a master's degree, I have received a lot of support from a wide range of people, both online and offline, some are friends or relatives while some have never been in contact.

First, I would like to sincerely thank my supervisor, Sui, whose expertise and resources are significant in both giving recommendations for revising the published articles as well as this thesis and informing possible journals and/or conferences to submit potentially publishable drafts. His insightful suggestions make my work more coherent and professional.

I would like to acknowledge my colleagues Guanqin Zhang, Zexin Zhong and Yanxin Zhang, who have helped me with reviewing drafts. I want to thank them all for their advice on perfecting the work to a higher level.

Also, I would like to thank some of the security professionals I have followed from Twitter, who are very resourceful and let me know many useful tools for both reverse engineering and analysis. Among them, @xabc0 a.k.a. Ahmet Bilal Can is the one that leads me into the world of banking trojans and malware analysis by his posts. @pr3wtd a.k.a. Witold Precikowski is the one who founded Apkdetect [1], and helped me with answering my questions when I encountered some weird code in the reverse-engineered source code. @LukasStefanko a.k.a. Lukas Stefanko also provided me with good ideas when I had some confusion when analyzing some malicious payloads of Bankbot Anubis.

In addition, I would like to thank my parents. I appreciate their encouragement when I felt frustrated or disappointed during my research. Their support made me through the plight. Moreover, this dissertation would not be completed without the company and communication from my friends, Wei Liu and Zhe, who have given me helpful suggestions and provided informative discussions based on their experiences of pursuing a PhD degree. Besides, the entertainment with them refreshed my mind from intensive research.

Finally, I would like to thank Bilibili as well as Youtube for providing records of great courses as well as amusing content for cheering and charging me up during my off-work hours.

Contents

1	Introduction	7
1.1	Android operating systems	7
1.2	Android malware and Android banking trojan	9
1.3	APK Structure	14
1.4	Research Objectives	15
1.5	Thesis Organization	16
2	Literature Review	17
2.1	Android Trojan Evolution	17
2.2	Android Malware Families Analysis	17
2.3	Android malware detection	18
2.3.1	Static Approaches	19
2.3.2	Dynamic Approaches and Hybrid Approaches	20
2.4	Android Malware Clustering and Classification	21
3	Deep Analysis of Recent Android Trojans	22
3.1	Data Collection and Extraction	22
3.2	Attributes of Recent Android Trojans	27
4	Analysis of the Bankbot Anubis family	40
4.1	Anubis’s Approach	41
4.1.1	Phase 1: Mobile Devices Cyber Attack	41
4.1.2	Phase 2: Privilege Escalation	42
4.1.3	Phase 3: C&C Server	44
4.1.4	Phase 4: Decrypting the encrypted	47
4.2	Data and Analysis	47
4.2.1	Anubis Roadmap	48
4.2.2	Development of Encryption	50
4.2.3	AndroidManifest	50
4.2.4	Hard-coded API classes and methods	54
4.2.5	Opcode Sequences	57
5	An Observation for Android Malware Detection	59

6 Conclusion and Future Work	64
Appendix A Publication List	78
Appendix B Figures	78

List of Figures

1	Market shares of different operating systems for mobile phones[2]	8
2	Number of applications available on Google’s official store[3]	8
3	Percentage of Android devices (i.e. Cumulative distribution in the screenshot) that the application could run on w.r.t the minimum SDK version (i.e. Android platform version in the screenshot)	8
4	Distribution of Android versions based on the cumulative distribution data . . .	9
5	Banking trojans detected by Kaspersky from 2015 to 2019	14
6	Dissected file structure of an APK file	15
7	The screenshot that the hashtags added by an analyst’s comment	23
8	The screenshot of the first five samples when searching with ”tag: trojan” on Koodous platform	23
9	The workflow of generating family names of Android trojans through utilizing the Koodous platform	24
10	The captured brief information that matches the family name (e.g. Anubis) input in the search box in Apkdetect	25
11	The captured detailed information that either matches or is somehow relevant to the family name (e.g. Anubis) input in the search box in Apkdetect	25
12	The workflow of using Apkdetect for generating configuration information of the samples	27
13	Anubis Attack Procedures	41
14	Infection Process	42
15	Code Injection in Forged App	43
16	Interface of Anubis Accessibility Extraction	44
17	Anubis C&C Server Model	45
18	The breakdown of the Anubis samples regarding configurations and versions . . .	48
19	Emergence of different versions of Anubis from January 2018 to July 2019	49
20	Top 4 frequent API classes used by four versions of Anubis	56
21	The name of the newly implemented module for new versions	57

22	Top 3 frequent (including overlapping) documented Opcode sequences of four versions of Anubis	58
23	The workflow of identifying suspicious and unsuspecting APKs	61
24	The AUC-ROC curve of the results	63
B.1	”3458” in Rotexy	78
B.2	”393838” in Rotexy	78
B.3	How Anubis displays toast	79
B.4	How Anubis tailors the text based on the language	79
B.5	Hard-coded ”Enable access for” in different languages	79
B.6	Base64 decoded image from the source code of most higher versions of Anubis . .	80
B.7	A sharper image from double Base64-decoded source code snippets of an Anubis dropper	80

List of Tables

1	Differences between different types of malware	10
2	The timeline of 20 (taking sub-families into account) Android trojans in the collection	26
3	The (abbreviated) dangerous and sensitive permissions requested by recent Android trojans	29
4	An overview of recent Android trojan (sub)families in regard to the categories of permissions requested	30
5	The (abbreviated) intents that are of interest to malware authors to trigger trojans	32
6	An overview of recent Android trojan (sub)families in regard to the categories of intent actions used	33
7	An overview of recent Android trojan (sub)families in regard to the anti-analysis techniques	35
8	An overview of recent Android trojan (sub)families in regard to the persistence techniques	37
9	An overview of recent Android trojan (sub)families in regard to the communication methods with C&C	39
10	Frequently used permissions by Anubis	52
11	Shared intent filters by all versions of Anubis	53
12	API classes abused by different versions of Anubis	55

13	Top 3 documented API methods that are the most different in terms of their average frequency	56
14	Seven types of Dalvik Opcodes	57
15	The statistics of the APK samples in the dataset	59
16	The confusion matrix of the results	62