International Journal of Engineering & Technology, 7 (4.15) (2018) 59-62

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

A New Mobile Malware Classification for Audio Exploitation

Muhamad Nur Arif[1], Azreena Abu Bakar[1], Madihah Mohd Saudi[1,2]*

[1]*Faculty of Science and Technology, Universiti Sains Islam Malaysia (USIM), Malaysia*
[2]*CyberSecurity and Systems Research Unit, Islamic Science Institute (ISI), Universiti Sains Islam Malaysia (USIM), Malaysia*
**Corresponding author E-mail: madihah@usim.edu.my*

Abstract

Rapid growth and usage of Android smartphones worldwide have attracted many attackers to exploit them. Currently, the attackers used mobile malware to attack victims' smartphones to steal confidential information such as username and password. The attacks are also motivated based on profit and money. The attacks come in different ways, such as via audio, image, GPS location, SMS and call logs in the smartphones. Hence, this paper presents a new mobile malware classification for audio exploitation. This classification is beneficial as an input or database to detect the mobile malware attacks. System calls and permissions for audio exploitation have been extracted by using static and dynamic analyses using open source tools and freeware in a controlled lab environment. The testing was conducted by using Drebin dataset as the training dataset and 500 anonymous apps from Google Play store as the testing dataset. The experiment results showed that 2% suspicious malicious apps matched with the proposed classification. The finding of this paper can be used as guidance and reference for other researchers with the same interest.

Keywords: *Audio Exploitation; Android Smartphone; Malicious Apps; Mobile Malware.*

1. Introduction

With the proliferation of mobile devices, there is an increasing threat from mobile malware such as worm, Trojan, spyware, adware, virus, spam and other malicious software. Exploited Android devices by malware can be manipulated such as to retrieve any crucial information like background process and services on the device. Additionally, the device also can be used by the attacker to record audio, send short messages service, make calls, execute any malicious command and delete browser history [1]. There is 0.15% of devices infected with malware in 2014, and some of them can steal bank account information via reviewing emails in Gmail [2]. Furthermore, there is a Trojan that specializes in accessing audio data and steal the audio data without the user's knowledge [3]. The Trojan uses a sensitive sensor which is a context sensitive reference to monitor the Audio Flinger. From that audio service, the Trojan changes the media data from the kernel service. This Trojan can block other application from accessing audio data when the call is being used. After that, the controller is alerted from the system when the sensitive call is made.

Therefore, the objective of this paper is to develop a new mobile malware classification for audio exploitation based on system call and permission. Based on the experiment conducted, there are 32 patterns of classification for the audio exploitation and 10 out of 500 mobile apps matched with the proposed classification. The scope of this paper is on Android smart phone only. This is due to the worldwide usage of Android with 86.1% in the market and Android has become the most targeted smartphone by the attackers in the world [4-5].

Malware can be referred as virus, worm, Trojan, botnet, adware and spyware. There are many techniques such dynamic analysis or static analysis to analyse the malware. For dynamic analysis, the malware sample is executed in a controlled environment to see the payload [6]. As for static analysis, the malware dataset is being reverse engineered, and the source code is being analysed to see the command and payload inside the source code [7]. Examples of works that are related to malware analysis are research work by [8-13]. Each of the static and dynamic analyses has it owns strength, but under certain condition where the malwares payload is hard to be analysed, both analyses need to be combined. This is known as hybrid analysis where it combines static and dynamic analyses, which has been used by [8, 14]. The strength of the hybrid analysis is both conditions can be monitored for optimum result. Therefore, our paper has implemented this technique for the experiment conducted.

The rest of this paper is written as follows. Section 2 presents the methodology used in this paper. Section 2 presents the experimental result and Section 4 concludes this paper and discusses the future work.

2. Methodology

The overall experiment for malware analysis processes is summarized in Figure 1. It is beneficial to extract the system call and permission from the mobile apps.

There are two types of dataset which are training and testing. Drebin dataset with a total of 5560 was used as the training dataset to produce the pattern of the classification, while the testing dataset was taken from 500 anonymous mobile apps from Google Play Store for evaluation. The experiment was conducted in a controlled environment, where no outgoing network is allowed to avoid malware spreads. 80% of the software used are open source, which includes SDK tool for dynamic analysis, Genymotion for android emulator, apk tool to decompile apk resource file into a folder and strace to capture system call behaviour. During the experiment, hybrid analysis that combines dynamic and static analyses was conducted. There is no standard sequence to run dynamic or static analysis. As for this experiment, the dynamic analysis was conducted earlier due to the time consuming and then followed by static analysis to verify any hidden apps payload. The dynamic analysis is conducted as follows:

- Run dataset in Genymotion
- The apps processes were identified
- The system calls ran were traced
- The system calls were captured and documented

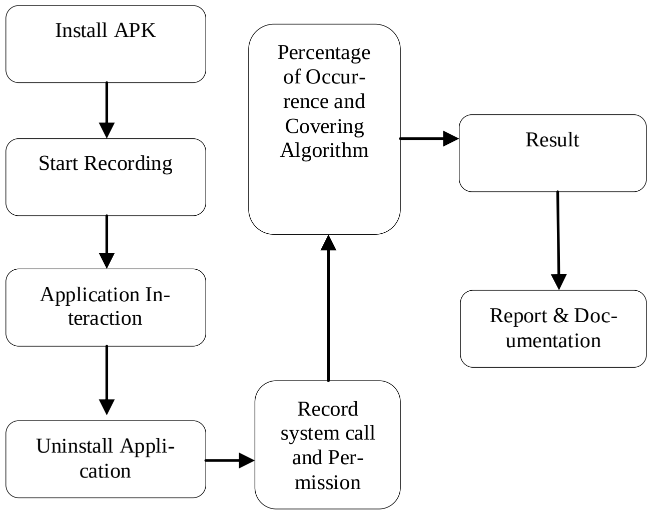

Fig. 1: Malware Analysis Processes

While for static analysis, it is used to capture the permission that runs in the apps. The extracted permissions were then being classified using the covering algorithm.

- Installed Dexplorer in the Genymotion
- Run dataset in Genymotion
- The permissions were traced
- The permissions were captured and documented.

Then, after permissions and system calls extraction were completed, percentage of occurrence was applied then followed by covering algorithm. Percentage occurrence was applied to calculate the total number of system calls and permissions existence from the extracted samples. The most used system calls and permissions were identified based on the frequency and then the patterns were developed using the covering algorithm. The covering algorithm is summarized as follows [14]:

- If the system calls and permissions are covered by the set rule, then remove it and continue until all the system calls and permissions were covered.
- The idea is to include as many instances of the desired permission and system call as possible and exclude as many instances of other features as possible.

3. Findings

Thousands of system calls and permissions were extracted, but the focus of this paper is only on those that generate bad activities for audio exploitation. There are 58 system calls and 41 permissions out of 5560 training dataset that are related to audio exploitation. The extracted system calls and permissions are displayed in Table 1 and Table 2.

Table 1: System Calls Extraction and Representation

Nominal Data	Syscall	Nominal Data	Syscall
c1	clock_gettime()	c30	socket()
c2	epoll_wait()	c31	bind()
c3	recvfrom()	c32	getsockname()
c4	sendto()	c33	unlinkat()
c5	futex()	c34	madvise()
c6	gettimeofday()	c35	pwrite64()
c7	writev()	c36	setsockopt()
c8	getuid32()	c37	lseek()
c9	read()	c38	nanosleep()
c10	ioctl()	c39	getrlimit()
c11	write()	c40	brk()
c12	close()	c41	fchown32()
c13	open()	c42	getpid()
c14	mmap2()	c43	gettid()
c15	mprotect()	c44	lstat64()
c16	dup()	c45	recvmsg()
c17	fcntl64()	c46	recv()
c18	epoll_ctl()	c47	stat64()
c19	munmap()	c48	sigprocmask()
c20	pread()	c49	select()
c21	sched_yield()	c50	umask()
c22	getsockopt()	c51	getpaid()
c23	clone()	c52	pread64()
c24	access()	c53	rename()
c25	fstat64()	c54	fdatasync()
c26	chmod()	c55	mkdir()
c27	fsync()	c56	uname()
c28	connect()	c57	rt_sigreturn()
c29	sendmsg()	c58	_llseek()

Table 2: Permissions Extraction and Representation

Nominal Data	Permission	Nominal Data	Permission
pr1	Access_Course_Location	pr 22	Install_Packages
pr2	Access_Fine_Location	pr 23	Install_Shortcut
pr 3	Access_Gps	pr 24	Internet
pr 4	Access_Location_Extra_Commands	pr 25	Kill_Background_Process
pr 5	Access_Network_State	pr 26	Modify_Audio_Setting
pr 6	Access_Wifi_State	pr 27	Read_Calendar
pr 7	Battery_Stat	pr 28	Read_Call_Log
pr 8	Bluetooth	pr 29	Read_Contact
pr 9	Bluetooth_Admin	pr 30	Read_External_Storage
pr 10	Call_Phone	pr 31	Read_Logs
pr 11	Camera	pr 32	Read_Phone_State
pr 12	Change_Network_State	pr 33	Read_Settings
pr 13	Change_Wifi_Multicast_State	pr 34	Read_Sms
pr 14	Change_Wifi_State	pr 35	Receive_Boot_Complete
pr 15	Clear_App_Cache	pr 36	Receive_Mms
pr 16	Control_Location_Updates	pr 37	Receive_Sms
pr 17	Delete_Packages	pr 38	Record_Audio
pr 18	Disable_Keyguard	pr 39	Restart_Packages
pr 19	Expand_Status_Bar	pr 40	Write_External_Storage
pr 20	Get_Accounts	pr 41	Write_Settings
pr 21	Get_Tasks		

The system calls and permissions are presented in nominal data to ease and fasten the pattern formation. Based on the pattern matching conducted, 32 patterns of system call and permission classifications were developed as displayed in Table 3. These classifications need a high analysis expertise to ensure the pattern produced is beneficial to detect any mobile malware attacks for audio exploitation. Furthermore, the pattern is formed based on the most predicted features that might lead to the audio exploitation and function analysis for each feature of the system calls and permissions.

Table 3: System Calls and Permission Patterns

Pattern Representation	Pattern
Pattern1	pr10 +pr19+pr29+pr31+pr39+ c1+c2+c9+c10+c11 +c7
Pattern 2	pr10 +pr19+pr29+pr31+pr39+ c1+c2+c9+c10+c11 +c7+c28
Pattern 3	pr10 +pr19+pr29+pr31+pr39+ c1+c2+c9+c10+c11 + c7+c28+c30
Pattern 4	pr10 +pr19+pr29+pr31+pr39+ c1+c2+c9+c10+c11 +c7+c28+c30+c31
Pattern 5	pr10 +pr19+pr29+pr31+p39+ c1+c2+c9+c10+c11 +c7+c28+c30+c31+c46
Pattern 6	pr10 +pr19+pr29+pr31+pr39+ c1+c2+c9+c10+c11+c7+c28+c30+c46
Pattern 7	pr10 +pr19+pr29+pr31+pr39+ c1+c2+c9+c10+c11+c7+c28+c31
Pattern 8	pr10 +pr19+pr29+pr31+pr39+ c1+c2+c9+c10+c11+c7+c28+c31+c46
Pattern 9	pr10 +pr19+pr29+pr31+pr39+ c1+c2+c9+c10+c11+c7+c28+c46
Pattern 10	pr10 +pr19+pr29+pr31+pr39+ c1+c2+c9+c10+c11+c7+c30
Pattern 11	pr10 +pr19+pr29+pr31+pr39+ c1+c2+c9+c10+c11+c7+c30+c31
Pattern 12	pr10 +pr19+pr29+pr31+pr39+ c1+c2+c9+c10+c11+c7+c30+c31+c46
Pattern 13	pr10 +pr19+pr29+pr31+pr39+ c1+c2+c9+c10+c11+c7+c30+c46
Pattern 14	pr10 +pr19+pr29+pr31+pr39+ c1+c2+c9+c10+c11+c7+c31
Pattern 15	pr10 +pr19+pr29+pr31+pr39+ c1+c2+c9+c10+c11+c7+c31+c46
Pattern 16	pr10 +pr19+pr29+pr31+pr39+ c1+c2+c9+c10+c11+c7+c46
Pattern 17	pr10 +pr19+pr29+pr31+pr39+ c1+c2+c9+c10+c11+c28
Pattern 18	pr10 +pr19+pr29+pr31+pr39+ c1+c2+c9+c10+c11+c28+c30
Pattern 19	pr10 +pr19+pr29+pr31+pr39+ c1+c2+c9+c10+c11+c28+c30+c31
Pattern 20	pr10 +pr19+pr29+pr31+pr39+ c1+c2+c9+c10+c11+c28+c30+c31+c46
Pattern 21	pr10 +pr19+pr29+pr31+pr39+ c1+c2+c9+c10+c11+c28+c30+c46
Pattern 22	pr10 +pr19+pr29+pr31+pr39+ c1+c2+c9+c10+c11+c28+c31
Pattern 23	pr10 +pr19+pr29+pr31+pr39+ c1+c2+c9+c10+c11+c28+c31+c46
Pattern 24	pr10 +pr19+pr29+pr31+pr39+ c1+c2+c9+c10+c11+c28+c46
Pattern 25	pr10 +pr19+pr29+pr31+pr39+ c1+c2+c9+c10+c11+c30
Pattern 26	pr10 +pr19+pr29+pr31+pr39+ c1+c2+c9+c10+c11+c30+c31
Pattern 27	pr10 +pr19+pr29+pr31+pr39+ c1+c2+c9+c10+c11+c30+c31+c46
Pattern 28	pr10 +pr19+pr29+pr31+pr39+ c1+c2+c9+c10+c11+c30+c46
Pattern 29	pr10 +pr19+pr29+pr31+pr39+ c1+c2+c9+c10+c11+c31
Pattern 30	pr10 +pr19+pr29+pr31+pr39+ c1+c2+c9+c10+c11+c31+c46
Pattern 31	pr10 +pr19+pr29+pr31+pr39+ c1+c2+c9+c10+c11+c46
Pattern 32	pr10 +pr19+pr29+pr31+pr39+ c1+c2+c9+c10+c11

To ensure the developed classifications in this paper are useful, 500 anonymous mobile apps were downloaded from Google Play Store for testing and evaluation. As a result, 2% of the mobile apps were identified as matched with the proposed classifications. The evaluation result is summarized in Table 4, where the matched mobile apps with the proposed patterns came from different categories. Based on the evaluation result, it can be concluded that even genuine mobile apps have potential for audio exploitation by the attackers. Therefore, end users must take extra precaution about the privacy risk of the mobile apps when downloading it. Hence, as a prevention step from being infected or exploited via mobile apps, it is highly recommended for end users to scan with anti-virus for any mobile apps, prior downloading it.

Table 4: Percentage of Pattern Matched

Pattern	Google Play	Category	Percentage
Pattern 1	4	Social, Simulator, Tool, Communication	0.8%
Pattern 2	2	Music Game, Communication	0.4%
Pattern 3	1	Video	0.2%
Pattern 4	1	Simulator	0.2%
Pattern 5	1	Music Game	0.2%
Pattern 6	1	Communication	0.2%

4. Conclusion

As a conclusion, this paper has successfully developed 32 new mobile malware classifications for audio exploitation based on system call and permission features. The evaluation result shows that even though the mobile apps were downloaded from the trusted source, end users must remember that there is a possibility that the genuine mobile apps can be exploited by the malwares and audio is one of the surveillance features in smartphone that is most targeted by many attackers. The work in this paper is a part of an ongoing project in combating mobile malware attacks. The results of this research paper will be used as an input for mobile malware detection model. Researchers with the same interest could use this paper as a guidance and reference in doing their research.

Acknowledgement

The authors would like to express their gratitude to Ministry of Higher Education (MOHE), Malaysia and Universiti Sains Islam Malaysia (USIM) for the support and facilities provided. This project is funded under grant: USIM/FRGS/FST/32/50114.

References

[1] Soriano, A.," A. Software, Avast Blog mobile malware", (2016), https://blog.avast.com/topic/mobile-malware.

[2] Lemos, R, "New malware threats emerge on mobile platforms", (2016), http://www.eweek.com/security/new-malware-threats-emerge-on-mobile-platforms-studies-find.

[3] Schlegel, R. Zhang, K. & Zhou, X. "Soundcomber: A stealthy and context-aware sound Trojan for smartphones", Proceedings of the 18th Annual Network and Distributed System Security Symposium, (2011), pp. 17–33.

[4] Alcatel-Lucent, "Mobile malware: A network view", (2015), https://www.blackhat.com/docs/ldn-15/materials/london-15-McNamee-Mobile-Malware-A-Network-View-wp.pdf.

[5] Gartner, "Gartner says worldwide sales of smartphones grew 9 percent in first quarter of 2017", (2017), http://www.gartner.com/newsroom/id/3725117.

[6] Junaid, M. Donggang, L. & David, K. (2016), Dexteroid: Detecting malicious behaviors in android apps using reverse-engineered life cycle models. Computers and Security, 59, 92–117.

[7] Ping, W. & Wang, Y.-S. (2015), Malware behavioural detection and vaccine development by using a support vector model classifier. Journal of Computer and System Sciences, 81(6), 1012–1026.

[8] Lindorfer, M., Neugschwandtner, M., Weichselbaum, L. Fratantonio, Y., van der Veen, V. & Platzer, C. (2014), ANDRUBIS -- 1,000,000 apps later: A view on current android malware behaviors. Proceedings of the 3rd International Workshop on Building Analysis Datasets and Gathering Experience Returns for Security, pp. 3–17.

[9] Feizollah, A., Anuar, N. B., Salleh, R. & Abdul Wahab, A. W. (2015), A review on feature selection in mobile malware detection. Digital Investigation, 13, 22–37.
[10] Hashim, H. A.-B., Saudi, M. M., & Basir, N. "A systematic review analysis of root exploitation for mobile botnet detection", Proceedings of the 1st International Conference on Communication and Computer Engineering, (2015), pp. 925-938.
[11] Bhatt, M. S., Patel, H. & Kariya, S. (2015), A survey permission based mobile malware detection. International Journal of Computer Technology and Applications, 6(5), 852–856.
[12] Karim, A., Salleh, R. & Khan, M. K. (2007), SMARTbot: A behavioral analysis framework augmented with machine learning to identify mobile botnet applications. PLoS One, 11(3), p. e0150077.
[13] Wu, S., Wang, P. Li, X. & Zhang, Y. (2016), Effective detection of android malware based on the usage of data flow APIs and machine learning. Information and Software Technologies, 75, 17–256.
[14] Saudi, M. M. & Husainiamer, M. A. (2017), Mobile malware classification via system calls and permission for GPS exploitation. International Journal of Advanced Computer Science and Applications, 8(6), 277-283.