International Journal on Recent and Innovation Trends in Computing and Communication
Volume: 5 Issue: 6

ISSN: 2321-8169
1111 – 1117

A Preface on Android Malware: Taxonomy, Techniques and Tools

Rashmi Rupendra Chouhan
MCA Department,
Sarvajanik College of Engineering and Technology,
Surat, India
rashmi.chouhan@scet.c.in

Alpa Kavin Shah
MCA Department,
Sarvajanik College of Engineering and Technology,
Surat, India
alpa.shah@scet.ac.in

***Abstract*—** Android OS has an open architecture and provides Application Programming Interface (APIs)enabling it to earn a huge market share and interest in the developer community. Android has become the most well-liked smartphone Operating System in current digital world. With the increased popularity of Android devices and open source features the malware threat has also increased. Android mobile operating system applications have right to use to a lot of personal information when granted certain permissions at the time of app installation. Apps can have access to the contacts, e-mails, can track the physical location, access gallery, and others. Due to this reason, Android users are looking for better security solutions to protect their smartphones from malicious actions. To cope up with this exponential growth of mobile users and malware threats, we have presented and analyzed Android malware trends till 2016 and continuous growth in malware till first quarter of May 2017. To clinch, we have summarized Android malware detection techniques.

Keywords*-Android malware, static analysis, dynamic analysis*

*****

I. INTRODUCTION

The rate of smartphone adoption has increased massively since last decade. Smartphone's have become the integral part of our lives as they provide services such as social networking, banking, e-commerce trade, surfing, online videos and music and others. Modern smartphones contain many features like GPS, Wi-Fi, video calls, Bluetooth, Sensors, Cameras and players. There comes a need for security for the smartphones with all these features. Android applications published as of February 2017 in Google Play store has over 2.7 million [1] and more than 65 billion times of apps have been downloaded as of May 2016 [2]. Millions of Android devices are at risk of being infected with malicious program. Some of the Android malware listed by detected by Quick Heal in 2016 are Android.Smsreg.DA, Android.Airpush.G, Android.Agent.TN, Android.Ztorg.A, Android.Rooter.E, Android.Jiagu.A, Android.Rootnik.D, Android.Downloader.D, Android.Triada.F, Android.Loki.Am. Other malwares are AccuTrack, Badnews, BeanBot, Cellspy, DroidDeluxe, DroidDream, to name a few. One of the latest Android Malware found in the month of May 2017 according to security researchers at Check Point is Judy malware. It affected up to 36 million Android users, which was found in more than 50 apps on the Google Play store. Android samples received by Quick Heal Threat Research Labs for the year 2016 is depicted below in Fig. 1

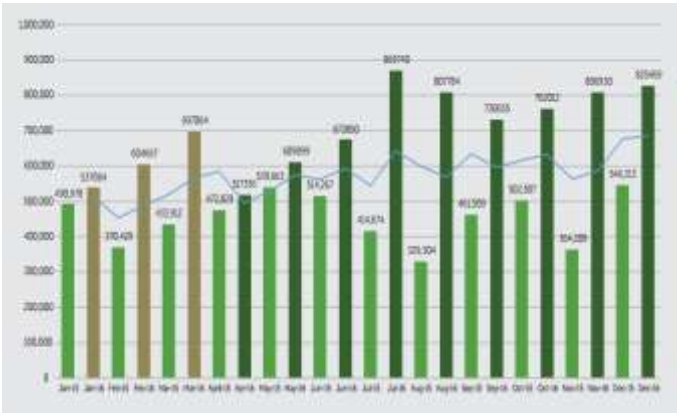

Fig. 1 Android Threats Analysis by Quick Heal Threat Research Labs for the year 2015 -2016[34]

During the first quarter of this year Security researchers from antivirus software firm ***G Data*** have revealed that more than 750,000 new malicious apps are found, which estimates the total number will grow up to a staggering 3.5 million by the end of 2017.The report further described the problem is specifically widespread amid devices from third-party phone makers where software updates that tend to receive software updates less frequently and fewer times with significant delays. To cite some more examples, G Data researchers also note that in comparison to this year, they identified 3.2 and 2.3 million infected apps in 2016 and 2015 respectively. Internet connectivity and availability of personal information such as messages, contacts, browsing history, social network access and banking credentials has attracted the attention of malware developers towards the mobile devices in Android. Android malware such as Spyware, SMS Trojans, Botnets, Adwares,

 ISSN: 2321-8169
1111 – 1117

Backdoors attack exploits reported exponential rise apart from being distributed from the secure Google.

II. ANDROID MALWARES

Ever since worms, viruses and Trojan horses were counted, the number of new malware has increased. We now present a most common of Malwares found in Android.

A. Trojans

Trojans appear as a useful app to user, the malicious code can be embedded into any android application and will run silently behind an otherwise harmless app. Such apps get right to use to the messages, browsing history, contacts and device IMEI numbers etc. of victim's device and steal this information without the consent of user [3]. FakeNetflix [4] is the malware uses a trojan that provide user interface similar to original Netflix app. When we click on the app icon, it disappears and background process starts, the hacker has gained access to your device. This permits them to use your camera, contact, microphone, text messages. Ackposts is a trojan that steals contact information from the infected device and uploads them to a remote server. Acnetdoor is also a Trojan that opens a backdoor on the compromised device and sends the IP address to a remote server. Adsms this is a Trojan which send SMS messages containing the download link.

B. Backdoors

Backdoors grants root privileges to the malwares and make easy them to hide from antiviruses. Exploid, Rageagainstthecage (RATC) and Zimperlich are the three root exploits which gain complete control of infected device [5]. AnServer/Answerbot is a backdoor and capable to steal personal information which will be uploaded to a remote server afterwards.

C. Worms

It creates copies of itself and distributes them over the network. For example, Bluetooth worms distribute malware to paired devices through the Bluetooth network. *Selfmite* is a SMS worm it uses a legal advertising platform and pay-per-install for monetization and spread through SMS messages.

D. Spyware

Spyware apps monitor the user's confidential information such as contact, bank information, messages, location etc. for some objectionable cost to the attacker who installed that software on victim's device.

Smack the spyware is based on Open Source XMPP client library for instant messaging and presence. It has following features like Upload short messages, phone records, contact information, short messages, GPS location and date, hide its icon and intercepts specified short messages. *Tracer* is *a* Commercial Spyware.

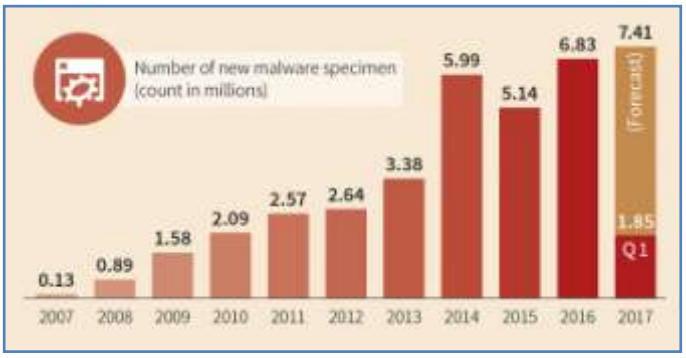

Fig:2 Malware Trends 2017 [35]

E. Botnets

A Botnet is a network of infected Android devices, where the network is used by the malware to spread. Botmaster, a remote server, manages the Botnet through the C&C network. Geinimi [6] is one of the Android botnets.

F. Ransomwares

Ransomware is a malware that prevent the user from accessing their data on device and demands money in exchange for some hacked information. Because it's a new way of extracting money, many fall victim to it. Simplocker this is the first-ever Androidransom ware that encrypts files. XBot is the relatively recent malware that can steal Android users' personal data and banking credentials by leveraging a phishing hoax. To perform this job, it imitates Google Play payment screen and Login interfaces for several e-banking applications. Another malicious functionality is remote data encryption – Xbot can encode files stored on the SD card.

G. Riskwares

Riskwares are the legitimate programs exploited by the malicious users to lessen the performance of device or harm the data e.g., delete copy or modify etc. [7]. Riskware can include so many programs to access legal data like Remote administration utilities, Dialer programs, File downloader's, Password management utilities, Internet server services. Android/SmsPay may potentially be adware or may infect your personal information like contacts, e-mail address. Another example of riskware is Android:Riskware-A.
Malware Trends 2017 are shown in Fig. 2

III. ANDROID MALWARE DETECTION

A. Static Approach

Static analysis approach consists of analyzing the executable file without viewing or executing the actual instructions. Instructions can be fetched from Android manifest file or Java bytecode. Static analysis is more efficient and informative, particularly for highly obfuscated code.

 ISSN: 2321-8169
1111 – 1117

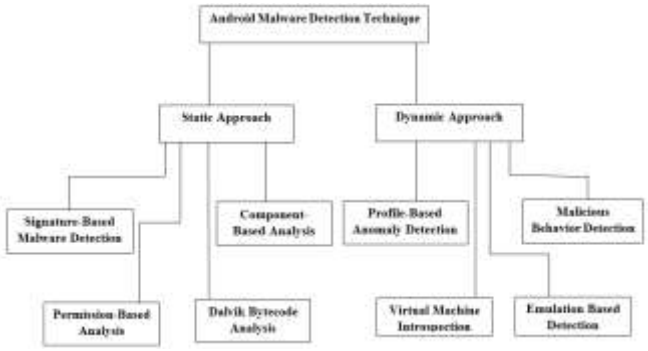

Fig: 2 Types of Android Malware Detection Techniques

1. *Signature-Based Malware Detection*

Signature based malware detection methods extracts the syntactic patterns and creates a unique signature. A program is considered as a malware if its signature matches with the list of known malicious files. It also looks within files to find signatures of malicious code. This detection scheme works on the assumption that malware can be described through patterns also called signatures [8]. The drawback of Signature-based method is that it derives signature byte patterns from known malware, so these byte patterns are also commonly known in advance and it also fails against the unseen variants of already existing and known malware. It wants immediate update of malware variants as soon as they are detected. *AndroSimilar* proposed by Faruki*et al*. [9], a robust approach to detect the unknown variants of existing malwares that are usually generated by code obfuscation and repackaging techniques. Code obfuscate techniques can be used to provide security to Android code by shrinking and optimize your code. Obfuscated code can be more difficult for other people to reverse engineer. Tools can be used to renames classes, fields, and methods with semantically obscure names and removes unused code. IBM ProGaurd is an Obfuscate tool. Repackaging techniques are the new version of an original app maliciously modified by an attacker. The attacker disassembles the original app executable (apk file), changes or inserts new functionalities, reassembling the modified apk afterward.

2. *Component-Based Analysis*

To perform detailed app-security assessment or analysis, an app can be divided into parts such as AndroidManifet.xml, bytecode and resource to extract important content. Manifest stores important meta-data about the app such as list of the components (i.e., activities, intents, services, receivers etc.) and required permissions to execute the app. Bytecode and component can be examine to identify the vulnerabilities. [10], [11], [12].

3. *Permission-Based Analysis*

Android security model always ask permission to access a sensitive resource model. It does not allow any application to affects user security. For declaring the malware app just identifying the prohibited permission request is not sufficient, but still the permissions mapping requested and used permissions is an important risk identification technique. Sanz Borja et al. [13] used _uses−permission_ and _uses−features_ tags present in AndroidManifest.xml to identify malware apps.

4. *Dalvik Bytecode Analysis*

Android app, written in Java language is compiled to Java Bytecode and then to Dalvik Bytecode that runs under newly created runtime Dalvik Virtual Machine (DVM). Bytecode analysis helps to examine the app behavior. Control and data flow analysis detect the dangerous functionalities performed by malicious apps.

A static analyzer SCANDAL developed by Jinyung Kim et al. [14] that analysis the dalvik byte code of applications and detects the privacy leakage in app. SCANDAL a sound and automatic static analyzer for identifying privacy leaks in Android apps. As mentioned in [15] the ComDroid is a tool that performs function like flow sensitive, intra-procedural static analysis with limited inter-procedural analysis of Dalvik bytecode programs. This was created to evaluate the communication between Android apps through intents, the Android equivalent of events and to find potential security vulnerabilities in the communication patterns of applications. ComDroid tool is also used as a component in another analysis tool called Stowaway that analyses API calls in applications to determine if they are over-privileged. It uses *Dedexer*tool [16] to disassemble the dex files in the app.

B. Dynamic Approach

Dynamic analysis is based on code execution. Dynamic analysis deals with dynamic code loading and system calls that are gathered while the application is running.

1. *Profile-Based Anomaly Detection*

Malicious apps sometimes may generate Denial of Service (DoS) attacks by over utilizing the constrained hardware resources. Range of parameters such as network traffic pattern, battery usage CPU usage, memory utilization statistics and malware apps are collected from the Android subsystem. App-security, Automatic permissions and assessment solutions can analyze the components using their definition and bytecode interaction to identify the vulnerabilities [17], [18], [19].

2. *Malicious Behavior Detection*

Specific malicious behaviors can be accurately detected by monitoring the particular features of interest [20]–[22], like sending SMS/emails, sensitive data leakage, voice calls without user consent.

3. Virtual Machine Introspection

The shortcoming of app behavior monitoring from an emulator (VM) is that an emulator is susceptible against the malicious app which defeats the analysis purpose. So as a solution Virtual Machine Introspection approaches can be employed to detect app behavior by observing the activities out of the emulator [23].

4. Emulation Based Detection

Android dynamic analysis platform DroidScope was presented by Yan et al. [24], based on Virtual Machine Introspection. The antimalware detect the occurrence of malwares because both of them reside in the same execution environment. DroidScope monitors the complete operating system by residing out of the execution environment and thus have more privileges than the malware programs. It monitors the Dalvik semantics thus the privilege increase attacks on kernel can also be detected easily. DroidDream and DroidKungFu[25] were detected with QEMU technique.

Android Application Sandbox (AASandbox) proposed Blaising et al. [26] detect the suspicious apps by performing both static and dynamic analysis on them. It first extracts the .dex file into human readable form and then performs static analysis on app. Then it examines the low level interactions with system by execution of apps in isolated sandbox environment. Actions of apps are restricted to sandbox due to security policy and it does not affect the data on device. It cannot detect the new malware types.

IV. COMPARATIVE ANALYSIS OF THE MALWARES TECHNIQUES

In previous sections, we have discussed in detail the malwares and their detection techniques. Antimalware techniques are also specified with detection technique. Table 1 shows malware detection through static analysis along with some basic details. Table 2 shows malware detection through dynamic analysis along with some details.

Through this analysis we confront the benefits and limitations of various techniques. Also we highlight multiple tools available under each technique.

Table 1 Malware Analysis through Static Approach

Technique	Tools	Benefits	Limitations	Author
Signature Based	AndroSimilar[9]	1) Effective against code obfuscation and repackaging.	1) Signature database has limited entries 2) It only detects known malware variants	Farukiet al.
	YARA[27]	1) It's a open source tool 2) Cut the cost of Reverse Engineering 3) Compatible with Perl-based Regular Expressions 4) Analyze the suspected files/directories and match strings as is defined in the YARA rules with the file.	1) YARA tool only does pattern/string/signature for matching the detecting malware is available. 2) It may match the signature with files in which the specified criteria exist and yet do not possess the same semantics as expressed in the original file. 3) YARA signatures is that they can be easily analyzed in the face of changing codebases. A malware author can change his/her code to suit an ever-shifting set of goals, and keeping up with these changes is particularly challenging.	Victor Alvarez
	DroidAnalytics [28]	1) Effective against mutations and repackaged apps. 2) Malware at op-code level get associated 3) Simple malware and dynamic payload tracking. 4) Detect dynamic malware payloads.	1) It may classify legitimate apps as malicious. 2) Level 2 signatures are classified as malwares and also used by legitimate apps. 3) It cannot detect unknown malware types.	M. Zheng, M. Sun, and J. C. S. Lui

Component Based	ComDroid [15]	1)Generates warnings about threats.	1) Does not validate the existence of malware 2) Users investigate warnings manually	Erika Chin, Adrienne Porter Felt,KateGreenwood,David Wagner
Permission Based Analysis	PUMA[13]	1) High detection rate	1) High false positive rate 2) Not adequate for efficient malware detection	B. Sanz, I. Santos, C. Laorden, X. Ugarte-Pedrero, P. G. Bringas, and G. Álvarez
	Stowaway [31]	1) It notifies about the over privileged apps.	1) Complex reflective calls Cannot be resolved	Adrienne Porter Felt, Dawn Song, David Wagner, Steve Hanna
Dalvik Bytecode Analysis	SCANDAL [14]	1) Privacy leakage of data can be preserved 2) Dalvik bytecode is mostly available. 3) Does not need reverse engineering tools	1) Consumes more time and memory 2) Yet needs performance improvement techniques to implement. 3) Does not support applications that use reflections for privacy leakage 4) Java native interface libraries are not supported	Jinyung Kim
	DroidMOSS [29]	1) Effective detection of repackaged apps.	1) Assumes all the Google Play apps as legitimate apps. 2) Limited database to detect. 3) If original app is not present in database, it cannot detect repackaged apps.	J. Kim, Y. Yoon, and K. Yi
	SCanDroid [30]	1) It provides security at installation time.	1) This tool cannot be applied to packaged apps.	JesusFreke

Table 2 Malware Analysis through Dynamic Approach

Technique	Tools	Benefits	Limitations	Author
Profile Based Analysis	CrowDroid [32]	1) Analyze device profoundly.	1) Requires the installation of CrowDroid client application to perform detection. 2) Results incorrect if legitimate app invokes more system calls.	M. Egele, T. Scholte, E. Kirda, C. Kruegel
	AntiMalDroid [33]	1) Higher detection rate and can detect unknown malwares and their variants at runtime. 2) Dynamically extends malware database. 3) Better performance along with low cost	1) More time consumption.	Min Zhao, Fangbin Ge, Tao Zhang, Zhijian Yuan
Malicious Behavior	TaintDroid [20]	1) Resourceful tracking of susceptible information	1) It cannot track information that leaves the device and return in network reply.	E. William, G. Peter, C. Byunggon, C.

Analysis				Landon
Virtual Machine Introspection	DroidScope [23]	1) Helps in detecting privilege escalation attacks on the kernel.	1) Code coverage is very limited	Yan et al
Emulation Based	AASandbox [26]	2) Improve the efficiency of the antimalware programs for Android OS	2) Unable detect new malwares	Blaising et al

V. CONCLUSIONS

As we are aware that Android has become the most well-known smartphone Operating System, with the increasing popularity of Android devices and open source features the malware threat has also increased. Many Researchers have proposed various approaches to detect malware at two stages, first is before execution i.e. Static approach and second, is at the time of execution i.e. Dynamic approach. Both approaches have its own limitations and benefits. In this paper we have detailed the techniques and tools used for malware detection in Android. Our work will serve as a base to understand the taxonomy of malwares in Android. Our future work encompasses in developing an efficient tool to confront Android Malwares.

REFERENCES

[1] "Number of Android applications". AppBrain. February 9, 2017. Archived from the original on February 10, 2017. Retrieved March 12, 2017.

[2] Statt, Nick (May 18, 2016). "Android users have installed more than 65 billion apps from Google Play in the last year". The Verge. Vox Media. Retrieved March 12, 2017.

[3] Parvez Faruki, Ammar Bharmal, Vijay Laxmi, Vijay Ganmoor, Manoj Singh Gaur, Mauro Conti, Senior Member, IEEE, and Muttukrishnan Rajarajan,Android Security: A Survey of Issues, Malware Penetration, and Defenses- IEEE COMMUNICATION SURVEYS & TUTORIALS, VOL. 17, NO. 2, SECOND QUARTER 2015

[4] R. Raveendranath, V. Rajamani, A. J. Babu, and S. K. Datta, "Android malware attacks and countermeasures: Current and future directions," 2014 Int. Conf. Control. Instrumentation, Commun. Comput. Technol., pp. 137–143, 2014.

[5] "root exploits." [Online]. Available: http://www.selinuxproject.org/~jmorris/lss2011_slides/caseforse android.pdf. [Accessed: 15-Dec-2015].

[6] Y. Zhou and X. Jiang, "Dissecting Android Malware: Characterization and Evolution," 2012 IEEE Symp. Secur. Priv., no. 4, pp. 95–109, 2012.

[7] "Riskware | Internet Security Threats." [Online]. Available: http://usa.kaspersky.com/internet-security-center/threats/riskware#.Vm-5IUp97IU. [Accessed: 15-Dec-2015].

[8] Mila DallaPreda, Mihai Christodorescu, SomeshJha and SoumyaDebray, "A Semantics-Based Approach to Malware Detection", ACM Transactions on Programming Languages and Systems, Vol. 30, No. 5, Article 25, Pub. Date: August 2008.

[9] P. Faruki, V. Ganmoor, V. Laxmi, M. S. Gaur, and A. Bharmal, "AndroSimilar: Robust Statistical Feature Signature for Android Malware Detection," Proc. 6th Int. Conf. Secur. Inf. Networks, pp. 152–159, 2013.

[10] A.P. Felt, E. Chin, S. Hanna, D. Song, D. Wagner, Android permissions demystified, in: Proceedings of the 18th ACM Conference on Computer and Communications Security (CCS 2011), 2011, pp. 627–638

[11] A. P. Fuchs, A. Chaudhuri, and J. S. Foster, SCanDroid: Automated security certification of Android applications, Manuscript. [Online]. Available: http://www.cs.umd.edu/~avik/projects/scandroidascaa

[12] L. Lu, Z. Li, Z.Wu,W. Lee, and G. Jiang, "CHEX: Statically vetting Android apps for component hijacking vulnerabilities," in Proc. ACM Conf. Comput. Commun. Security, T. Yu, G. Danezis, and V. D. Gligor, Eds.,2012, pp. 229–240. [Online]. Available: http://dblp.uni-trier.de/db/conf/ccs/ccs2012.html#LuLWLJ12

[13] B. Sanz, I. Santos, C. Laorden, X. Ugarte-Pedrero, P. G. Bringas, and G. Álvarez, "PUMA: Permission usage to detect malware in android," Adv. Intell. Syst. Comput., vol. 189 AISC, pp. 289–298, 2013.

[14] J. Kim, Y. Yoon, and K. Yi, "S CAN D AL : Static Analyzer for Detecting Privacy Leaks in Android Applications."

[15] E. Chin, A. Felt, K. Greenwood, D. Wagner, Analyzing inter-application communication in Android, in: Proceedings of the Annual International Conference on Mobile Systems, Applications, and Services, 2011.

[16] "Dedexeruser‟s manual." [Online]. Available: http://dedexer.sourceforge.net/. [Accessed: 08-Nov-2015].

[17] A. Shabtai, U. Kanonov, Y. Elovici, C. Glezer, and Y. Weiss, "'Andromaly': A behavioral malware detection framework for android devices,"*J. Intell. Inf. Syst.*, vol. 38, no. 1, pp. 161–190, 2012. [Online]. Available:http://dblp.unitrier.de/db/journals/jiis/jiis38.html#ShabtaiKEGW12

[18] A. Reina, A. Fattori, and L. Cavallaro, "A system call-centric analysis and stimulation technique to automatically reconstruct Android malware behaviors," in *Proc. EUROSEC*, Prague, Czech Republic.

[19] D. Damopoulos, G. Kambourakis, and G. Portokalidis, "The best of both worlds: A framework for the synergistic operation of host and cloud anomaly-based IDS for smartphones," in *Proc. 7th EuroSec*, New York, NY, USA, 2014, pp. 6:1–6:6. [Online]. Available: http://doi.acm.org/10.1145/2592791.2592797

[20] E. William, G. Peter, C. Byunggon, and C. Landon, "TaintDroid: An information flow tracking system for realtime privacy monitoring on smartphones," in Proc. USENIX, 2011.

[21] I. Burguera, U. Zurutuza, and S. Nadjm-Tehrani, "Crowdroid: Behavior-based malware detection system forAndroid," in Proc.

 ISSN: 2321-8169
1111 – 1117

1st ACM Workshop Security Privacy Smartphones Mobile Devices, 2011, pp. 15–26. [111] K. O. Elish, D. (Daphne) Yao, and B. G. Ryder, "User-centric dependence analysis for identifying malicious mobile apps," in Proc. Workshop MoST, 2012.

[22] J. Huang, X. Zhang, L. Tan, P.Wang, and B. Liang, "AsDroid: Detecting stealthy behaviors in android applications by user interface and program behavior contradiction," in Proc. ICSE, 2014, pp. 1036–1046.

[23] L. K. Yan and H. Yin, "DroidScope: Seamlessly reconstructing the OS and Dalvik semantic views for dynamic Android malware analysis," in Proc. 21st USENIX Security Symp., 2012, p. 29.

[24] "State of mobile security," Lookout Mobile Security, Tech. rep., 2012.

[25] "Current world of mobile threats," Lookout Mobile Security, San Francisco, CA, USA, Tech. rep., 2013.

[26] C. Lever, M. Antonakakis, B. Reaves, P. Traynor, and W. Lee, "The core of the matter: Analyzing malicious traffic in cellular carriers," in Proc. NDSS, 2013, vol. 13, pp. 1–16.

[27] "Signature-Based Detection With YARA"

[28] M. Zheng, M. Sun, and J. C. S. Lui, "DroidAnalytics : A Signature Based Analytic System to Collect , Extract , Analyze and Associate Android Malware," 2013.

[29] J. Kim, Y. Yoon, and K. Yi, "S CAN D AL : Static Analyzer forDetecting Privacy Leaks in Android Applications."

[30] "[Utility][Tool][Windows] Baksmali / Smali Ma… | Android Development and Hacking." [Online]. Available: http://forum.xda-developers.com/showthread.php?t=2311766. [Accessed: 22-Dec-2015].

[31] Android Permissions Demystified." [Online]. Available: https://www.trustst c.org/pubs/848.html. [Accessed: 06-Nov-2015].

[32] M. Egele, T. Scholte, E. Kirda, and C. Kruegel, "A survey on automated dynamic malware-analysis techniques and tools," ACM Comput. Surv., vol. 44, no. 2, pp. 1–42, 2012.

[33] "strace download | SourceForge.net." [Online]. Available: http://sourceforge.net/projects/strace/. [Accessed: 22-Dec-2015].

[34] http://dlupdate.quickheal.com/documents/others/Quick_Heal_Annual_Threat_Report_2017.pdf

[35]]https://www.gdatasoftware.com/blog/2017/04/29666-malware-trends-2017