A Review Paper on Malware and Malware Prevention and Detection

[1]Nilesh Makwana, [2] Chandresh Parekh,
[1]Research Scholar, [2]Assistant Professor,
[1 & 2] Department of Information and Technology & Tele-Communication,
[1 & 2]Raksha Shakti University, Ahmadabad, India.

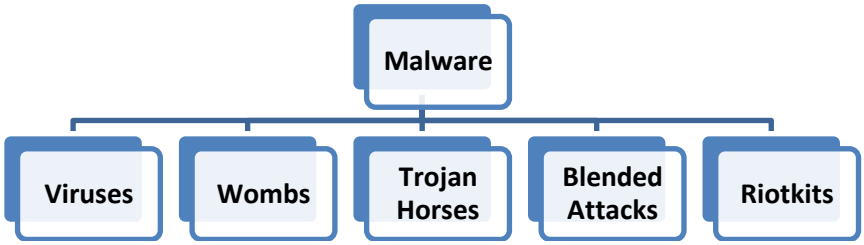
***Abstract*—In the course of the most recent decades, there were loads of concentrates made on malware and their countermeasures. The latest reports stress that the innovation of malevolent programming is quickly expanding. In addition, the serious utilization of systems and Internet expands the capacity of the spreading and the adequacy of this sort of programming. Then again, scientists and makers attempting awesome endeavors to create hostile to malware frameworks with powerful identification strategies for better insurance on PCs. Different applications contain bad conduct code; however those are not really malignant applications. The current framework classifications such applications as a malware applications. This issue will be overcome in proposed framework with the assistance of new element extraction calculation. In proposed framework chose android highlights will be separated for whole list of capabilities to recognize malware on four stages: bundle, client, application, and approval stage. The malware identification will be founded on behavioral and characterized by their hazard (High, Medium, and Low). This will be useful for the client to deal with the framework (Application) easily.**

***Keywords*—Malware, Malware Detection Systems, Antivirus, Malware Detection, Machine Learning, Pattern Recognition, Android.**

I. INTRODUCTION

Malware, otherwise called malignant code and noxious programming, alludes to a program that is embedded into a framework, normally clandestinely, with the plan of trading off the privacy, uprightness, or accessibility of the victim's information, applications, or working framework (OS) or of generally irritating or disturbing the casualty. Malware, for example, infections and worms is typically intended to play out these odious capacities such that clients are ignorant of them, at any rate at first. In the 1980s, malware was incidentally a disturbance or burden to people and associations; today, malware is the most outside danger to most frameworks, causing boundless harm and interruption and requiring broad recuperation endeavors inside generally associations. Spyware malware proposed to abuse a user's privacy has additionally turns into a noteworthy worry to associations. Despite the fact that security damaging malware has been being used for a long time, its utilization turned out to be considerably more across the board in 2013 and 2014, with spyware attacking numerous frameworks to screen individual exercises and direct money related misrepresentation.

As an establishment for later areas, this segment gives an outline of the different classes of malware, which incorporate infections, worms, Trojan steeds, and vindictive versatile code, and in addition mixes of these, known as mixed assaults. Malware likewise incorporates assailant apparatuses, for example, indirect accesses, rootkits, and keystroke lumberjacks, and following treats utilized as spyware. The dialog of every class clarifies how the malware regularly enters and taints frameworks and spreads; how it works (as a rule terms); what its targets are; and how it influences frameworks. The segment likewise gives a short talk of a couple of dangers that are not malware, but rather are frequently examined in conjunction with malware. The segment additionally shows a concise history of malware to demonstrate the relative significance of each malware class before and the present. The area finishes up by contrasting the classes of malware with recognize and contrasts. Malware can be clearly classified in following categories as given.



(Fig.1 Types of MALWARE)

1) Viruses:-

An infection is intended to self-replicate make duplicates of it and disperse the duplicates to different files, programs, or PCs. Every infection has a disease instrument; for instance, infections can embed themselves into have projects or information records (i.e., pernicious full scale code inside a word preparing file).The infection payload contains the code for the virus's objective, which can run from the generally kind (e.g., irritating individuals, expressing closely-held convictions) to the amazingly vindictive (e.g., sending individual data to others, wiping out frameworks). Numerous infections additionally have a trigger a condition that makes the payload be executed, which for the most part includes client association (e.g., opening a record, running a program, tapping on an email document connection). The two noteworthy sorts of infections are gathered infections, which are executed by a working framework (OS), and deciphered infections, which are executed by an application. This area talks about the two sorts of infections, and additionally the different confusion systems that infections use to stay away from location.

2) Worms:-

Worms are self-replicating programs that are completely self-contained, meaning that they do not require a host program to infect a victim. Worms also are self-propagating; unlike viruses, they can create fully functional copies and execute themselves without user intervention. This has made worms increasingly popular with attackers, because a worm has the potential to infect many more systems in a short period of time than a virus can. Worms take advantage of known vulnerabilities and configuration weaknesses, such as unsecured Windows shares. Although some worms are intended mainly to waste system and network resources, many worms' damage systems by installing backdoors perform distributed denial of service (DDoS) attacks against other hosts, or perform other malicious acts. The two primary categories of worms are network service worms and mass mailing worms. Network service worms spread by exploiting vulnerability in a network service associated with an OS or an application.

Once a worm infects a system, it typically uses that system to scan for other systems running the targeted service and then attempts to infect those systems as well. Because they act completely without human intervention, network service worms can typically propagate more quickly than other forms of malware. The rapid spread of worms and the intensive scanning they often perform to identify new targets often overwhelm networks and security systems (e.g., network intrusion detection sensors), as well as infected systems. Examples of network service worms are Sasser and Witty.

3) Trojan Horses:-

Named after the wooden stallion from Greek folklore, Trojan steeds are non-repeating programs that give off an impression of being kindhearted yet really have a concealed pernicious reason. Some Trojan stallions are planned to supplant existing documents, for example, framework and application executables, with malignant adaptations; others add another application to frameworks as opposed to overwriting existing records. Trojan stallions have a tendency to fit in with any of the accompanying three models:

1) Continuing to play out the capacity of the first program and furthermore performing partitioned, irrelevant vindictive movement (e.g., an amusement that likewise gathers application passwords)

2) Continuing to play out the capacity of the first program however altering the capacity to perform vindictive movement (e.g., a Trojan stallion rendition of a login program that gathers passwords) or to camouflage different malignant action (e.g., a Trojan steed variant of a procedure posting program that does not show different pernicious procedures)

3) Performing a noxious capacity that totally replaces the capacity of the first program (e.g., a record that cases to diversion however in reality just erases all framework documents when it is run).

Trojan steeds can be hard to distinguish. Since many are particularly intended to disguise their essence on frameworks and play out the first program's work legitimately, clients and framework heads may not see them. Numerous more up to date Trojan steeds additionally make utilization of a portion of similar muddling methods that infections use to maintain a strategic distance from discovery.

4) Blended Attacks:-

A mixed assault is an occasion of malware that uses numerous contamination or transmission techniques. The outstanding Nimda worm is really a case of a mixed assault. It utilized four dispersion techniques:

1) E-mail:- When a client on a powerless host opened a tainted email connection, Nimda misused defenselessness in the Web program used to show HTML-based email. Subsequent to contaminating the host, Nimda at that point searched for email addresses on the host and afterward sent duplicates of itself to those addresses.

2) Windows Shares: - Nimda examined has for unsecured Windows document shares; it at that point utilized NetBIOS as a vehicle component to contaminate records on that host. On the off chance that a client ran a tainted document, this would actuate Nimda on that host.

3) Web Servers: - Nimda examined Web servers, searching for known vulnerabilities in Microsoft Web Information Services (IIS). On the off chance that it found a defenseless server, it endeavored to exchange a duplicate of itself to the server and to taint the server and its documents.

4) Web Clients: - If a powerless Web customer went by a Web server that had been tainted by Nimda, the clients workstation would wind up plainly contaminated.

5) Rootkits:-

A rootkit is a gathering of documents that is introduced on a framework to adjust the standard usefulness of the framework in a vindictive and stealthy way. On some working frameworks, for example, adaptations of Unix and Linux, rootkits adjust or supplant handfuls or several records (counting framework pairs). On other working frameworks, for example, Windows, rootkits may adjust or supplant documents or may dwell in memory just and change the utilization of the OS's worked in framework calls. Many changes made by a rootkit conceal proof of the rootkits presence and the progressions it has made to the framework, making it extremely hard to verify that a rootkit is available on a framework and recognize what the rootkit changed. For instance, a rootkit may stifle index and process posting passages identified with its own documents. Rootkits are frequently used to introduce different sorts of aggressor devices, for example, indirect accesses and keystroke lumberjacks, on a framework.

II. LITRACURE REVIEW

a) **ZOZZLE: Fast and Precise In-Browser Java Script Malware Detection:-**

Over the most recent quite a while, we have seen mass-scale abuse of memory-based vulnerabilities move towards pile showering assaults. This is on account of more customary vulnerabilities, for example, stack-and pile based cradle invades, while still present, are currently frequently moderated by compiler methods, for example, Stack Guard [7] or working framework components, for example, NX/DEP and ALSR [12]. While a few load showering arrangements have been proposed [8, 9, 21], apparently, none are sufficiently lightweight to be incorporated into a business program. In any case, a program based location method is as yet appealing for a few reasons. Offline filtering is frequently utilized as a part of present day programs to check whether a specific site the client visits is generous and to caution the client generally. In any case, since it requires a long time to filter countless that are in the recognizable web, a few URLs will just be missed by the output. Offline checking is likewise not as compelling against transient malware that shows up and vanishes every now and again.

ZOZZLE is a generally static JavaScript malware locator that is sufficiently quick to be utilized as a part of a program. While its investigation is completely static, ZOZZLE has a runtime part: to address the issue of JavaScript confusion, ZOZZLE is coordinated with the program's JavaScript motor to gather and process JavaScript code that is made at runtime. Note that completely static investigation is difficult on the grounds that JavaScript code jumbling and runtime code age are so basic in both kind and malevolent code.

1) Challenges:-

Any specialized answer for the issue delineated above requires conquering the accompanying difficulties:

- **Execution: -** Discovery is frequently too ease back to be conveyed in a standard program.
- **Jumbled Malware: -** On the grounds that both considerate and vindictive JavaScript code is as often as possible muddled, absolutely static identification is by and large ineffectual.
- **Low false positive rates:** -It given the quantity of URLs on the web, while false positive rates of 5% are viewed as worthy for, say, static investigation devices, rates even 100 times bring down are not satisfactory for in-program location.
- **Malware short life: -** Transient malware bargains the viability of offline-just examining.

Since it works in a program, ZOZZLE utilizes the Java Script run time motor to open endeavors to cloud malware by means of employments of eval.document.write, and so forth by snaring the runtime and breaking down the JavaScript just before it is executed. We pass this unfurled JavaScript to a static classified that is prepared utilizing highlights of the JavaScript AST (theoretical language structure tree). We prepare the classified with a gathering of marked malware tests gathered with the NOZZLE dynamic stack showering locator [11].

Related work [4, 2] likewise classified JavaScript malware utilizing a blend of static and dynamic highlights, however depends on copying to the code and to watch dynamic highlights. Because we avoided mulation.our analysis is faster and, as we appear, frequently prevalent mistake.

2) Commitments: -

This paper makes these commitments:

- **Mostly static malware recognition: -** We propose ZOZZLE, an exceedingly exact, lightweight, for the most part static JavaScript malware indicator. ZOZZLE depends on broad experience breaking down a huge number of genuine malware locales found while performing dynamic creeping of a large number of URLs utilizing the NOZZLE runtime finder.
- **AST-based identification: -** We depict an AST-based procedure that includes the utilization of various leveled (setting touchy) highlights for distinguishing vindictive JavaScript code. This setting touchy approach gives expanded exactness in content based classification.
- **Fast classification: -** Since quick examining is vital to in-program selection, we display quick multi-highlight coordinating calculations that scale to hundreds or even a huge number of highlights.
- **Evaluation: -** We assess ZOZZLE as far as execution and malware location rates, both false positives and false negatives. ZOZZLE has a to a great degree low false positive rate of 0.0003%, which is short of what one of every a quarter million comparable to business hostile to infection items we tried against. To get these numbers, we tried ZOZZLE against a gathering of more than 1.2 million considerate JavaScript tests. In spite of this high accuracy the classified is quick, with a throughput at more than one megabyte of JavaScript code every second.

Classified-based devices are powerless to being evaded by an aggressor who knows the internal workings of the device and knows about the rundown of highlights being utilized, in any case, our preparatory involvement with ZOZZLE proposes that it is equipped for identifying a huge number of noxious destinations every day in nature. We think about the issue of avoidance.

3) Paper Organization: - Whatever remains of the paper is composed as takes after. Some foundation data on JavaScript misuses and their discovery and condenses our experience of performing offline checking with NOZZLE on a huge scale.

b) A Review Paper on Effective Behavioral Based Malware Detection and Prevention Techniques for Android Platform:-

Android is Linux based open sources operation framework created by Google. Presently a day it is extremely prominent because of its few highlights. As expanding in number of Android telephones there is concurrent increment in versatile malware applications that performs malignant exercises like abusing client's private data as sending messages. Malware is a kind of noxious programming which intrude on the distinctive operation on portable framework, crashes the vital data or private data of the versatile framework. As such malignant programming, malware alludes to programming programs intended to harm or do other undesirable activities on a portable framework. Besides, every one of these mischievous activities can be performed on Android gadgets without the client see (or when it is past the point of no return). It has been as of late announced that right around 60% of existing malware send stealthy premium rate SMS messages. The greater part of these practices is displayed by a classification of applications called Trojanized that can be found in online commercial centers not controlled by Google. Be that as it may, likewise Google Play, the official market for Android applications, has facilitated applications which have been observed to be noxious [1] [12].

Existing framework comprise of some constrained highlights of android application, malware discovery depends on behavioral base. The malware identification and counteractive action process is additionally static which make a few issues, for example, it increment false positive rate. Vindictive applications (nonexclusively called malware) constitute the primary vector for security assaults against cell phones. Masked as conventional and accommodating applications, they stow away slippery code that performs activities inside the foundation that undermines the client security, the gadget honesty, or possibly client's credit. Some basic cases of assaults performed by Android vindictive applications are taking contacts, login accreditations, instant messages, or noxiously subscribing the client to costly premium administrations. The present world is versatile (Smartphone) world. Because of low costs of Smartphone's is accessibility in 3G and 4G systems. Cell phones and tablets have turned out to be amazingly well known in the most recent years. Toward the finish of 2014, the quantity of dynamic cell phones worldwide was right around 7 billion, and in created countries the proportion between cell phones and individuals is evaluated as 120.8 % [16]. The majority of these many are average folks who don't realize what is the android structure and how android framework function. Because of well interface and transparency android is exceptionally famous step by step. There are significantly more application accessible which increment exhibitions of framework and in addition lessen time and cost. The client is uninformed of which application is great and which are destructive. Numerous applications are taking client individual information. More than 1 a large number of pernicious applications are as of now accessible on the planet [13]. Numerous applications looks like as would be expected and valuable applications, yet they stow away slippery code which performs activities out of sight that debilitates the client security, the gadget honesty, or even client's credit. Every single privately owned business and government associations moved their work to android application. The odds of data spillage and burglary of individual information is expanded. In existing framework concentrate on restricted highlights of android application to identify malware. In this way here framework is proposed to better malware identification in android Devices.

c) Android Malware Detection & Protection: A Survey:-

Since 2008, the rate of cell phone reception has expanded massively. Cell phones give diverse network choices, for example, Wi-Fi, GSM, GPS, CDMA and Bluetooth and so on which make them a universal gadget. Google says, 1.3 million Android gadgets are being enacted every day [1]. Android working framework abandoned its rivals far by catching over 78% of aggregate piece of the overall industry in 2013 [2]. Gartner report 2013 of cell phone deals demonstrates that there is 42.3% expansion in offers of cell phones in examination with 2012. As indicated by International information organization IDC, Android OS rules with 82.8% of aggregate pieces of the pie in 2Q 2015 [3]. Figure 1 demonstrates the pieces of the pie of Android working framework on yearly premise. It could be watched that Android has turned into the most broadly utilized working framework throughout the years. Android stage offers complex functionalities requiring little to no effort and has turned into the most famous working framework for handheld gadgets. Aside from the Android notoriety, it has turned into the principle focus for assailants and malware designers. The official Android advertises has a great many applications that are being downloaded by the clients in an extensive number regular [4].

The fame of Android working framework is expanding massively. The yearly records, displayed by IDC [3], demonstrate that Android OS pieces of the pie in second quarter (2Q) of 2015 are 82.8%, which is 2% diminish from the 2Q 2014. In the event that the esteem continues as before till the finish of year and continue diminishing each year with a similar rate then we can expect that in 2018, the Android pieces of the overall industry will drop to 76.8%. As per same record, the Android shares have expanded 5% out of 2014 from earlier year. In the event that it continue expanding with a similar rate and increments up to 89.8% till the finish of 2016 then we can state that the Android offers will grow up to 99.9% of every 2018. Moreover, it is anticipated that the pieces of the overall industry of the Android will be all things considered 88.4% of every 2018. The estimations and future forecasts of the Android advertise are registered and plotted in Figure 5. It ought to be noticed that with the expanded use of the Android based gadgets, the quantity of malwares assaulting Android is expanding at an exponential rate. In 2015, number of Android malwares spiked to 7.10 million. This figure is 2.84 million more than the earlier year [8] [9]

Rather than malwares, the antimalware have been planned and created in a wide range with a specific end goal to ensure the gadgets. It is derived that an antimalware utilizing static approach is less effective in distinguishing the noxious substance that are stacked progressively from remote servers. In spite of the fact that, the dynamic approach is effective as it continues checking the application and ready to identify the noxious substance at execution time. In any case, the parts of malignant code that are not executed stay undetected. It is trusted that any single security arrangement in Android can't give full insurance against the vulnerabilities and malwares. It is smarter to convey more than one arrangement at the same time for instance, a cross breed of two methodologies, i.e. static and dynamic. The half breed approach will first statically examine the application and will at that point perform dynamic investigation. This crossover arrangement might be a costly strategy to apply on account of the constrained accessible assets, for example, battery, memory and so on. Be that as it may, the restriction of this half and half arrangement can be tended to in twofold. Initially, the static investigation can be performed locally on the Android gadget; and subsequently, the dynamic examination could be performed in a dispersed manner by sending the vindictive action or occasion as a log record to a remote server.

d) A Survey on Malware and Malware Detection Systems:-

Host-based intrusion detection systems monitor dynamic behavior and state of specific computer system to see if there are any internal or external activities defraud the system policy. This kind of malware detection systems idiomatically named (“in-the-box”) because they reside in the same host that they are monitoring [13]. Network-based intrusion detection systems (IDS) are used to sniff all the packets on network nodes for analysis. In this type a single sniffer module placed in each network segment to monitor traffic in that segment. In contrast distributed network-based intrusion detection system has multiple modules placed in each node to monitor traffic in those nodes. Network-based malware detection systems idiomatically named (“out-of-the-box”) because they reside outside the host that they are monitoring [19]. There are hybrid intrusion detection systems; used with mixture of host-based and network-based capabilities. This type of IDS consists of multiple subsystems locating on separate nodes in the network for monitoring and gathering data from these nodes. The data collected by these subsystems is sent to the main system for analysis and classification [5]. Regarding effectiveness issue both host-based and network based detection systems have their drawbacks, while host based protects effectively internal system but it is susceptible to external attack, network-based can prevent external attack but it can’t protect inside host [4].

According to [19] a virtual machine (VM) is defined as an efficient isolated duplicate of real machine with characteristics of conformity with the original system, efficiency and full control of system resources. Virtual machine-based malware detection systems are constructed on the basis of the mentioned concept. There exist three classes of VM used by malware detection systems; Sandbox is the first one where computer resources have to be reached through specific API provided by the VM where system receives information of a suspicious executable program from a user, analyzes its behavior by performing it in a controlled environment (sandbox) and sends analysis reports back to the user who has issued the information.

Secondly emulation where simulating the entire computer system for running the guest operating system and the VMM provides an execution environment for programs that are identical to the original machine with exception of differences caused by the availability of system resources or by timing dependencies while efficiency is the core characteristic of emulators. An emulator is a piece of software that acts as a hardware (i.e. CPU emulator simulates CPU functionality using software). The emulator does not directly execute a code; instead instructions are intercepted by the emulator, translated into corresponding sequence of instructions compatible with the targeted platform. System emulators are hidden to detection code so it is regarded as a suitable environment for malware analysis. Thirdly, in native system virtual machines, a virtual machine monitor (VMM) is a smaller piece of privileged code that privileging VM on the host computer. This characteristic makes it native VM with good performance, but liable to errors and tamper resistance [14].

III. CONCLUSION

This paper surveys the current malware discovery and avoidance procedures. Here existing malware location and anticipation approaches were talked about and outlined pernicious programming are available in substantial numbers on android stage contrast with others the damage of malevolent conduct is expanding as of late which conveys more noteworthy difficulties to discovery and counteractive action of android malignant programming. Here framework is proposed with the assistance of new element extraction calculation for recognition of android malware applications and hazard investigation of it. Contingent on framework result client will start fitting activity. This framework will address android stage security issues will be valuable in not so distant future, it gives an up and coming near examination for a large portion of malware families and it condenses various malware recognition frameworks. In spite of the fact that the creating procedures of malware and their discovery frameworks are quickly developing, this examination can be considered as a key reference for the engineers in the field. It is discovered that there are diverse strategies like consent based procedure, signature based methods, and design based systems which are being used in this field. The paper endeavors to review and investigate every strategy and a short discourse about them is given. In future, new calculations, for example, machine learning systems, for example, grouping and forecast can be used in this field. Additionally a similar strategy can be connected in different fields, for example, android malware identification and so on.

REFERENCES

[1] NwokediIdika and Aditya P. Mathur "A Survey of Malware Detection Techniques" pp.1-47, February 2, 2007.

[2] Dennis Distler "Malware Anylasis an introduction" pp.1-64, December 14, 2007.

[3] Vinod.P, Laxmi.V.,Chauhan.G "MetamOrphicMalware Exploration Techniques Using MSA signatures"In: International Conference on Innovations in Information Technology (IIT),pp 232-237(2012).

[4] Kumar,N.D., Mishra.L., Charan.M.S., Kumar.B.D " The New Age Of Computer Virus and Their Detection"In: International Journal of Network Security & Its Applications (IJNSA), Vol.4, pp 79-96 ,(2012).

[5] Rafique,M.Z.,Chen,P.,Huygens,C., Joosen,W "Evolutionary Algorithms for Classification of Malware Families through Different Network Behaviors" Pp, 1-8, (2014).

[6] Imtithal A. Saeed,AliSelamatand Ali M. A. Abuagoub "A Survey on Malware and Malware Detection Systems" In:International Journal of Computer Applications,Vol.67,pp 25-31,(2013).

[7] Agrawal,H., Bahler,.L, Micallef,J., Snyder,S., Virodov,A.: Detection of Global, Metamorphic Malware Variants Using Control and Data Flow Analysis. Pp 1-6 ,IEEE(2013).

[8] Emad,S.A., Hashemi,.S. " A General Paradigm for Normalizing Metamorphic Malwares".In:10th International Conference on Frontiers of Information Technology. pp 349-353(2012).

[9] Cai,Y.L.,Ji,.D., Cai,.D.F. "A KNN Research Paper Classification Method Based on Shared Nearest Neighbo". Pp 336-340, (2010).

[10] Baoli,.L, Shiwen,.Y., Qin"An Improved k-Nearest Neighbor Algorithm for Text Categorization".In:20th International Conference on Computer Processing of Oriental Languages, pp 1-6 ,(2003).

[11] 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

[12] Wei Wang, Xing Wang, Dawei Feng, Jiqiang Liu, Zhen Han, and Xiangliang Zhang, Member, IEEE, "Exploring Permission-Induced Risk in Android Applications for Malicious application Detection" IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, VOL. 9, NO. 11, NOVEMBER 2014

[13] Yuan Zhang, Min Yang, Zhemin Yang, Guofei Gu, Peng Ning, and Binyu Zang, "Permission Use Analysis for Vetting Undesirable Behaviors in Android Apps" IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, VOL. 9, NO. 11, NOVEMBER 2014

[14] Silvio Cesare, Member, IEEE, Yang Xiang, Senior Member, IEEE , and Wanlei Zhou, Senior Member, IEEE, "Control Flow-Based Malware Variant Detection" IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, VOL. 11, NO. 4, JULY/AUGUST 2014

[15] Christopher S. Gates, Jing Chen, Ninghui Li, Senior Member, IEEE, and Robert W. Proctor, "Effective Risk Communication for Android Apps" IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, VOL. 11, NO. 3, MAYJUNE 2014

[16] Zhiyong Shan and Xin Wang "Growing Grapes in Your Computer to DefendAgainst Malware" IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, VOL. 9, NO. 2, FEBRUARY 2014

[17] Vaibhav Rastogi, Yan Chen, and Xuxian Jiang ,"Catch Me If You Can: Evaluating Android Anti-Malware Against Transformation Attacks" IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, VOL. 9, NO. 1, JANUARY 2014

[18] Wei Peng, Student Member, IEEE, Feng Li, Member, IEEE,Xukai Zou, Member, IEEE, and Jie Wu, Fellow, IEEE "Behavioral Malware Detection in Delay Tolerant Networks", IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 25, NO. 1, JANUARY 2014

[19] Junghwan Rhee, Member, IEEE, Ryan Riley, Member, IEEE, Zhiqiang Lin, Member, IEEE, Xuxian Jiang, and Dongyan Xu, Member, IEEE, "Data-Centric OS Kernel Malware Characterization" IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, VOL. 9, NO. 1, JANUARY 2014