International Journal of Engineering & Technology, 7 (3.34) (2018) 8-14

International Journal of Engineering & Technology

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

Research paper

A Survey on Malware Detection Techniques on Linux Powered Smart Phones using Machine Learning Approaches

[1]Mr.Rahul Y. Pawar, [2]Dr.C.Mahesh

[1] Computer Engineering Department Dr.D.Y.Patil Educational Complex, DYPCOE,Akurdi Pune-44.
[2] Department of Information Technology Veltech University Avadi, Chennai.
**Corresponding author E-mail: rahulpawar8087@gmail.com*

Abstract

Mobile Phone manufacturers are continuously working to take move on with rapid pace on their new models and to match with the need of customer, they need to customize their system. However the security scenarios of such practice are not that known, due to this various malware and viruses are increasing day by day and causing harm to the devices. Due to the substantial damage caused by malware in last few years certain significant efforts on developing detection and defense mechanism against malwares. For detecting such malicious applications and malwares a security system should be developed which will target such anomaly or outliers in system. In data mining anomaly detection system plays a major role by monitoring the behavior of an application and categorizing them in to normal and abnormal to detect malwares present in the system.

Keywords: *smart phones; System calls; targets; malwares; machine learning; ptrace.*

1. Introduction

The difficulties for cell phone security are winding up fundamentally the same as those that PCs experience and basic desktop security arrangements are frequently being cut back to cell phones. As a for example, dissected normal desktop security arrangements and assessed their materialness to cell phones. In any case, a portion of the desktop applications like antivirus can't be utilized on cell phones as they expend excessively CPU and memory and might bring about fast depleting of the power source. Since the reaction time of antivirus suppliers may shift between a few hours to a few days to recognize and distinguish the new malware and after that by creating new signature, refresh their customer's signature database; in such cases programmers have a slight edge of such opportunities.

2. Survey of Literature

'Droidtrace system: based on tracing process in dynamic analysis with capabiltiy of forward execution'
This research work explains the working of DroidTrace, a forward execution (analysis at runtime) capable malware detection tool.
The key points made in this research work are:
i. Most of the malware detectors for android uses the static methodology for analysing malwares. Static methodology needs the code to be compiled first. But, DroidTrace uses dynamic methodology which, basically, can analyse the applications even at runtime there are lots of static and dynamic loading tools available but they all have some kind of limitations.
ii. DroidTrace makes use of ptrace command to extract certain system calls of the applications for targeting the process (android application)
iii. The uniqueness in DroidTrace is that it considers physical modifications as a parameter too and hence is capable of automating all the user interactive feature of the application such as registrations, login details, button click and other various interactive events
iv. This research has carried out an analysis on 50,000 apps out of which 36,170 uses dynamic payloads (runtime changeable features) and has classified 294 malwares in 10 families
v. DroidTrace can also be used on smartphones which are in use so software analist can check hardware platforms without restricting to emulators very helpful in observing the behaviour of certain applications on specific hardware specifications
How they did it:

Stage 1: by Setting Functions as Targets

a. In This Stage Target is on dex file of the application and it does reverse engineering on the dalvik executable file. Disassembling lets us know how variables are used and how functions are being invoked. In this way DroidTrace identifies the functions which have runtime loading behavior. This is necessary to separate out the functions which might change their behaviour during runtime and hence they need to be handled with the help of forward execution technique

Step 2: Generates ACFG

b. After selecting & setting the target functions, it makes
a.Functions : Control Flow Graphs (CFG)
b.Apllications : Function Call Graph (FCG)

CFG + FCG = Application Control Flow Graph (ACFG)
ACFG is generated by DroidTrace, keeping in account and combining the information obtained from CFG and FCG.

c.DroidTrace keeps in mind the following:

a.Multiple application entry points: An android application may have multiple entry points such as the main activity, services, broadcast listeners etc and hence it takes into consideration all of it by taking help of the parser into the XML file namely Android Manifest, where all the important information is kept.

b. Components connections: UI and other features of the android applications which work on various IDs or need extra information from somewhere else are handled and are connected with their respective handlers in the ACFG. For example a button is connected and linked to it's respective onClick() function in the ACFG.

c. Conditional connections: It handles the if….else, switch statements by forming a new execution path for each condition and then running through each of them in the detection phase

Step 3: Forward Execution

a.This research eliminates use of monkey runner as it needs IDs of each components at compile time but components of UI and other elements are given IDs at runtime.

b.It introduces a new concept where the apk is rebuilt. It will add user event trigger codes in the modified apk and it will also add the code for conditional execution as discussed in the previous step.

Step 4: Malware Detection

a.This research uses one of the many techniques for analysing and monitor system calls, that is ptrace.

b.After using ptrace, it classifies the various system calls combining them with their proper description and useful titles.

This paper then illustrates the working of DroidTrace by providing the statistics of the analysis they did on 50,000 applications and at the end by providing a case study on 'CEPlugnew' and stating how approach followed by DroidTrace can be useful.

3. Securing the Mobile Applications: Threats and Defense Mechanisms

In the paper Authors concentrated on two noteworthy viewpoints, first why PDAs are defenseless against different security assaults, malignant conduct and dangers of malware. In next half it surveys the current malware anticipation and discovery Methods. The paper contains study of different recognition methods of malware location.
The paper summarizes the critical attacks caused by malware in linux powered smart phones, preventive approaches and detection mechanisms. In future there is scope to develop such detection systems targeting advance malwares.

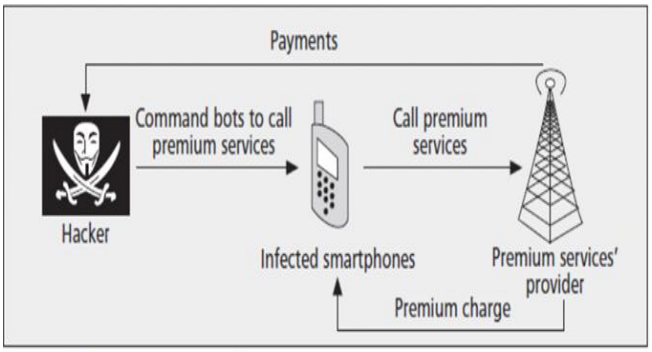


Figure 1. Diallerware attacks.

Table 1: Datasets and Methods Used

Attack	Description
Phishing	Clients 'certifications, for example, account points of interest and charge card numbers are gathered by methods for applications, messages, or SMS, which appear to be certified.
Spyware	Users'activities on the cell phones are being observed, which implies individual data is removed or deduced. Contrasted with reconnaissance assaults, spyware does not have particular focused on casualties
Surveillance attacks	A particular client is under observation by methods for his/her tainted cell phone, making utilization of the implicit sensors.
Diallerware attacks	Users'money is stolen utilizing the malware that makes shrouded calls to premium numbers or SMS administrations.
Financial malware attacks	Such assaults plan to take users'credentials from the cell phones or perform man-in-the-center assaults on money related applications.
Worm-based attacks	A worm is a malware that copies itself, normally proliferating starting with one gadget then onto the next, utilizing diverse means through a current system without users'intervention.
Botnets	A botnet is an arrangement of zombie gadgets that are tainted by malware with the goal that a programmer can remotely control them.

Defense methodology: A two stage mechanism is deployed of defense against malware. First to prevent the malware form getting in Smartphone and second to proactively detecting the existing malware and cleaning it.

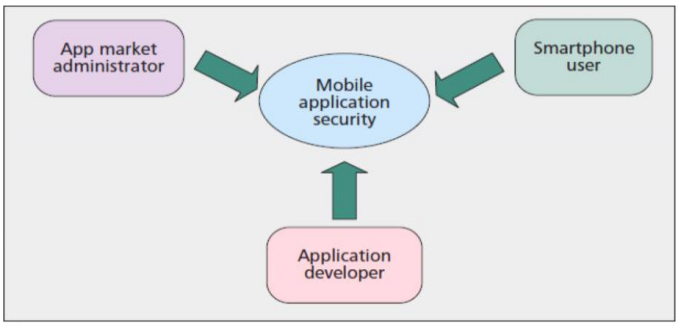


Figure 2. Co-operation among the stake holders.

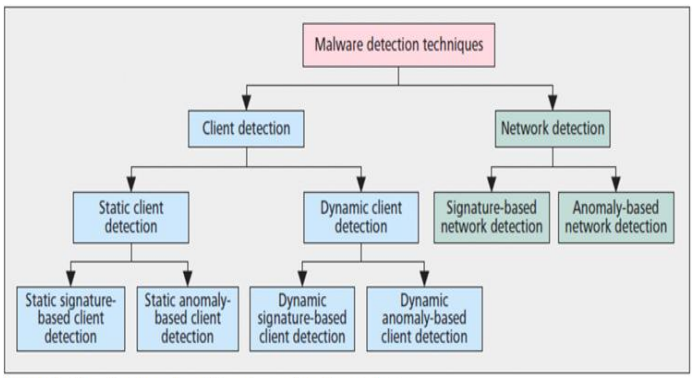


Figure 3. Classification of mobile malware detection techniques.

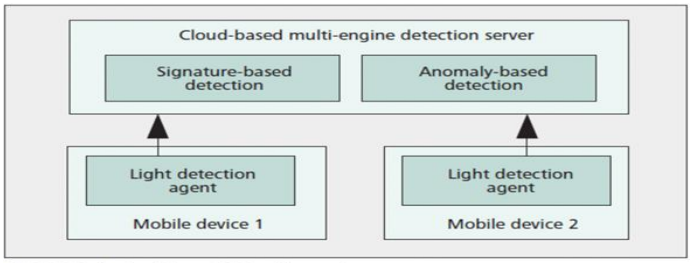


Figure 4. A cloud-based detection system.

In addition to these techniques user should take more precautions such as,

-Not clicking any random pop ups.
-Read terms and condition before signing up anywhere online,
-Use anti spyware scanner.etc.

3.1. Results:

Different significant assaults caused by malware are condensed and conceivable preventive methodologies and systems are proposed.

In future the malware expelling assignment will be shared amongst cloud and gadget so the computationally serious errands ought to be done in cloud which will make the framework more adaptable

4. Catch Me if you can: Evaluating Android Anti-Malware Against Transformation Attacks

The paper has summary and evaluation of various malware protection products, The Detection system identifies and concluded the protocol of professional mobile malware detection systems for Android Operating system and testing has been done such to analyze the resistivity verses various common methods which are less clear as far as understanding level.
The approach is to apply different changes to the android applications containing malware tests. Certain changes, for example, local code encryption are not done to totally robotize on the grounds that and that is already taken care. As future work, a more extensive assessment can be performed utilizing a considerably bigger number of malware tests and hostile to malware devices.
In this method the anti malware apps are evaluated to reduce the number of signatures generated that such systems use for malware Detection and to check the resistance of these apps against changes in malware binaries.

4.1. Data Sets and Method:

TABLE I: Anti-malware products evaluated.

Vendor	Product	Package name	Version	# downloads
AVG	Antivirus Free	com.antivirus	3.1	50M-100M
Symantec	Norton Mobile Security	com.symantec.mobilesecurity	3.3.0.892	5M-10M
Lookout	Lookout Mobile Security	com.lookout	8.7.1-EDC6DFS	10M-50M
ESET	ESET Mobile Security	com.eset.ems	1.1.995.1221	500K-1M
Dr. Web	Dr. Web anti-virus Light	com.drweb	7.00.3	10M-50M
Kaspersky	Kaspersky Mobile Security	com.kms	9.36.28	1M-5M
Trend micro	Mobile Security Personal Ed.	com.trendmicro.tmmspersonal	2.6.2	100K-500K
ESTSoft	ALYac Android	com.estsoft.alyac	1.3.5.2	5M-10M
Zoner	Zoner Antivirus Free	com.zoner.android.antivirus	1.7.2	1M-5M
Webroot	Webroot Security & Antivirus	com.webroot.security	3.1.0.4547	500K-1M

TABLE II: Malware samples used for testing anti-malware tools

Family	Package name	SHA-1 code	Date found	Remarks
DroidDream	com.droiddream. bowl-ingtime	72adcf43e5f945ca9f72064b81dc0062007f0fbf	03/2011	Root exploit
Geinimi	com.sgg.spp	1317d996682f4ae4cce60d90c43fe3e674f60c22	10/2011	Information exfiltration; bot-like capabilities
Fakeplayer	org.me.androidappli-cation1	1e993b0632d5bc6f0741 0ee31e41dd3164354d997	08/2010	SMS trojan
Bgserv	com.android.vending. sectool.v1	bc2dedad0507a916604f 86167a9fa3069339e2080	03/2011	Information exfiltration; bot-like capabilities; SMS trojan
BaseBridge	com.keji.unclear	508353d18cb9f5544b1e d1cf7ef8a0b6a5552414	05/2011	Root exploit; SMS trojan packed as payload
Plankton	com.crazyapps.angry. birds.rio.unlocker	bee2661a4e4b347b5cd2 a58f7c4b17bcc3efd550	06/2011	Dynamic code loading

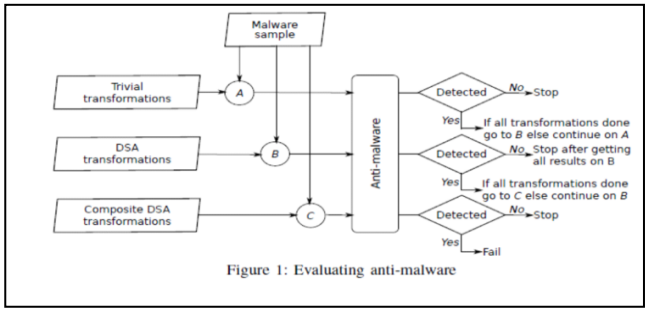


Figure 1: Evaluating anti-malware

4.2. Results:

These are the results of evaluation of 10 anti malware android apps. 6/10 apps are susceptible to common evasion techniques.

TABLE IV: DroidDream transformations and anti-malware failure. Please see Table III for key. 'x' indicates failure in detection.

	AVG	Symantec	Lookout	ESET	Dr. Web	Kaspersky	Trend M.	ESTSoft	Zoner	Webroot
P										
A									x	
RP	x							x	x	
EE			x						x	
RI		x							x	x
ED									x	
CR									x	
CI									x	
JN									x	
RI+EE		x	x	x					x	x
EE+ED			x			x			x	
EE+RF			x				x		x	
EE+CI			x		x				x	
RP+RI+EE+ED+RF+CI	x	x	x	x	x	x	x	x	x	x

TABLE V: Fakeplayer transformations and anti-malware failure. Please see Table III for key. 'x' indicates failure in detection. EE transformation does not apply for lack of native exploit or payload in Fakeplayer.

5. Machine Learning Classifiers Evaluation for Malware Detection in Smartphones

The Paper summarizes different Detection techniques based on classifiers and Proposed an effective solution for assessing malware location utilizing various machine learning algorithms on the basis of anomaly based approach. Four classes chose are essential data, association based data, content based data, and time based data.
This examination displayed an assessment utilizing various machine learning algorithms to differentiate adaptable threats appropriately by selecting fitting system highlights for review by the classifiers, and additionally to decide the perfect classifier in light of True Positive values. The Importance of this testing is latest information gathered by itself and. Also, the outcome and Performance of the machine learning classifiers is very convincing.

5.1. Method:

In the system architecture a Solution is proposed for detecting malwares using the anomaly-based approach as well as use of machine learning classifier is done. This Paper consists a solution of on finding malwares of android softwares with the help of machine learning and consistently. This paper uses KNN algorithm, Bayes Network, & Random forest classifiers to gain output .This paper uses publically available Datasets (MalGenome) and self collected datasets to find malwares.

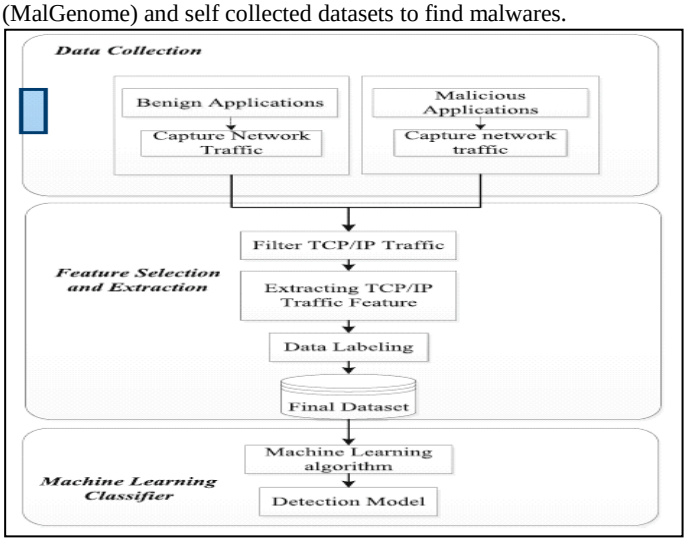


Figure 5:

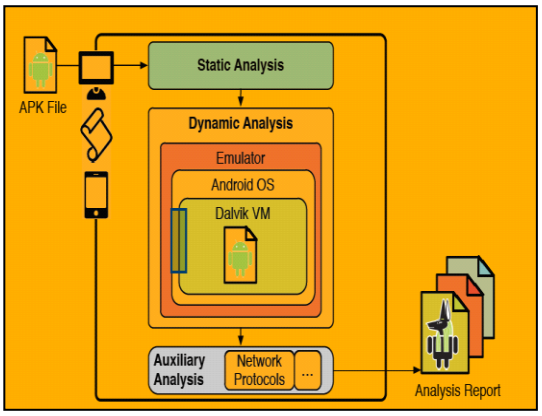
Result: There are three Stages of machine learning classification:
(1) Data Collection;
(2) Feature Extraction and feature selection;
(3) Classification using machine learning.
(4)When Genome Malware dataset is used with the random forest classifier, the experimental results shows detection rate with 99.99% accuracy.
(5) The detection rate with 84.57 % accuracy achieved when KNN Algorithm is used.

6. Andrubis: a Scenario on Current Android Malware Behaviors.

In this exploration paper Martina Lindorfer, Matthias Neugsch wandtner Proposed a completely computerized, freely accessible and thorough malware detection framework for Android applications. ANDRUBIS combines both static and dynamic approaches on framework and Dalvik Virtual Machine level. and in addition a few incitement methods to build code scope. 1,000,000 Android applications till date are tested on ADRUBIS framework. From last four years assessment of the dataset crossing tests, examination concluded about the diversity of malware dangers.
Dynamic code was used previously for identifying the abnormal behavior, is especially for various good ware, and, there information leakage of important information value while differentiating between normal and malicious apps. In future there is scope to explore the networking entities targeted by Android malware further with communication with windows operating systems.

6.1 Method:

This Paper consist of Analysis of android malware softwares with the help of ANDRUBIS which utilizes openly available Datasets Sand Droid.But this paper also traces malwares dynamically with constant track on activities performed by softwares at runtime and static analysis of the same malware software is carried out with the help of analysis of features extracted while installing.



The main Method observes all the system calls performed by the android device. In this paper different analysis techniques are used such as
-Static Analysis
-Dynamic Analysis
-Auxiliary Analysis

6.2. Result:

This system works on static and dynamic analysis which tends to gain maximum throughput with the help of Dalvik virtual machine and dataset available openly.

7. Analyzing Anti-Malware Framework over the Operating Systems of Smartphones.

The Research paper focuses on some malware detection frameworks for Smartphone operating systems and proposed a Secures Anti-Malware framework (SAM) for hardening of Smartphone operating systems to prevent malicious activities. The core idea of the framework resembles a smart city. The framework acts as the government of the city and treats the applications as citizens.
In the desktop computer world, malware are detected and removed by the anti-virus software. However, anti-virus software is mostly ineffective in mobile platforms due to their different security model. Also, they are useless when their database is out of date. Goal can be achieved to make the framework as general as possible so that the framework can be deployed in any operating system.

7.1. Method:

This paper has proposed a new framework SAM (Secure Anti Malware) which not only monitor activities but also blocks the malicious activities of the application. For carrying this activity we it also uses some LAWS and POLICIES. It has following features.
1. Zones for specified access only.
2. The framework dynamically switches to different security levels on the basis of applications.
3. Monitoring device activities
4. The framework uses API's for providing limited and needful resources only to the software's Methods used are Constantly focusing on actions took by device and it makes a list of trusted devices which makes it simple to manage and monitor other devices.I th uses its sensors to detect physical entities such as Home Office etc.All the tasks n SAM are executed with the help of an Manager.

7.2. Result:

SAM provides a framework which completely analyses the software and segregates its functionalities and resources needs and communication channels so as to attain less threats and hacking on OS.

8. Malware Detection on the Basis of String for Andriod Applications

The paper survey introduced string based malware detection which primarily targets to improve malware detection using only the permissions which are given to android applications that are widely used by the anti-viruses which are static engines and binary information, for providing a scalable and effective solution based on machine learning. The conclusion of this paper is that machine Learning is the best and effective technique amongst all types of detection frameworks. The later stage will be concentrating on improvement in including different new features such as applications which are third party application(not from play store)and their API calls and other data.

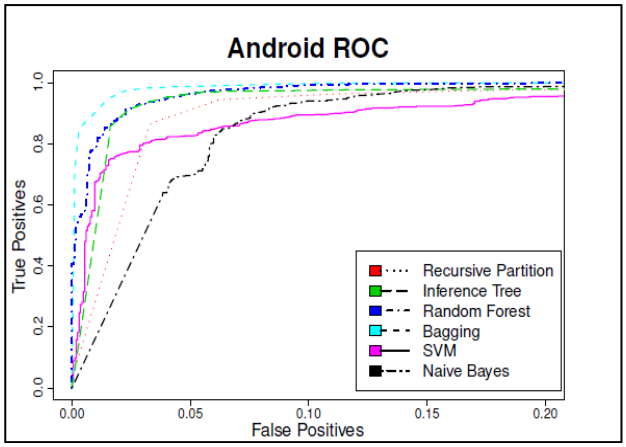
8.1. Method:

Extraordinary compared to other identification philosophies are those in light of machine realizing when they are normally supporting static and dynamic both methodologies. These techniques associated successfully in different and heterogeneous zones, for instance, relational associations, self-decision systems, PC recreations and improvement .this are more adaptable and

genuinely supportive. In machine getting the hang of bunching is a standout amongst the most critical systems for grouping of information.

8.2. Results:

This graph shows the complete results for the all machine learning classification algorithms. The results show that random forest algorithm is having best results with average accuracy 0.936 whereas Naive Bayes classifier is having average accuracy of 0.83 which is worst result amongst all classifiers.



9. Android Security: a Survey of Issues, Malware Penetration and Defences

The paper focuses on overview on different malware recognition strategies. The discoveries are Android malware dangers are determined because of expansive number of gadgets as yet running on more established and helpless OS renditions Static and Dynamic investigation produce application movement reports to help malware examiners to settle on obscure suspicious specimen. It can be closed as half breed discovery approach(s) are picking up unmistakable quality for investigating malware. So robotized, half and half approach for Android malware examination is suggested.

9.1. Method:

To achieve malware detection there are two methods used such as static and dynamic
1) Static: This approach actually stores the signatures of android malwares and uses them for detection without execution of application.
2) Dynamic: This approach is used to monitor their activity and behavior of application.

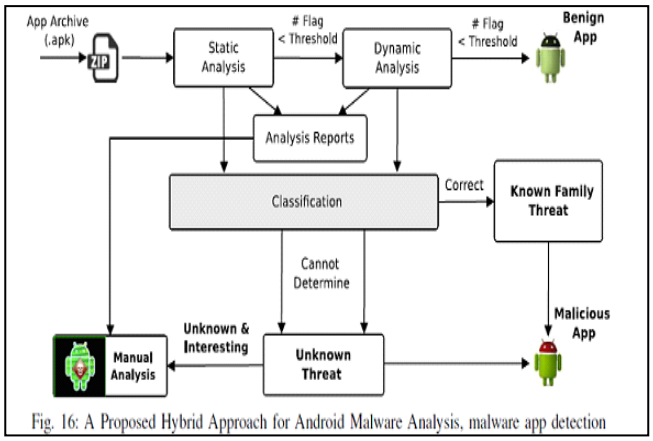


Fig. 16: A Proposed Hybrid Approach for Android Malware Analysis, malware app detection

9.2. Results:

With static analysis Hybrid approach is dissected for Android malware analysis of APK file. If in case of its static analysis fails against byte code which is encrypted, dynamic analysis performs behavioral detection. In results it is observed that hybrid detection approach is improving performance over detection of malware.

10. A Survey on Android Malware Mitigation Techniques and Behaviours.

The paper focused on review of famous security postured by Android malware. Primary attention is on the feature generally found in corrupted android applications and follows the root level features which are actually kernel level and plays major role for authorize area frameworks.

10.1. Method:

The destination number and message body is accepted by sendTextMessage() method and it provides couple of the contentions as fixed admires setContentView() technique call to show a new tab for this operation Uri.parse() strategy acknowledges a top notch telephone number for communicating something specific he purpose is propelled utilizing startActivity() strategy call.

10.2. Result:

As a Result no notification or information sent for various activities like incoming call or text messages to the target device. The permission required is only edit phone settings permission and to be saved accordingly. This technique is bit complex and results are stored in non relational kind of databases.

11. Trigger Scope: Detection of Logic Bombs in Linux based Applications

According to the authors t android is most well known stage in now days. The working arrangement of versatile is focused by malware. Malevolent application rationale is difficult to distinguish without a determination of the "ordinary," expected usefulness of the application. Noxious application reason that is executed, or initiated, quite recently under certain (every now and again tight) conditions as a rationale bomb they make an underlying move towards recognizing rationale bombs. In Detailed view, trigger investigation is proposed which is another static examination system that looks to consequently distinguish triggers in Android applications examination joins delegate execution, way predicate the amusement and data reduction, and covers the procedural control exam to enable the authentic ID and depiction of triggers.

11.1. Methods:

Mainly there are three methods involved in trigger scope

1. Super control-flow graph (sCFG) for control dependency analysis and inter-procedural CFG and the intra-procedural CFGs are layered on each other.
2. Date.getYear() method is used for accessing the Year over a given time object.
3. Date.getMinutes() method is used for accessing the Minutes Block Predicate Extraction.

11.2. Result:

The system recorded 35 applications as malicious applications from the various applications which are available on the Play Store. In the Following Table summarizes the results obtained after analyzing each and every step. These analysis techniques are capable of reducing the number of recorded applications.

Domain	# Apps	# Apps With Checks	# Apps With Suspicious Checks	# Apps With Suspicious Triggered Behavior	# Apps After Post-Filter Steps
Time	4,950	1,026	302	30	10
Location	3,430	137	71	23	8
SMS	1,138	223	89	64	17

Table I: This table summarizes how the different steps of our analysis are able to drastically reduce the number of false positives when detecting triggered malware in a large set of benign applications obtained from the Google Play store.

C. Trigger Analysis Results

12. Analysis of Permission Use for Evaluating Malicious Behaviors in Android Apps.

In the paper author presented a examine the android app permissions .Protect the sensitive information from untrusted apps . User can give the permission to app. Early syscall-based analysis technique not suitable for android app. The Vet-Droid is a Dynamic analysis of a android system. Vet-Droid is examining the behavior of the system by user perspective. Vet-droid provide a systematically permissions, i.e. how application are used the permissions for accessing system resource vet-droid can observe it.
Additionally security examination is perform. Vet-droid use to security and noxious conduct detection. VetDroid can distinguish unpretentious vulnerabilities in a few applications generally difficult to recognize.

12.1. Method:

The above examination demonstrates that a general technique to dissect touchy practices of Android applications is exceedingly wanted. programmed strategy to sift through most APIs that certainly can't enroll callbacks, and physically check few unused APIs and Finally, the separating technique finds into PScout's interface rundown to select such interfaces with a contention sort which is available in subclass list or the interface list programmed separating technique predominantly chooses interfaces that makes list of all remaining intents . The determination technique is to pinpoint all possible interfaces whose contentions may contain a Pending Listener.

Table VI:

	Android 2.3	Android 4.0	Android
Total APIs	10906	15013	13009
Selected callback API	286	319	338
New callback API	-	65(33)	42(20)

A pursuit on PScout's Application package interface list, 58 interfaces found with this technique which contains a Pending Listener separating strategy significantly lessens the manual endeavors and requires just little endeavors to keep pace with the Android variant.

12.2. Result:

Vet Droid reconstructed the use of permission behaviors for detecting malwares are can eventually ease the analysis of malwares.

13. Future Scope

As future work, some dynamic detection techniques can be extended to automatically self detection applications when they are getting updated. Three different areas are to be targeted attributes to be selected, Detection method and machine learning technique respectively. More stress can be given on system calls selection as feature extraction for system level work.

14. Conclusion

This survey paper summarizes the various malware detection techniques, categorizes various malware detection algorithms and compares the static and dynamic way of malware detection. It also focuses on performance measurement and accuracy from all above mentioned published technical papers and review articles in techniques of malware detection. Generally the detection performance and accuracy of dynamic detection techniques are more superior to various malware detection models. The final conclusion is that behavior based or dynamic detection system is better than static detection or signature based techniques. The malware detection systems should be equipped with new algorithms by considering various system level features.

References

[1] He, Daojing, Sammy Chan, and Mohsen Guizani. "Mobile application security: malware threats and defenses." IEEE Wireless Communications 22.1 (2015): 138-144.

[2] Rasthofer, Siegfried, et al. "Droidforce: Enforcing complex, data-centric, system-wide policies in android." Availability, Reliability and Security (ARES), 2014 Ninth International Conference on. IEEE, 2014.

[3] Rastogi, Vaibhav, Yan Chen, and Xuxian Jiang. "Catch me if you can: Evaluating android anti-malware against transformation attacks." IEEE Transactions on Information Forensics and Security 9.1 (2014): 99-108.

[4] Bartel, Alexandre, et al. "Static analysis for extracting permission checks of a large scale framework: The challenges and solutions for analyzing android." IEEE Transactions on Software Engineering 40.6 (2014): 617-632.

[5] Narayanan, Annamalai, et al. "Adaptive and scalable Android malware detection through online learning." Neural Networks (IJCNN), 2016 International Joint Conference on. IEEE, 2016.

[6] Alzaylaee, Mohammed K., Suleiman Y. Yerima, and SakirSezer. "DynaLog: An automated dynamic analysis framework for characterizing Android applications." Cyber Security And Protection Of Digital Services (Cyber Security), 2016 International Conference On. IEEE, 2016.

[7] Saracino, Andrea, et al. "Madam: Effective and efficient behavior-based android malware detection and prevention." IEEE Transactions on Dependable and Secure Computing (2016).

[8] Medvet, Eric, and Francesco Mercaldo. "Exploring the Usage of Topic Modeling for Android Malware Static Analysis." Availability, Reliability and Security (ARES), 2016 11th International Conference on. IEEE, 2016.

[9] Narudin, Fairuz Amalina, et al. "Evaluation of machine learning classifiers for mobile malware detection." Soft Computing 20.1 (2016): 343-357.

[10] Zhou, Xiaoyong, et al. "The peril of fragmentation: Security hazards in android device driver customizations." Security and Privacy (SP), 2014 IEEE Symposium on. IEEE, 2014.

[11] Allix, Kevin, et al. "A Forensic Analysis of Android Malware--How is Malware Written and How it Could Be Detected?." Computer Software and Applications Conference (COMPSAC), 2014 IEEE 38th Annual. IEEE, 2014.

[12] Zhauniarovich, Yury, and Olga Gadyatskaya. "Small changes, big changes: an updated view on the Android permission system." International Symposium on Research in Attacks, Intrusions, and Defenses. Springer International Publishing, 2016.

[13] Lindorfer, Martina, et al. "Andrubis--1,000,000 apps later: A view on current Android malware behaviors." Building Analysis Datasets

and Gathering Experience Returns for Security (BADGERS), 2014 Third International Workshop on. IEEE, 2014.

[14] Iqbal, MdShahrear, and Mohammad Zulkernine. "SAM: A secure anti-malware framework for the smartphone operating systems." Wireless Communications and Networking Conference (WCNC), 2016 IEEE. IEEE, 2016.

[15] Batten, Lynn M., VeelashaMoonsamy, and MoutazAlazab. "Smartphone applications, malware and data theft." Computational Intelligence, Cyber Security and Computational Models. Springer Singapore, 2016. 15-24.

[16] Martín, Alejandro, Héctor D. Menéndez, and David Camacho. "String-based malware detection for android environments." International Symposium on Intelligent and Distributed Computing. Springer International Publishing, 2016.

[17] Martín, Alejandro, Héctor D. Menéndez, and David Camacho. "String-based malware detection for android environments." International Symposium on Intelligent and Distributed Computing. Springer International Publishing, 2016.

[18] Faruki, Parvez, et al. "Android security: a survey of issues, malware penetration, and defenses." IEEE communications surveys & tutorials 17.2 (2015): 998-1022.

[19] Allix, K., Bissyandé, T. F., Jérome, Q., Klein, J., & Le Traon, Y. (2016). Empirical assessment of machine learning-based malware detectors for Android. Empirical Software Engineering, 21(1), 183-211.

[20] Liang, Shuang, and Xiaojiang Du. "Permission-combination-based scheme for android mobile malware detection." Communications (ICC), 2014 IEEE International Conference on. IEEE, 2014.

[21] Cooper, Vanessa N., Hossain Shahriar, and Hisham M. Haddad. "A survey of Android malware characterisitics and mitigation techniques." Information Technology: New Generations (ITNG), 2014 11th International Conference on. IEEE, 2014.

[22] Zheng, Min, Mingshen Sun, and John CS Lui. "DroidTrace: a ptrace based Android dynamic analysis system with forward execution capability." Wireless Communications and Mobile Computing Conference (IWCMC), 2014 International. IEEE, 2014.

[23] Li, Li, Alexandre Bartel, Tegawend´e F. Bissyand´ e, Jacques Klein, Yves Le Traon, Steven Arzt, Siegfried Rasthofer, Eric Bodden, Damien Octeau, Patrick McDaniel "Iccta: Detecting inter-component privacy leaks in android apps." Proceedings of the 37th International Conference on Software Engineering-Volume 1. IEEE Press, 2015.

[24] Dash, Santanu Kumar, et al. "Droidscribe: Classifying android malware based on runtime behavior." Security and Privacy Workshops (SPW), 2016 IEEE. IEEE, 2016.

[25] Fratantonio, Yanick, et al. "Triggerscope: Towards detecting logic bombs in android applications." Security and Privacy (SP), 2016 IEEE Symposium on. IEEE, 2016.

[26] Zhang, Yuan, et al. "Permission use analysis for vetting undesirable behaviors in android apps." IEEE transactions on information forensics and security 9.11 (2014): 1828-1842.

[27] Jang, Jae-wook, et al. "Detecting and classifying method based on similarity matching of Android malware behavior with profile." SpringerPlus 5.1 (2016): 273.