International Journal of Advance Research in Science and Engineering
Vol. No.5, Special Issue No. (02), March 2016
www.ijarse.com
IJARSE
ISSN 2319 - 8354

ANDROID ANTI-MALWARE AGAINST TRANSFORMATION ATTACKS

Ajinath N. Pawar[1], Rupali A.Holkar[2], Poonam S. Ahire[3], Saiprasad K. Malekar[4], Prof. Ashwini S. Bhamare[5]

[1,2,3,4,5]*Dept. of Computer Engineering, BVCOE & RI, Anjaneri, Trimbakeshwar, Nashik (India)*

ABSTRACT

Now a day's Android is the most popular and useful operating system for mobile. Attack of malware threats have recently becoming a real problem in android devices. In this paper, we have stated a simple and high efficient technique for detecting malware or viruses inside the Android apps on Google Play store which required to be installed by various users in their smart phones. In addition, a majority of them can be finding by calculating risk score of particular app with less effort. If the applications are having some malicious intention; it might be possible that most of these applications come from an unknown developer and so there is higher possibility of them being malicious. To overcome these problems, we are developed a system in which we first find out the risk score of particular app using Naive Bayes and classify that app into normal and malicious, if app is normal you can download that app which by which your android device remains safe and if app is malicious then our android Anti-malware system suggest you similar kind of apps with low risk score. We have applied our Anti-malware detection system over the exiting apps given on a play store; it provides appropriate risk score of every app.

Keywords- Risk-Score, Malware, Mobile, Android, Anti-Malware, Security, Mobile Apps, Polymorphism, Transformation.

I. INTRODUCTION

Android devices such as smart phones, tablets and palmtop computers are incrementing day by day. Unfortunately, this popularity attracts malware attackers too. Recently mobile malware has been already become a serious problem because of craze of various apps. There are many apps available on Google playstore, Malware attackers can attack your android devices via these apps. As we know that Android is one of the most popular platforms, with increase in android devices malwares also has been increases constantly. With the rise of malware attacks, the platform has seen an evolution of anti-malware tools, with a range of free and paid service that are now available on the official Android mobile app market called as Google Play Store. In this paper we aim to evaluate the risk score of searched app by using signature and script record of that associated app and categorize that app into malicious and normal by using Naïve Bayes algorithm which is applied on Training set where as training set consist of sets, malware and words. Additionally we have used the term 'transformation' for reference of various polymorphic or metamorphic changes. Our domain study is different from that we exclusively focus on android devices like smart phones, tablets that require various ways for anti-

malware design. Malware attacks on android devices have presently increased in the extent of their evolution but the capabilities of existing anti-malware tools are difficult to understand. So we provide Anti-malware detection system which is very easy to understand and provide appropriate and efficient result.

We regularly and systematically evaluate anti-malware product for android devices regarding its resistance against various transformation techniques in already known malware space. So we developed Droid Chameleon that has regular and systematic framework with various transformation techniques. We have been implemented a prototype of Droid Chameleon and use it to evaluate ten popular anti-malware products for Android then Our analysis shows that all of them are vulnerable to common evasion techniques. The signature studied do not requires static analysis of byte code. We has been studied the evolution of anti-malware tools over a period of two years. Our basic analysis shows that some anti-malware tools try to strengthen their signature with a trend towards content-based signature while previously they escaped by certain transformations not involving code-level changes. The improved signatures still show to be vulnerable. Based on our evaluation results, here explored possible ways to improve current anti-malware solutions. To be precise highlighted out that android eases developing modern detection techniques because much code is high-level byte codes rather than native and primary codes. Lastly, certain platform support can be enlisted to cope with advanced transformations.

II. LITERATURE SURVEY

Ajinath N. Pawar, Rupali A. Holkar, Saiprasad K. Malekar, Poonam S. Ahire A. Y. Bhamare[18] Android Anti-Malware Against Transformation Attacks it provide requirement design analysis and of algorithms and all related object oriented models with respect to proposed system described in this study. In Jul. 2012, "ADAM: An automatic and extended platform is to stress test Android anti-virus systems" was developed by M. Zheng, P. Lee, and J. Lui [2]. It was an automated and extended system that evaluated the effectiveness of anti-virus using various malware samples for Android platform. It automatically changes an Android malware samples into different variants through various repackaging and difficult techniques, while preserving the original malicious behavior. ADAM can automatically change an original malware sample to different variants via repackaging and difficult techniques in order to evaluate the robustness of different anti-virus systems against malware mutation [2]. ADAM is designed by connecting different building blocks. These blocks are tested using different anti-viruses against malware samples

Advantages -It can be used for study of very large-scale malware samples and changes is done manually so there is no need to apply manual modification of malwares. This results less overhead on codes.

ADAM is not always capable to avoid an anti-malware tool. It implements only some of changes, such as renaming methods, introducing junk methods. It cannot be said that ADAM will always provide the better detection mechanisms and this is main limitation of this system.

"A taxonomy of obfuscating[3] transformations" declared by C. Collberg, C. Thomborson and D. Low, Dept. Comp. Sci., Auckland University, Auckland, New Zealand, Tech. Rep. 148, 1997. It have been the focus of much research due to their relevance. This helps to preserve privacy policies between sender and receiver. In this technique executer does the actual execution.

Advantages-Obfuscations can be easily used for tracing software pirates.

Limitations- The obfuscated software remains secret until the powerful deobfuscator to be built. Therefore, there must be short time period between the releases of obfuscated software versions.

"Semantics-preserving Malware Detection" technique was invented by M. Christodorescu[4], S. Jha, and C. Kruegel, in 2007, it proposed malware detector that is used to find out the malicious behavior of a program. Most of times hackers use obfuscation to change the malwares. In these, detectors use pattern-matching technique to search the obfuscations made by hackers. The advantage of the system here is totally syntax based technique. So, it is easy to be understood by detectors and it has relatively low run time overhead. In these system limitation is mandatory to save the patterns of malicious instructions into templates which need to use of large databases.

In 2009, "Effective and efficient malware detection at the end host," was developed by C. Kolbitsch[5], P. Comparetti, C. Kruegel, E. Kirda, X. Zhou, and X. Wang, in Proc. 18th Conf. USENIX Security Symp. It proposed a novel malware detection approach that is both effective and efficient and thus, can be used to replace old anti-virus tool at the end host. This technique analyzes a malware to build a model that has behavioral characteristics. Such models describe the information flows between the system calls essential to the malware's mission, and therefore, which cannot be easily evaded by simple obfuscations or polymorphic techniques. Then extract the program slices responsible for such information flows. For detection, execute these slices to match with these models against the runtime behavior of an unknown program.

Advantages- It can effectively detect running malicious code on an end user‟s host with a small overhead. It generate effective tool that capture detailed information about the behavior of a malwares variation. Scanner that can efficiently match the activity of an unknown program against this system.

Disadvantages-It cannot generate system call signatures or find a starting point for the slicing process. The new algorithms should be implemented for above limitation.

In 2012, "RiskRanker: Scalable and accurate zero-day android malware detection[6] ," was proposed by M. Grace, Y. Zhou, Q. Zhang, S. Zou, and X. Jiang, in Proc.10th Int. Conf. Mobile Syst., Appl., Services, , pp. 281–294. It proposed proactive scheme to spot zero-day Android malware, It does not depend on malware samples and their signatures. It is an automated system called RiskRanker to scalable analyze whether a particular app exhibits dangerous behavior (e.g. launching a root exploit or sending background SMS messages). It checks and translates potential security risks into corresponding detection modules of two orders of complexity. The first-order modules handle non-obfuscated apps by evaluating the risks in a straightforward manner; the second-order modules capture certain behaviors (e.g., encryption and dynamic code loading) to detect malware.

In 2011, "Automated Remote Repair for Malware" was proposed by Y. Nadji, J. Giffin, and P. Traynor, in this the malicious network traffic increases because of intruders. The problem can be solved by using Airmid, which is an automated system for remote remediation of mobile malware. After the detection of malicious traffic, the cellular network interacts with the source device to identify its originality of that traffic [7].

Disadvantages-It does not tie to device and its security. It is not able to characterize the traffic of large amount of malwares.

"Apps Playground: Automated Security Analysis of Smartphone Applications" was developed in Feb. 2013, by V. Rastogi, Y. Chen and W. Enck for doing the automation of security analysis the tool Apps Playground is used. It integrates multiple components which comprises of different detection and automatic exploration techniques for this purpose [8].The system can be evaluated using multiple large and small scale experiments involving real benign and malicious application. The main advantage of this is it gives effective analysis even with large number of applications, with limitation of less effective at automatically detecting privacy leaks and malicious functionality in application.

In 2012 "Hey, you, get off of my market: "Detecting malicious apps in official and alternative Android markets" was developed by Y. Zhou, Z. Wang, W. Zhou and X. Jiang.

To find out malicious applications related to android permission based behavioral foot printing scheme is used. It is used for known malwares. For that a heuristics based filtering scheme is applied to unknown malwares. This total system with known and unknown malicious families is called "Droid Ranger" [9].

III. EXISTING SYSTEM

In existing malware detection system like anti-virus we first download any apps then anti-virus system scan that Apps and display the result of detected viruses or malwares under that particular app. To evaluate existing anti malware system have develop number of systematic framework such as Droid Chameleon[1] with different transformation techniques that are used in the system which can be change the Android application automatically. Some of these changes are highly specific for the Android platform. Based on the framework, which pass known malware samples through these changes have generated new variants of malware which verify to possess the' original malicious functionality. here used that variants to evaluate the effective and popular anti-malware tool. Droid-Dream [12] and BaseBridge [13] are malware with root exploit packed into benign applications. DroidDream tries to get out root privileges by using two vatious root exploits the rage against the cage and exploid. BaseBridge includes only one exploit, rage against the cage. If these exploits are successful then both DroidDream and BaseBridge install payload applications. Geinimi [14] is a trojan packed into benign applications. It communicates with remote C&C servers and exfiltrates user information. Fakeplayer [15], the first known malware on Android, sends SMS messages to premium numbers, thus costing money to the user. Bgserv [16] is a malware injected into Google's security tool to clean out DroidDream and distributed in third party application markets. It opens a backdoor on the device and exfiltrates user information. Plankton [17] is a malware family that loads classes from additional downloaded dex files to extend its capabilities dynamically.

International Journal of Advance Research in Science and Engineering
Vol. No.5, Special Issue No. (02), March 2016
www.ijarse.com
IJARSE
ISSN 2319 - 8354

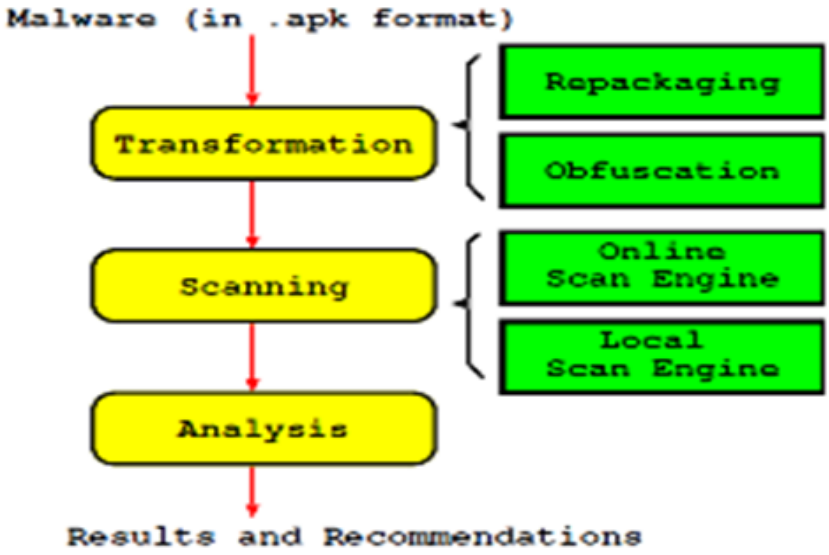

Figure1 : ADAM

IV. PROPOSED SYSTEM

Our approach is to provide anti-malware detection system which aims to detect viruses or malwares before download any app and shows the risk score of that app. If risk score of particular app is high then system searches for similar app with low risk score, Otherwise in case of low risk score user can download that app. Therefore user's android devices remain safe from malware attacks.

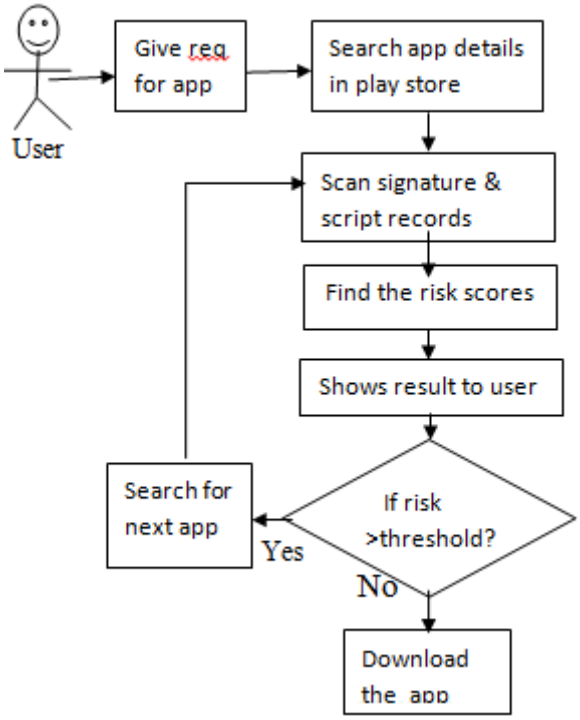

Figure2. Proposed System

Algorithm for Android anti-malware against transformation attack is given below:

Step 1: Start

Step 2: User give request for app.

Step 3: System search app details on Play store.

Step 4: System scans the signature and script record for app which is requested by user.

Step 5: Finding the risk scores of app requested by user.

Step 6: if risk score is higher than threshold then Search for next app.

Go to step 4

Step 7: Else result is low risk score then download the app directly.

Step 8: Stop

Algorithm: Naïve bayes classifier:

Input: D set of tuples with n number of attributes as Z = (z1, z2, z3,….zn), 'm'

Classes: (c1,c2,c3,…..cm)

Processing:

1. Start
2. Calculate probability of P of X with each Class c_i
3. Check whether $P(c_i|Z) > P(Z|c_i)$
4. Calculate $P(c_i|Z) = P(Z|c_i) P(c_i) P(Z)$
5. Maximize $P(c_i|Z) = P(Z|c_i) P(c_i)$ as P(Z) is constant.
6. Probablity can be calculated as:

   $P(Z\backslash C_i) = P(z1\backslash c_i)* P(z2\backslash c_i)*\ldots*P(z_n\backslash c_i)$
7. Stop.

Where,

P(c\Z) = posterior probability of class(target) given predictor attribute.

P(Z) = prior probability of predictor.

P(c) = prior probability of class.

P(Z\c) = likehood which is the probability of predictor given class.

V. SYSTEM OVERVIEW

There are various apps available on playstore. These apps can be malicious or it can be normal. So to categorize these apps into malicious or normal we are using naive bayes algorithm**.**

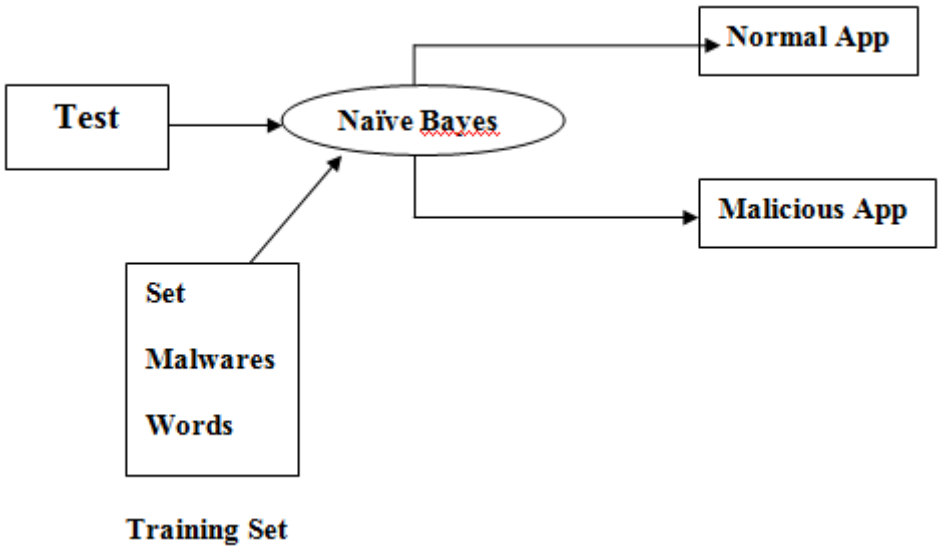

Figure 3: Working of Anti-Malware System

Training Set: Training set consists of set, malwares or words.

If we give input word as "Trojan" then system can't understand the meaning of Trojan, So that we are applying training set as a input to the system. Naïve bayes algorithm compares App signatures which are stored in test and training set, according to result of that comparison naïve bayes classifies app into malicious and normal.

The output of Naïve bayes is value i.e. risk score between 0.1 to 0.9, if value is between 0.1 to 0.5 then app is consider as normal app and user can download that app. If the output value is between 0.5 to 0.9 then app is consider as malicious then system check for similar app with low risk score.

VI. EXPERIMENTAL RESULT

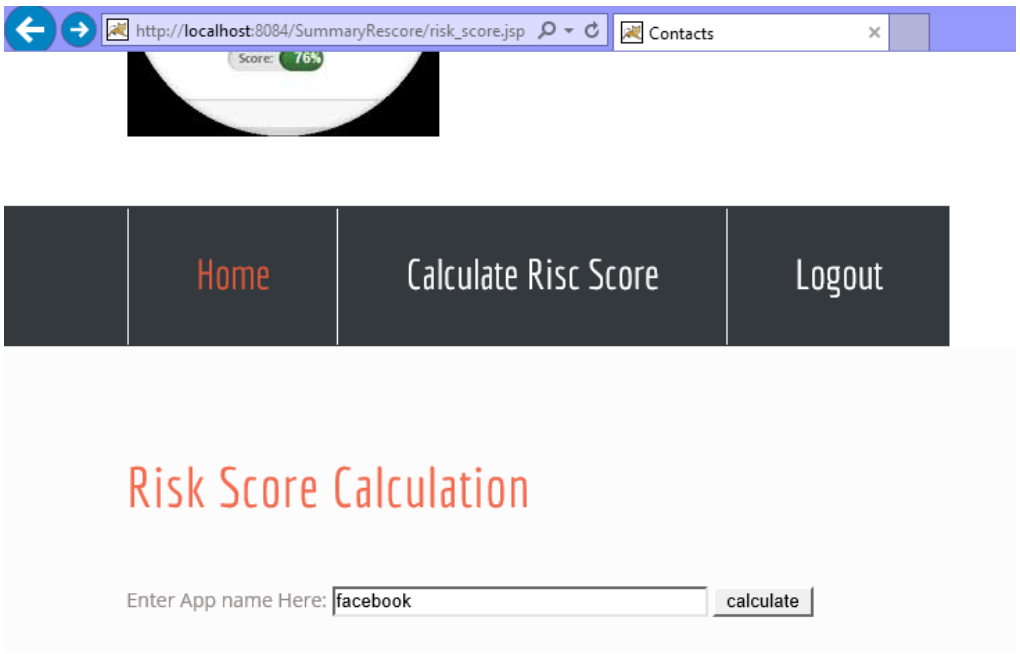

Figure2: calculate risk score

International Journal of Advance Research in Science and Engineering
Vol. No.5, Special Issue No. (02), March 2016
www.ijarse.com
IJARSE
ISSN 2319 - 8354

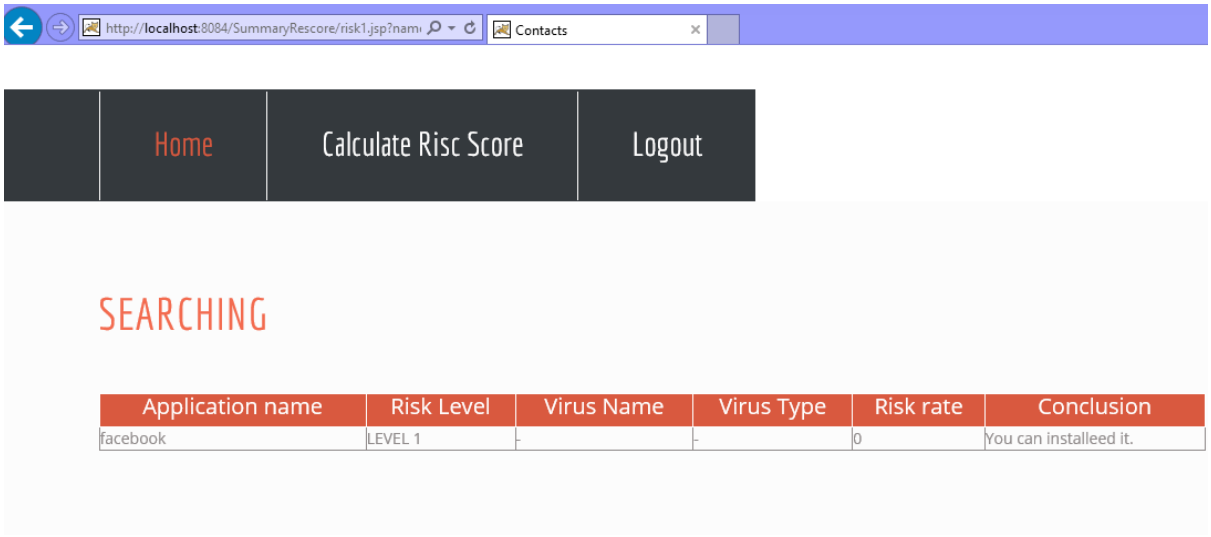

Figure 3: shows the final risk score of apps.

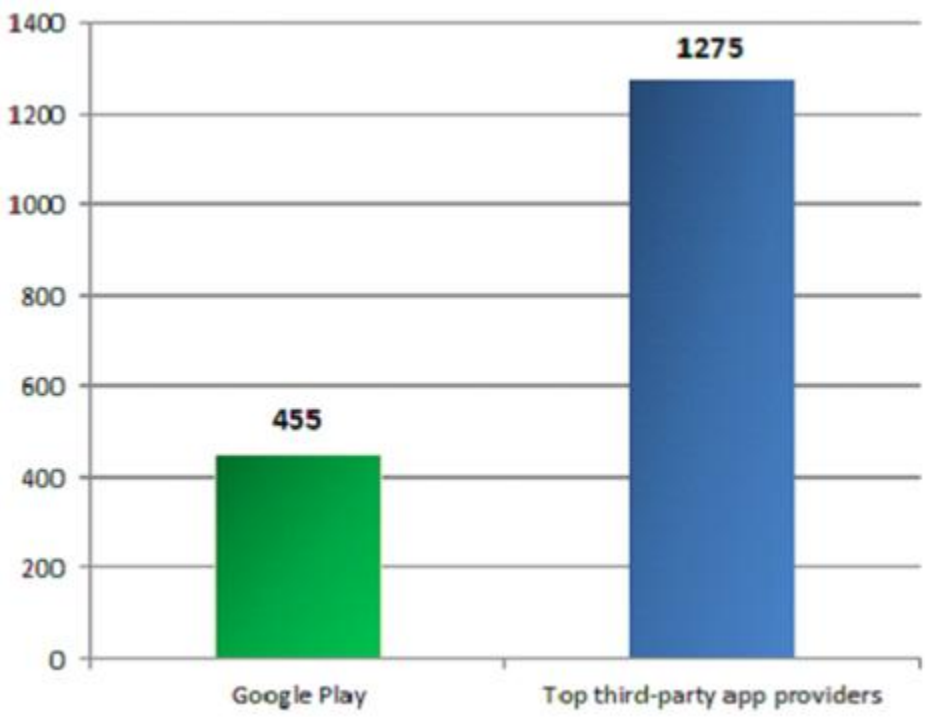

Figure 4. Experimental Result

Figure 3, shows the statistics of number of malicious applications in Google Play and Top third-party app providers. A malware can get itself into the system by different means like copying of files from external sources or devices onto the system and downloaded files from the internet, it checks the vulnerabilities of the system and infects the system at the point of its vulnerabilities. Smartphone malwares are capable of doing many things, such as: stealing and transffering the contact list and other data, locked the device completely, giving remote access to hackers, sending the SMS or MMS messages without user notice or permission etc[2].

Mobile malwares is of great public concern as the population of smartphones is much larger than the population of PCs.

VII. CONCLUSION

We has been taken analysis of different anti-malwares tools which can be used for avoidance of different malware attacks. ADAM tools complex mechanism are used for privacy preserving but with fewer transformations malware detectors that use complex techniques requires pattern matching techniques which was very complicated. A framework based on DroidChameleon[1] have been used more changes which are more accurate and efficient with anti-malware tools that can found. It is very necessary to protect the android device from malware attacks. We have provided a simple and high efficient technique for protecting the android devices from malware attacks and calculating the risk scores before downloading the apps from google play store. This anti-malware system is important for not only measuring the risk scores of mobile malware threats but also propose effective, next generation solutions. We exercise DroidChameleon[1], a systematic framework with various transformation techniques. We have developed this application because in our research it is found that the existing anti malware products have been fails to provide protection to common malware transformation techniques. Our results on various popular merchantile anti-malware applications for android are unreassuring none of these tools is tolerant against common malware transformation techniques. Additionally majority of this can be trivially discomfited by applying slight transformation over known malware with little effort for malware authors. Finally our results have proposed possible remedies for improving the current state of malware detection on all android devices and providing security to your android devices during downloading any app from Google play store.

VIII. ACKNOWLEDGMENT

We wish to express our sincere gratitude to Prof. C. K. Patil, Principal and Prof. Hemant D. Sonawane, Head of Computer Engineering Department, Bramha Valley College of Engg for his indispensable support, suggestions. I would like to take this opportunity to thank my internal guide Prof. Ashwini Y. Bhamare for giving me all the help and guidance I needed. I am really grateful to them for their kind support. Their valuable suggestions were very helpful. In the end our special thanks to Sai Infocorp Solution for providing various resources such as laboratory with all needed software platforms, continuous Internet connection, for Our Project. We also wish to express our gratitude to the official and other staff member who rendered their help during the period of making our review paper. Last but not least we wish to avail our self of this opportunity, express a sense of gratitude and love to our friends and our parents for their manual support, strength,help and for everything.

REFERENCES

[1] V. Rastogi, Y. Chen, and X. Jiang, "DroidChameleon: Evaluating Android anti-malware against transformation attacks," in Proc. ACMASIACCS, May 2013, pp. 329–334.

[2] M. Zheng, P. Lee, and J. Lui, "ADAM: An automatic and extensible platform to stress test Android anti-virus systems," in Proc. DIMVA, Jul. 2012, pp. 1–20.

[3]. C. Collberg, C. Thomborson, and D. Low, "A taxonomy of obfuscating transformations," Dept. Comput. Sci., Univ. Auckland, Auckland, New Zealand, Tech. Rep. 148, 1997.

[4]. M. Christodorescu, S. Jha, S. Seshia, D. Song, and R. Bryant, "Semantics-aware malware detection," in Proc. IEEE Symp. Security Privacy, May 2005, pp. 32-46.

[5] C. Kolbitsch, P. Comparetti, C. Kruegel, E. Kirda, X. Zhou, and X. Wang, "Effective and efficient malware detection at the end host," in Proc. 18th Conf. USENIX Security Symp., 2009,pp. 351–366..

[6] M. Grace, Y. Zhou, Q. Zhang, S. Zou, and X. Jiang, "RiskRanker: Scalable and accurate zero-day android malware detection," in Proc.10th Int. Conf. Mobile Syst., Appl., Services, 2012, pp. 281–294.

[7] Y. Nadji, J. Giffin, and P. Traynor, "Automated remote repair for mobile malware," in Proc. 27th Annu. Comput. Security Appl. Conf., 2011, pp. 413-422.

[8] V.Rastogi, Y.Chen, and W.Enck ,"AppsPlayground: Automatic security analysis of smartphone applications," in Proc. ACM CODASPY, Feb. 2013, pp. 209–220.

[9] Y. Zhou, Z. Wang, W. Zhou, and X. Jiang, "Hey, you, get off of my market: Detecting malicious apps in official and alternative Android markets," in Proc. 19th Netw. Distrib. Syst. Security Symp., 2012, pp. 1–13.

[10] (2013, Dec. 3).ProGuard [Online].

[11] R. Komondoor and S. Horwitz, "Semantics-preserving procedure extraction," in Proc. 27th ACM SIGPLAN-SIGACT Symp. POPL, 2000, pp. 155–169.

[12] Lookout, San Francisco, CA, USA. (2011). Update: Security Alert: DroidDream Malware Found in Official Android Market[Online].

[13] (2013, Dec. 3). Android.Basebridge—Symantec [Online].

[14] (2013, Dec. 3). Android.Geinimi—Symantec [Online].

[15] (2013, Dec.3). AndroidOS.FakePlayer—Symantec [Online].

[16] (2013, Dec. 3). Android.Bgserv—Symantec [Online].

[17] (2013, Dec. 3). Plankton [Online]. [18] Naïve Bayes Classification.

[18] Ajinath N. Pawar, Rupali A. Holkar, Saiprasad K. Malekar, Poonam S. Ahire, prof. A. Y. Bhamare "Android Anti-malware Against Transformation Attacks" in IEEE sep 2015, p-ISSN: 2395-0072