Source Code Analysis for Software Vulnerabilities in Android based Mobile Devices

R.Dhaya, PhD
Department of CSE,
Velammal Engineering College,
Chennai,

M.Poongodi
Department of CSE,
Velammal Engineering College,
Chennai,

ABSTRACT
Smartphone users are growing very fast in recent years, along with this mobile threats also increasing side by side. A mobile malware is a malicious code that aims to harm the devices .Malwares can cause system failure, decreasing battery charges, steals the information and corrupts data and go up the maintenance cost. So mobile phone security is vital one. Downloading mobile apps from the third party play store is risky one because a malware programmer inserts malevolent code into this. Users use these applications in their mobile phones and the malicious code misuse it without their knowledge. So many techniques are used to detect malwares. This paper uses a source/static code analysis to find the vulnerabilities in the applications and it also uses N-gram analysis to detect unknown malware characteristics.

Keywords - Malware, Android, Static Analysis, N-Gram, SVM, Vulnerability, CVSS

1. INTRODUCTION
Mobile devices such as Smartphone, tablets and Personal Digital Assistant are well liked gadget in recent years. Once a mobile device grows simultaneously their functional and technical complexity also increases. Latest Smartphone proffers a lot of services and applications than services gave by the computers. Most of the data transmission like Net banking, online shopping etc. will be done through mobile phones i.e. smart phones. The result of the mobile phone usage is to raise the criminals to get the benefit of these activities for unlawful gains. So data should be secured from any kind of electronic attacks. A mobile malware like virus, worms, Trojan etc is a malicious software program that aims to damage the mobile devices such as tablets, smart phones, Personal Digital Assistants, etc. Malwares broadcast via Short Messaging Service (SMS), Multimedia Messaging Service (MMS), and Bluetooth etc. Mobile viruses can spread from PC networks into mobile networks or vice versa. Because of this mobile security is vital one for the above. The major part of the mobile companies is using the Android Platform. Android platform is becoming very popular today and it is getting more vulnerable because it is an open source and easy to develop the applications freely. A Malware developer takes this one as an advantage to write malware programs. Because of these malwares, Android based Smart phones are easily attacked and it performs malicious activities such as theft the sensitive data, drain the battery without user's knowledge. Different security countermeasures are being developed and applied to Smartphone to mitigate the security threats. This paper illustrates N-gram and static analysis based malware prevention techniques for Android based mobile phones. This method is used to categorize the malware/benign mobile applications.

Many antivirus companies uses signature based detection algorithm but is not able to identify new viruses [1]. So we designed a software program that uses machine learning based algorithm (N-gram) to detect the given mobile applications is having malicious characteristics or not. For this we divided our dataset into two types.1.Training set – was used by N-gram to create a classifier to classify in the past unknown features of source code as malware or benign. 2. Test set- is an element of dataset that does not have any instances of previous one which are trained by N-gram analysis. Test set is used to check the performance and accuracy of the algorithm over unnoticed instances.

The rest of the paper is sorted as follows section 2 talks about an overview of the proposed system and Section 3 describes the implementation of proposed system. Section 4describes the analysis part of the proposed system and section 5 contains the conclusion of the proposed system.

2. SYSTEM OVERVIEW
2.1 Existing System
Most of the commercial antivirus companies used a signature based detection methods to identify malwares. Signature is the binary of pattern of the machine code of a particular virus [2]. It may be strings; binaries. It checks the content of the file dictionary of malware signatures [3]. This method lists the following disadvantages,

- It needs the huge database to store the malware signatures.
- It fails to identify an unknown malwares because a new malware may not contain a known signature of malwares [4].

2.2 Objective in Proposed System
This paper describes the static or source code analysis of an android application package files. Static approach assists to recognize the vulnerabilities such as SQL injection, Data Manipulation Language (DML), Password, Cookie poisoning, etc and join together with the software and evaluate the complexity necessitated in it using Common Vulnerability Scoring System. Because of this we are finding bugs in the mobile applications, before delivering it to the customer and detecting new vulnerabilities also. Previous system supports signatures (character strings, binary) to detect malwares. Our system considers signatures are Source code for this we are applying N-gram concept and these signatures are used to find the new security vulnerabilities in the applications.

Main Concepts are used in proposed system. They are,

- Source code analysis or Static analysis

It is used to pre checks the code to find bug in the application before it becomes to an issue [13].

- N-gram analysis

It is a one type of probabilistic model; it uses the concept of Markov chain model [10]. N-gram is a contiguous sequence of n times from a given sequence of input data and it is used to predict the next item [14].

Advantages of proposed system

- Quick and proactive
- Cheap in economical wise, Flexible and easily automated

2.3 Proposed System Architecture Diagram

This paper mainly focuses on identification of malwares in mobile phone application using search based malware detection algorithm namely, N-gram analysis. If malware is present in any mobile application, it performs malicious activities like sending SMS to address book, etc. Static code analysis is used to reverse the code without executing the android application file (apk) and convert it into the source code of the file and extracting the features of the application and use the machine learning tool to create the learning model database which is based on malware and benign applications to classify the malwares. We need to check whether the given test application is having malware or not. For this, compare the N-gram signatures of the test application and signature which is already stored in the files. The result will be the malicious or benign code of the given test application.

The proposed system explains the steps in the following block diagram shown in figure 1

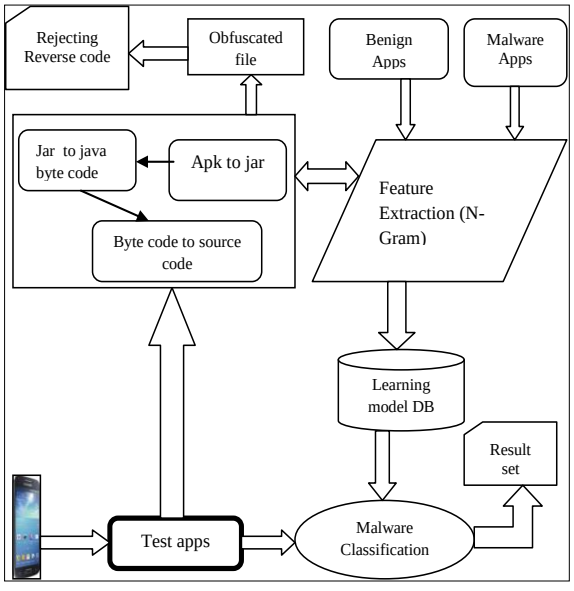

Fig.1 Architecture Diagram

3. IMPLEMENTATION

The proposed system contains four modules. They are followed by,

3.1. Reverse Engineering Automation

Reverse engineering is a method which is used to analyze an existing code in order to inspect the vulnerabilities or malicious characteristics in the software [5]. This method has the capability to create source code from an executable files. Many tools are used to perform the reverse engineering and static analysis methods in android mobile applications to find the malwares. Android application package files (.apk) are not allowed to execute in Java Virtual Machine (JVM), So Dalvik Virtual Machine (DVM) executes the applications in dalvik executable format (.dex). Classes.dex is a main component in android applications which is not able to view.Dex2jar is a tool which is used to convert dex files into .class file format. To convert the apk file with jar file using dex2jar tool [6] these file are converted into byte codes. It will be decompiled by JD-GUI. Java Decompiler-Graphical User Interface (JD-GUI) is a tool [8] to get the source code of the application as a java code. Apktool is used to decompile [7] and recompile the apk files. It contains manifest.xml file which includes the needed permissions of an android applications. Run a batch file to get the source code. Batch file is a set of commands which is used to perform a repeated task. Apk2java.bat filename.apk is used [9] to perform the reverse engineering process in the command prompt and it shows in figure 2.After executing the command the source code is created in the src folder of a particular mobile applications. It is shown in figure 3.

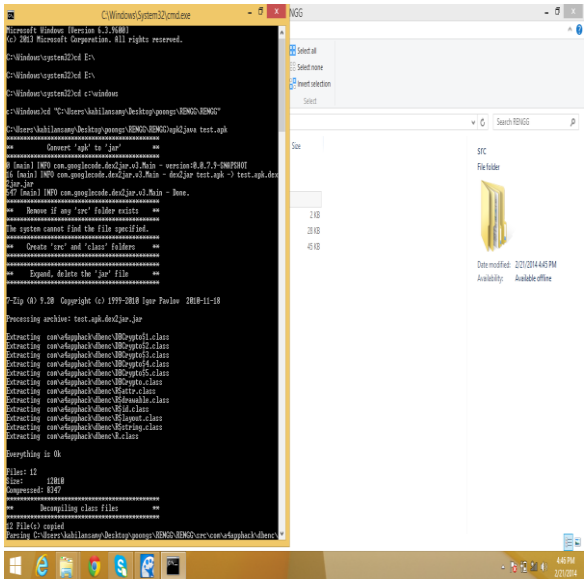

Fig. 2 apk2source Conversion

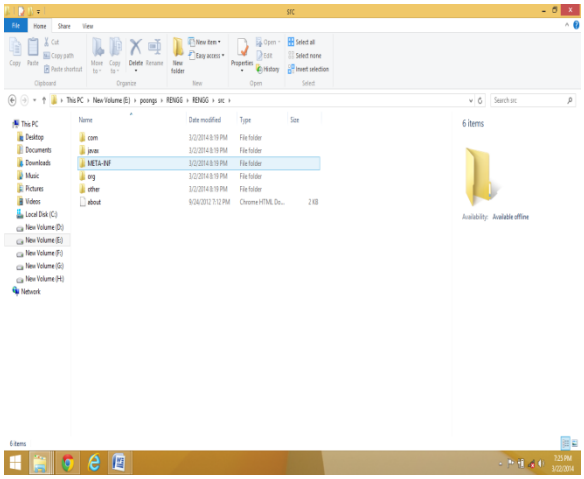

Fig 3. Checking Source Creation

3.2 Benign/Malware Feature Extraction from Public Resources

Training set was created based on Benign/Malware application's features like API call, Call flow, Device memory are extracted from android packages and Open Web Application Security Project (OWASP), torrent. The training set contains the vulnerabilities like SQL injection, Password, DML, Print stack trace ,parameter tampering, get attribute, Cookie poisoning, Cross Site Scripting(XSS) ,URL?,DML and so on and its feature vectors. It is gathered from the public resources. After extracting the source code, N-Gram concept is applied in the extracted features of the source code Signatures are text or strings instead of that source code is considered as N-Grams signatures. The following table1 lists the example of vulnerabilities and extracted features in android.

Table 1. Extracted Features

Vulnerabilities	Feature Vectors
Connection_String_Injection	CxList con = All.FindByName ("*get Connection"); CxList inputs = Find_Interactive_Inputs(); CxList sanitize = Find_General_Sanitize() + Find_Integers(); Result= con.Influenced ByAndNotSanitized (inputs, sanitize);
SQL_Injection	CxList db = Find_DB(); CxList read = Find_Read_NonDB(); CxList outputs = Find_XSS_Outputs(); CxList sanitize = Find_XSS_Sanitize(); result = All.FindXSS (db + read, outputs, sanitize);
XPath_Injection	CxList XPath = All.FindByMemberAccess("XPath.compile") + All.FindByMemberAccess("XPath.evaluate"); CxList inputs = Find_Interactive_Inputs(); CxList sanitized = Find_Sanitize(); result = XPath.InfluencedByAndNotSanitized(inputs, sanitized);

3.3 Reviewing Source Code

After creation of training set we are creating N-gram signatures for test set application using reverse engineering process and it is extracting features. Apk2java command is used to get the all java code and it is running from the scratch. Create a java file and compile it .Using these class files, create a jar file and run a Java Archive (JAR) file for reviewing the source code using java –jar filename.jar command. After executing this command one Comma Separated Value (CSV) file is created in the folder. A CSV file is any file containing text that is separated with a comma, or any other character.CSV file contains Line number, Location of the program, Vulnerability name and code of the vulnerability. N-gram source codes signatures are not able to store the SQLite database or other databases because huge space is need to store N-gram signatures of source code. So we are maintaining these signatures as Comma Separated value (CSV) file. It will be classify the application based on database files whether it is malware or benign applications. After executing above all steps and take the marked java file and import it using eclipse and check the source code has the issues or not. Figure 4 shows the test set application which ready for reverse engineering process.

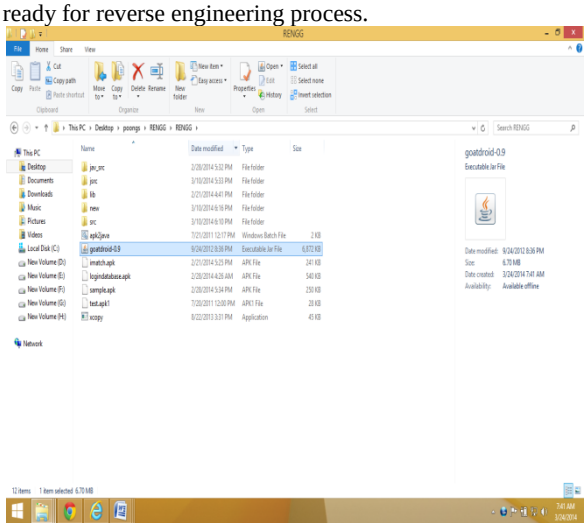

Fig .4 Test Set applications

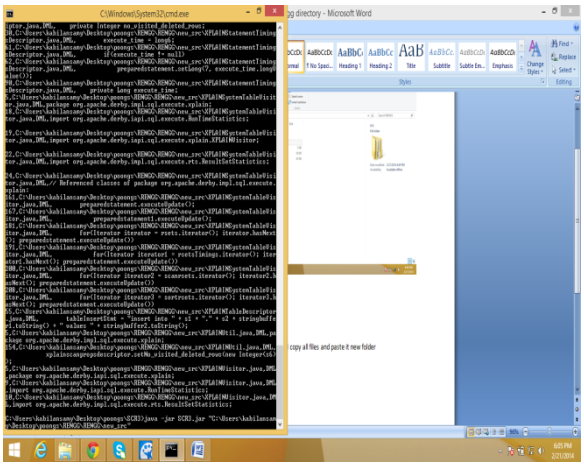

Fig. 5 Execution of Source Code Review file

Figure 5 show the execution of source code review of an application and figure 6 shows the output of an application which has the vulnerability.

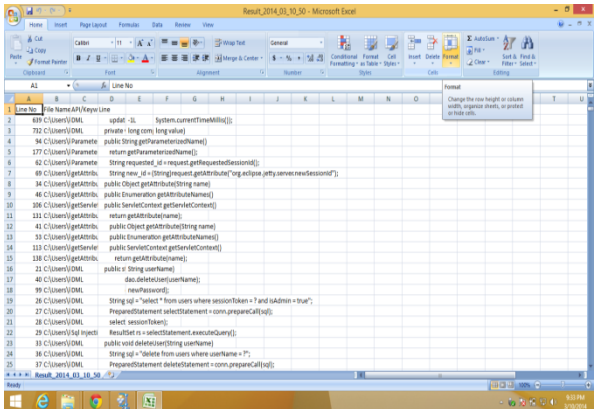

Fig.6 Result of the apk file

4. PERFORMANCE EVALUATION

Common Vulnerability Scoring System (CVSS) is a tool which is used to assess the vulnerabilities' severity level in software applications and it will be available in the internet and use it freely. This method uses the [12] CVSS to check the severity levels of vulnerabilities in the android application package files. To find the vulnerabilities in the android environment, source code analysis is used. After using this take the corrective action against vulnerabilities in the apk files to reduce the malware activities in the android based mobile devices [11]. The figure 7 shows the analysis part of a given applications.

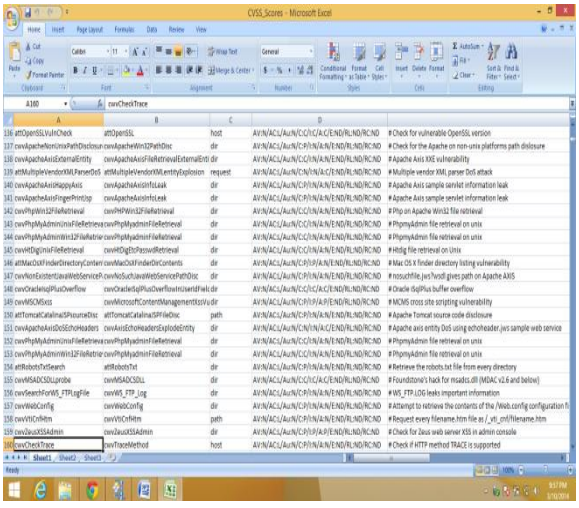

Fig.7 Analysis using CVSS

The column chart shows that number of applications and its CVSS scores. After reviewing the source code, it is evaluated by CVSS calculator. The calculator calculates the apps based on the base scores like Access vector, Authentication, Confidentiality, Integrity, Availability, etc. After getting the value, it shows that severity level of user environment in the mobile apps. The figure 8 shows the severity levels of some sample apps.

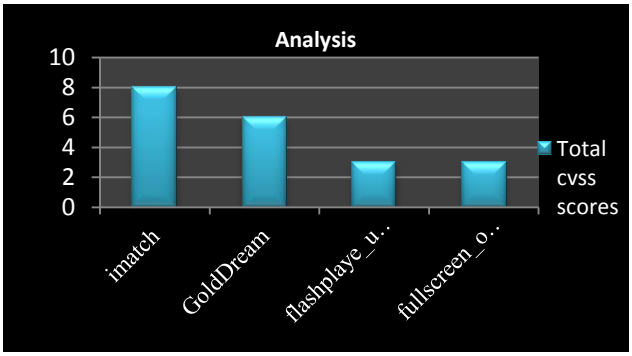

Fig.8 Severity levels of mobile apps

The table demonstrates the types of API calls used by the applications. Our source code reviewing approach discloses that samples of vulnerable API calls in the mobile applications. More than 100 applications are used to find the vulnerabilities in the application. Some samples are listed in the following table 2.

Table 2.Vulnerabilities in mobile apps

Mobile Applications Name	Vulnerable API Calls
imatch.apk	161
Golddream.apk	83
Flashplaye_uxiedbnx.apk	120
com.asurion.speeddialer.apk	78
contacts.Blast.1.8.apk	57

5. CONCLUSION

This method is supported to find the vulnerabilities/malicious characteristics in the mobile applications which are based on android based operating system. Signature based detection mechanism has two drawbacks 1. It fails to identify new malwares 2. Need huge database to store signatures. Our method achieves these drawbacks with the help of Source Code Analysis and N-gram analysis. In our method source code are considered as signatures. We can't able to take entire code for signatures. To avoid this we had chosen N-gram analysis and also it is used to detect new malicious characteristics in the application. Static analysis is used to find vulnerabilities in the mobile application before delivering it to end user. Common Vulnerability Scoring System which is used to evaluate the severity level of the vulnerabilities in application.

6. REFERENCES

[1] Kirti Mathur,Saroj Hiranwal, 2013 A Survey on Techniques in Detection analyzing malware executables" International Journal of Advanced Research in Computer Science and Software Engineering, Vol.3,Issues 4,pp 422-428.

[2] P.Vinod,V.Laxmi, M.S.Gaur,2009 Survey on Malware Detection Methods,3rd Hackers Workshop on Computer and Internet Security,Department of Computer Science Engineering, Prabhu Goel Research Centre for Computer and Internet Security,IIT,Kanpur,PP.74-79.

[3] A.Bose, X.Hu Kang,G.Shin and T.Park, 2009 Behavioral Detection of Malware on Mobile Handsets, IEEE International Conference on Mobile Systems, Applications, Services , pp 225-238.

[4] Marwa M.A.Elfattah,Aliaaa A.A.Yousif and Ebada sarhan amhed,2011 Handsets Malware Threats and Facing Techniques,International Journal of Advanced Computer Science and Applications , Vol.2,No.12,pp 42-48.

[5] Aubrey-Derrick Schmidt, Rainer Bye, Hans-Gunther Schmidt, Jan Clausen, Osman Kiraz , Kamer Ali Yuksel, Seyit Ahmet Camtepe, and Sahin Albayrak, 2009 Static Analysis of Executables for Collaborative Malware Detection on Android, IEEE International Conference on Communications,pp: 1-5.

[6] Dex2Jar [Online] Available: https://code.google.com/p/dex2jar/downloads/list Date Accessed: 2014 January.

[7] Android APKTOOL [Online], Available: http://code.google.com/p/android-apktoolDate Accessed: 2014 January.

[8] Java Decompiler [Online], Available: http://java.decompiler.free.fr/?q=jdgui, Date Accessed: 2014 January.

[9] APK2JAVA [Online], Available: https://code.google.com/apk2java/downloads/detail?name=apk2java_v_1.0.zip,Date accessed:2014 January.

[10] Igor Santos,Yoseba P.Kenyas,Jamie Devesa and Pablo G.Bringas, 2009 N-gram Based File Signatures for Malware Detection , International Conference on Enterprise Information Systems(ICEIS), pp 317-320.

[11] Common Vulnerability Scoring System (CVSS) [Online] Available:https://nvd.nist.gov/cvss.cfm?calculator&adv&version=2&vector

[12] Assad Ali, Pavol Zavarsky, Dale Lindskog, Ron Ruhl , 2011 A New CVSS-Based Tool to Mitigate the Effects of Software Vulnerabilities, International Journal for Information Security Research (IJISR), Volume 1, Issue 4, and pp: 178-182.

[13] R.Dhaya and M.Poongodi, 2014 Mobile Virus Prevention Techniques: A Survey Perspective, International Journal of Innovative Research in Computer and Communication Engineering,Vol 2,Issue 1,pp.1980-1985.

[14] R.Dhaya and M.Poongodi, 2014 "Preventing and Controlling Virus Dissemination in Mobile Devices", Proceedings of the International Conference on Science and Innovative Engineering, ISBN 978-81-904760-5-8.

7. AUTHOR'S PROFILE

Dr.R.Dhaya received her Bachelor of Engineering Degree in Computer Science and Engineering in 2004. Then she obtained his Master of Engineering Degree in Embedded System Technologies in 2006 from Anna University, Chennai, India. She completed his Ph.D in Computer Science and Engineering from M.S.University, Tirunelveli in 2013.She is currently working as Assisatnt Professor in the Department of Computer Science and Engineering at Velammal Engineering College, Chennai. She has published books for engineering students. She has published research papers in reputed international journals and presented her research works in various international conferences. Her interests include Embedded Systems, Wireless Communication systems and Computer Networks.

M.Poongodi obtained her Bachelor's Degree (2004) in Information Technology from Madras University, India. Now she is doing her Post Graduate (second year) in Computer Science and Engineering from Velammal Engineering College, Chennai under Anna University. Her research interests are Mobile Security and Operating Systems.