ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/314086348

AndroData: A Tool for Static & Dynamic Feature Extraction of Android Apps

Article *in* International Journal of Applied Engineering Research · January 2015

CITATIONS
5

READS
112

2 authors:

Sapna Malik
Maharaja Surajmal Institute Of Technology
10 PUBLICATIONS **24** CITATIONS
SEE PROFILE

Kiran Khatter
Ansal University
22 PUBLICATIONS **49** CITATIONS
SEE PROFILE

All content following this page was uploaded by Sapna Malik on 01 March 2017.

The user has requested enhancement of the downloaded file.

International Journal of Applied Engineering Research, ISSN 0973-4562 Vol. 10 No.94 (2015)
© Research India Publications; http/www.ripublication.com/ijaer.htm

AndroData: A Tool for Static & Dynamic Feature Extraction of Android Apps

Sapna Malik
School of Engineering and Technology
Ansal University, Gurgaon
Haryana, India
sapnadhankhar@gmail.com

Kiran Khatter
School of Engineering and Technology
Ansal University, Gurgaon
Haryana, India
kirankhatter@ansaluniversity.edu.in

***Abstract*—Being an open source software Android is a popular Operating System in market for smartphones. System calls play an important role in the execution of user programs in the Android OS. We can judge the behavior of user program by tracing its system calls. This paper presents an automatic system call hijacking and permission extraction tool,AndroData which works in kernel space for tracing system calls executed by the android application and records permission used by Android Application.**

Keywords—Intrusion Detection,Features Extraction, Android Application,System call,Permission System.

I. INTRODUCTION

The Smartphones are getting immense popularity all over the world. The smartphone has become almost everyone's online bank, online shopping mall, and online tutor along with traditional voice communication services. Mobile devices are equipped with advanced user interfaces, processing capability and adequate memory. Mobile devices haul lots of personal information like contact list, online banking passwords, credit card details and location of the proprietor. These features of Smartphone have attracted intrusions to compromise the mobile device and steal mobile data. Kruegelet. Al. [1] defined an intrusion as "a sequence of related actions performed by a malicious adversary that results in the compromise of confidentiality, integrity or availability of a target system". Blackberry, iOS, Android, Window Phones are the OS's for mobile devices. The Android operating system is used in mobile phones, tablets and Smart TV. Because of being an open source operating system, the android apps market is on boom. Intrusions are making malicious android applications which are difficult to detect and prevent intrusion because of limitation of Android Operating System over other mobile OS's. Moreover Android Operating System has multiple manufacturers, whichlead to lack of updated security patches. There are two types of intrusion detection techniques - misuse detection and anomaly detection technique. In misuse detection technique the intrusion is defined based on predefined signature. In anomaly intrusion detection technique, we analyze the behavior of intrusion through the events raised by intrusion and detect intrusion.

In anomaly detection technique, the software analysis is of two types, Static Analysis and Dynamic Analysis. In Static Analysis, the application analyses without executing, just by analyzing the code and finding the possible API that the application can invoke. In Dynamic Analysis the application is analyzed by executing the code. Intrusion Detection process for mobile is divided into two parts,feature extraction of android application and analysis of these features. This paper presents a tool-AndroData, an automatic static and dynamic feature extraction tool implemented on the Android Platform. This paper is organized into ninesections. The Section II explores Android Operating System from security aspect. The Section III discussed Related Work. The Section VI and Section V give detailed explanation of AndroDatatool. The Section VI does the comparison of AndroData with other Exiting Tool. The Section VII and SectionsVIII are Result and discussion Section respectively. The Section IX concludes the Work and gives the Future Scope.

II. ANDROID OPERATING SYSTEM

Android is Linux based open source operating system designed to run on energy and resource constrained mobile devices. In order to do malware analysis first we have to understand the architecture of Android Platform. The central component of Android OS is a kernel that acts as an interface between hardware and software. The application framework communicates with the kernel with the help of System Call. The System Call is the special low level I/O functions that help in passing arguments of executing program to kernel and enable the execution of the program. The Android operating system has around 220 System Calls. The principal processor of android device is ARM platform. For executing a tool in android we have to generate executable code for the ARM platform. Above the kernel there are C/C++ open source libraries.One of the important components in this layer is SQLite, which holds data of devices including confidential data. This component can be the prime target for malicious application if it is not properly protected. For executing an application Android has its own run time environment called Dalvik Virtual Machine. Every class file of Java applications for Android has to translate into.Docs file with the help of DX tool for running it on the Dalvik runtime environment. The next layer is an application framework that provides documented API which helps in understanding Java code of the application. Java is used for developing application for android device called as an Android Application package. An Android Application Package (.apk) has the components - manifest, classes.dex, resources, broadcast receiver, activities, services, content provider. Each component plays an important role that

help in analysis of a malicious application. Manifest is an XML file that declares all the components, requirement and permissions required by application for its execution. Classes.dex holds the Dalvik Byte Code, which can help with static analysis of malicious application by reverse engineering.

III. Related Work

Android is topic of research since it was released in 2008.For doing research in area of the Android Security first you need a dataset of the features of android applications for doing analysis of android application. Many researchers have developed their own tools for extracting various dynamic and static features of android apk files.Enck et al. [2] were thefirst ones who threw light on a permission model of content, services and URI pioneer and interaction between various components of Android Apk files. Schmidt et al. [3] used readelf commands for doing static analysis with function call in android environment. They had just analyzed log files of android phones for detecting malicious activities on Androidphones. Kirin [4] proposed a set of logical rules for certification of android application at installation time. They had taken samples of 311 applications for testing.SCanDroid [5] developed a Java program for static analysis of Android applications by extracting security specification from manifest file and analyzing the data flow through the components of android apps. TaintDroid, [6] is a real time solution implemented upon android mobile platform that keeps track ofthe flow of sensitive data like IMEI number.They have tested it with 30 applications.DroidBox [7] is another TaintDroid based dynamic tool developed with shell scripting for analysis of apk based files on the features of the API call, file system and Network activity and cell phone. In DroidBox user has to specify the path to apk file whose analysis he wants to do. It is not automated tool that can do an analysis of a batch of apk files of its own. ComDroid [8] is another static analysis tool that controls flow analysis of inter- application communication by the decompiled bytecode of android application. They have analyzed 20 applications for verifying the ComDroid Application. Iker Burgueraet. al. proposed Crowdroid [9],dynamic analysis tool based on feature System call issued by an application and finding the intrusion detection based on the sensitive system call executed by the malicious application. DroidRanger [10] is permission based behavioral foot printing and heuristics-based filtering tool for detection of malicious android application. The researchers have analyzed 2,04040 applications with their tool and helped in finding the new android malicious families. DroidMat [11] is an intrusion detection tool with permission, intent messages and API call tracing using machine learning. Anubis [12] is the first web based static and dynamic tool which gives a detailed report of features and malicious rating of an uploaded android application. MADAM[13] (Multilevel Anomaly Detector for Android Malware) is a host based tool used to detect intrusion at kernel and user level using feature using 12 features. Andromaly [14] is another host-based malware detection system which uses machine learning techniques for doing analysis of malicious applications.

Kim et al. [15] proposed intrusion detection tool which uses j48 decision tree for classification of malicious and benign application. They have designed an automatic featureextraction tool written with Javascripts which can extract two features Permission and method API. Huang et al. [16] suggested a machine learning approach for classification of malicious application based on permission feature.Yerima et al. [17] present static malicious detection approach based on the features API, permission and command. They have used his own tool with a component API detector, Command Detector and permission Detector. Canfora et al. [18] suggested a malicious classification approach based on system calls.Feizollah et al. [19] used three network features connection duration, number of GET/POST parameters and TCP size for classification of intrusion detection. Narudin et al. [20] proposed an approach for classification of malicious application network traffic feature.

There are also many repositories available at github.com for android intrusion detection by extracting features of android application. AndroWarn [21] is static dalvik byte code analyzer tool which does structural and data flow analysis. It can do analysis of one apk file at a time. It is a python script. ApkAnalyzer [22] is another static bytecode analyzer written in Java for analyzing one apk file at a time.Apkinspector [23] is GUI based shell script code for doing static analysis of function and graphic features analysis of malicious apk files. It also does permission analysis. It can analyze one apk file at a time.DidFail (Droid Intent Data Flow Analysis for Information Leakage) [24] is a static analysis tool for detecting dataflow of sensitive information between a set of application.Yee Au et. al. suggested a permission analysis PScout[25] tool which relates the permission requested with their intent and method calls. There are many other tools like Smali CFG generator[26],FlowDroid[27],Amandroid[28],SmaliSCA[29],CFGScanDroid[30], Android DBI frameowork[31],Android Malware Analysis Toolkit[32],AppUse[33],CobradroidViaLabCommunity Edition[34],Mercury[35],Drozer[36],Xposed[37],Android Hooker[38], Android tamer[39],Droidscope[40],CuckooDroid[41],Mem[42],Auditd Android[43],Android Security Evaluation Framework[44],Android Reverse Engineering – ARE (android reverse engineering)[45] are available in market which can do complete static analysis and dynamic analysis of apk file but there is no supporting tool available in market which can extract only the features of many apk files without human intervention for doing analysis of thousands of apk files. So to the best of my knowledgeAndroData is unique supporting tool for research community for extracting features of multiple apk files for doing analysis of apk files.

IV. AndroData

As discussed earlier System Callisthe interface between program and the Linux kernel. Some system calls are used for allocating resources,some are used for providing restriction to kernel functions, some are used to read and write sensitive data, some are used to provide access to confidential data to other user. Most of the system call can be executed by super user only. A malicious application can use these system calls as a powerful tool to take control of the mobile device and to harm the mobile device resources either by SMS flooding to an unknown person or by consuming battery of the mobile device or stealing the confidential data of the user.As we know application communicates with kernel with the help of system calls. Thus, we can observe the behavior of an application by tracing its system call.

The algorithm 1 shows the overall algorithm of the AndroData. In this algorithm after the installation of application on the Emulator, the user defined function ‘AppSimulator’ is called to simulate events of the application. The user defined functions ‘GetSystemCall’ and ‘Permission’ are used for extraction of system call and permission feature of the application respectively.

Algorithm 1. Overall AlgorithmofAndroData.

INPUT: N no of apk files.

OUTPUT: R no of reports

R: =0

For i := 1 to N

Install (i.apk)

Package_name:=get package name(i)

AppSimulator(Package_name)

Process_id[i]:=GetProcessId(i)

SystemCall: =GetSystemCall(Process_id)

Permission: =permission (Package_name)

R: =append (Package_name, SystemCall,Permission);

end for

AndroData tool runs in kernel space for accessing the system calls log. Reports generated by AndroData have the information about the type of system call called and how many times a system call is executed by the application. AndroData tools also gather information about permission used by the application under observation by assessing permission model of the application. The permission function is used this purpose. AndroData permission model can access the permission related to 121 permission of Android operating System. AndroData can be executed on a PC as well as Android Phone for accessing features of Android Application.Figure 1 shows the snapshot of the AndroData.

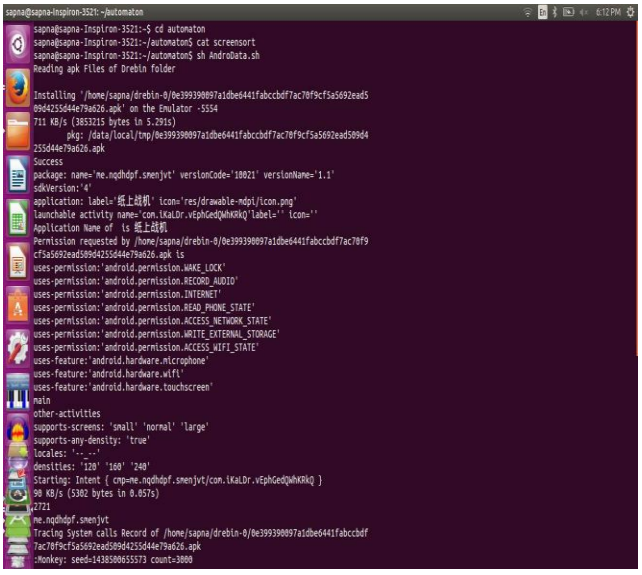

Fig. 1. Screenshot of AndroData

V. COMPONENTS OF ANDRODATA

The Figure 2 shows the components of the AndroData tool. Now lets us consider these components one by one.

A. *Installer*

As AndroDatais supposed to extract features of multiple Android Applications, Installer automatically installs the apk filesstored in a folderon the emulator. The 'adb install' command is used for this purpose.

B. *Application Simulator*

The main function of Application Simulator is to start intents and activities of the application. The monkey tool is a main functional element in Application Simulator. The monkey tool is the android testing tool that runs on android device and generates streams of user events such as clicks, touches or gestures and system level events. It is adb shell command called with the package name of the application as an argument.

C. *System Call Tracer*

The System Call Tracer traces the system calls used by the application during its running. In system call tracer the strace command is used for tracing system calls used by application & its count. The strace command is a diagnostic tool which runs a specified command till the command stops.When strace command is called with the process id of the application with its –p option,it intercepts and record the system calls called by the process by attaching to this process. It also records the signals received by the process.

D. *Permission Access System*

The function of permission model is to extract the information about permission used by its apk file. The permission Access system usesaapt dump command for this purpose. The aapt, Android Asset Packaging Tool is Android SDK tool used to view, create and update apk files. The aapt dump command is used for accessing information from manifest.xml file of apk file.

E. *Uninstaller*

When the static and dynamic features of an application are successfully extracted, we need to uninstall the application from the emulator to avoid the memory full error while extracting features of other application. The adb uninstall command is used for uninstalling the application.

After the execution of one instance of AndroData the report will be generated for the application under observation. The reports will be saved in the same folder in which apk files reside. The report contains the data- name of the system call and its count and permission used by the application.

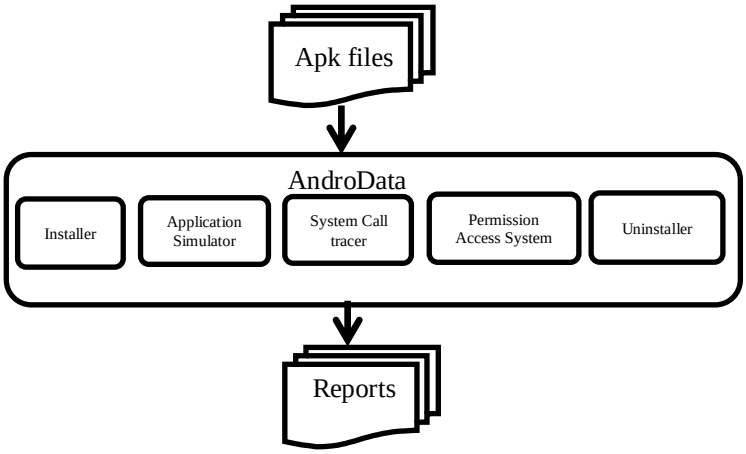

Fig. 2. Components of AndroData

Total time	15.9	119.9	18.1

VI. EXITING TOOLS COMPARISON

In the related work section we have discussed researches done in this area, but only a few researches are available in tool form currently.Most of the tools specified in the research are dead now or not available in the market for other researchers. In this section we will discuss some of the best tools available in the market with their pros and cons.Anubis [12] and NVISO [46]apkscan are two very good online tools which do static and dynamic analysis of android malicious application.Anubis does analysis based on features likeactivities,permission, fileoperations, started service,SMSsent, etc. and its report has data about these features.NVISO apkscan generates report having data from Android Manifest,Virus Total Scan result,Hardcoded URL's,Disk Activities, and Information Leakage.Anubis can analyze apk files of maximumsize 8MB whereas NVISO apkscan has no limit on the size of apk file that needs to be analyzed.As both the tools are online, user needs an internet connection and has to upload its apk file on website which makes it difficult to analyze a bulk of apk's file.ApkAnalyzer [22] is anoffline static, virtual analysis tool for analyzing binary code of android apps.It is developed in Java and can do analysis of one android app at a time.Apkinspector [23]is another powerful GUI tool for analyzing static features of apk files fromdecompiling the bytecode of apkfiles. DroidBox [7] is an offline dynamic analyzer tool which can analyze the apk file based on the features- network traffic, fileoperations,started service and loaded classes, cryptographicoperations, informationleaks, etc.ApkAnalyzer,Apkinspector, DroidBox all are offline tools which can do analysis of apk file on number of features but these tools can do analysis of one android app at a time. These tools cannot take multiple apk files as an input and generate their features.AndroData is an offline tool which has the capability of generating static and dynamic features of multiple android apps which will helpthe research community in analyzing these features.

VII. RESULT

AndroData, a static and dynamic analysis tool for feature extraction, is developed in shell scripting on Ubuntu 12.4 platform. Asthis tool requires application to be installed on a mobile device or emulator, so we require android SDK to be installed on your PC. We have tested it with the data set of Drebinproject [47] [48]. We have tested it with android applications of sizes 3KBto20MB. The running time performance of AndroData is shown in Table 1.As in Static features extraction, we just extract the features of .apk file without installing it, thus it requires lesser time, whereas extraction of Dynamic features of an Android application requires the app to run and to simulate user triggered events. Thus, Dynamic analysis requires more time. The experiment results depict the same fact.

Table 1. RUNNING TIME OF ANDRODATA (IN SECONDS)

Functionality	Min Time (Seconds)	MaxTime (Seconds)	Average Time (Seconds)
Static Feature Extraction	2.7	6.7	3.4
Dynamic Feature Extraction	13.2	113.2	14.7

VIII. DISCUSSION

AndroData is a lightweight application developed with shell scripting.AndroData can be used on PC as well as android phone for static and dynamicfeature extraction of an android application. Androdata has the following features.

A. *Features of AndroData*

1) As it is developed in shell scripting language,it does not need any special software for its execution.
2) AndroData can extract features of a batch of Android Application (System call & permission) automatically without human intervention. User needs to specify the path of the folder where all the apk files stored whose features we want to extract. Reports will automatically save in the same folder.
3) AndroData work in Kernel as well as in the user space for extracting static and dynamic features of Android Application.
4) AndroData is a useful feature extraction tool for the research community for extracting features of thousands of applications automatically.
5) AndroData can extract features of apk file of any size.

B. Limitation of AndroData

1) In AndroData the apk application is simulated with Monkey tool. The Monkey tool is not suited for genrating the events which require human intelligence or inputs which control application functionality.It is also unable togenerate System events and UI events.
2) AndroData can extract only two features system call and permission of android application.

IX. CONCLUSION & FUTURE SCOPE

AndroData is lightweight tools for extracting features of a bunch of Android Applications. It allows the user to sit back and get the features extracted for thousands of application features in a few hours instead of months and help in availing large set of dataset for analysis of malicious applications. It is very useful tool for researcher. Till now AndroData can extract only two features System Calls and Permission.So we can extend this tool with the ability to extract more features like API call, Function call etc.

REFERENCES

[1] C. Kruegel, F. Valeur, G. Vigna,Intrusion Detection and Correlation: Challenges and Solutions., Springer, 2005.

[2] William Enck, Machigar Ongtang, and Patrick Drew McDaniel. "Understanding Android Security". IEEE Security & Privacy, 7 (1): 50–57, 2009.

[3] Aubrey-Derrick Schmidt, Rainer Bye, Hans-Gunther Schmidt, Jan Clausen, Osman Kiraz, Kamer A. Yuksel, Seyit A. Camtepe, and Sahin Albayrak.Static Analysis of Executables for Collaborative Malware Detection on Android. In Proceedings of the 2009 IEEE International Conference on Communications, ICC'09, pages 631–635, Piscataway, NJ, USA, 2009. IEEE Press.

[4] William Enck, Machigar Ongtang, and Patrick McDaniel. On Lightweight Mobile Phone Application Certification. In Proceedings of the 16th ACM Conference on Computer and Communications Security, CCS '09, pages 235–245, New York, NY, USA, 2009. ACM.

[5] Adam P. Fuchs, Avik Chaudhuri, and Jeffrey S. Foster. SCanDroid: Automated Security Certification of Android Applications. Technical Report CS-TR-4991, Department of Computer Science, University of Maryland, College Park, November 2009.

[6] William Enck, Peter Gilbert, Byung-Gon Chun, Landon P. Cox, Jaeyeon Jung, Patrick McDaniel, and Anmol N. Sheth. TaintDroid: An Information-flow Tracking System for Realtime Privacy Monitoring on Smartphones. In Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation, OSDI'10, pages 1–6, Berkeley, CA, USA, 2010. USENIX Association.

[7] P. Lantz, A. Desnos, and K. Yang. Droidbox: Android application sandbox. Http's: //code.google.com/p/droidbox/

[8] Erika Chin, Adrienne Porter Felt, Kate Greenwood, and David Wagner. Analyzing inter-application communication in android. In Proceedings of the 9th International Conference on Mobile Systems, Applications, and Services, MobiSys '11, pages 239–252, New York, NY, USA, 2011. ACM.

[9] Iker Burguera, Urko Zurutuza, and Simin Nadjm-Tehrani. Crowdroid: Behavior- based malware detection system for android. In Proceedings of the 1st ACMWorkshop on Security and Privacy in Smartphones and Mobile Devices, SPSM '11, pages 15–26, New York, NY, USA, 2011. ACM.

[10] Yajin Zhou, ZhiWang,Wu Zhou, and Xuxian Jiang. Hey, you, get off of my market: Detecting malicious apps in official and alternative Android markets. In Proceedings of the 19th Annual Network & Distributed System Security Symposium, February 2012.

[11] Dong-Jie Wu, Ching-Hao Mao, Te-En Wei, Hahn-Ming Lee, and Kuo-Ping Wu. DroidMat: Android Malware Detection through Manifest and API Calls Tracing. In Information Security (Asia JCIS), 2012 Seventh Asia Joint Conference on, pages 62–69, Aug 2012.

[12] LukasWeichselbaum, Matthias Neugschwandtner, Martina Lindorfer, Yanick Fratan- tonio, Victor van der Veen, and Christian Platzer. Andrubis: Android Malware Under The Magnifying Glass. Technical Report TR-ISECLAB-0414-001, Vienna University of Technology, 2014.

[13] Gianluca Dini, Fabio Martinelli, Andrea Saracino, and Daniele Sgandurra. Madam: A multi-level anomaly detector for android malware. In Proceedings of the 6th International Conference on Mathematical Methods, Models and Architectures for Computer Network Security: Computer Network Security, MMM-ACNS'12, pages 240–253, Berlin, Heidelberg, 2012. Springer-Verlag.

[14] Asaf Shabtai, Uri Kanonov, Yuval Elovici, Chanan Glezer, and YaelWeiss. Andro- maly: A Behavioral Malware Detection Framework for Android Devices. Journal of Intelligent Information Systems, 38(1):161–190, February 2012.

[15] Dong-uk Kim, Jeongtae KIm, and Sehun Kim. A malicious application detection framework using automatic feature extraction tool on android market. In Proceedings of the 3rd International Conference on Computer Science and Information Technology, ICCSIT 2013, 2013.

[16] Chun-Ying Huang, Yi-Ting Tsai, and Chung-Han Hsu. Performance evaluation on permission-based detection for android malware. In Jeng-Shyang Pan, Ching-Nung Yang, and Chia-Chen Lin, editors, Advances in Intelligent Systems and Applications - Volume 2, volume 21 of Smart Innovation, Systems and Technologies, pages 111–120. Springer Berlin Heidelberg, 2013.

[17] S.Y. Yerima, S. Sezer, G. McWilliams, and I. Muttik. A New Android Malware Detection Approach Using Bayesian Classification. In Advanced Information Net- working and Applications (AINA), 2013 IEEE 27th International Conference on, pages 121–128, March 2013.

[18] G. Canfora, F. Mercaldo, and C.A. Visaggio. A classifier of malicious android appli- cations. In Availability, Reliability and Security (ARES), 2013 Eighth International Conference on, pages 607–614, Sept 2013.

[19] Ali Feizollah, Nor Badrul Anuar, and Rosli Salleh. A Study Of Machine Learning Classifiers for Anomaly-Based Mobile Botnet Detection. Malaysian Journal of Computer Science, 26(4):251–265, 2013.

[20] Fairuz Amalina Narudin, Ali Feizollah, Nor Badrul Anuar, and Abdullah Gani. Eval- uation of machine learning classifiers for mobile malware detection. Soft Computing, 2014.

[21] https://github.com/maaaaz/androwarn/

[22] https://github.com/sonyxperiadev/ApkAnalyser

[23] https://github.com/honeynet/apkinspector/

[24] https://www.cert.org/secure-coding/tools/didfail.cfm

[25] http://pscout.csl.toronto.edu/

[26] https://code.google.com/p/smali-cfgs/

[27] http://sseblog.ec-spride.de/tools/flowdroid/

[28] http://amandroid.sireum.org/

[29] https://github.com/dorneanu/smalisca

[30] https://github.com/douggard/CFGScanDroid

[31] http://www.mulliner.org/blog/blosxom.cgi/security/androiddbiv02.html

[32] http://www.mobilemalware.com.br/amat/download.html

[33] https://appsec-labs.com/AppUse

[34] http://thecobraden.com/projects/cobradroid/

[35] http://labs.mwrinfosecurity.com/tools/2012/03/16/mercury/

[36] https://labs.mwrinfosecurity.com/tools/drozer/

[37] http://forum.xda-developers.com/showthread.php?t=1574401

[38] https://github.com/AndroidHooker/hooker

[39] https://androidtamer.com/

[40] https://code.google.com/p/decaf-platform/wiki/DroidScope

[41] https://github.com/idanr1986/cuckoo-droid

[42] https://github.com/MobileForensicsResearch/mem

[43] https://github.com/nwhusted/AuditdAndroid

[44] https://code.google.com/p/asef/

[45] https://redmine.honeynet.org/projects/are/wiki

[46] http://apkscan.nviso.be/

[47] Daniel Arp, Michael Spreitzenbarth, Malte Huebner, Hugo Gascon, and Konrad Rieck "Drebin: Efficient and Explainable Detection of Android Malware in Your Pocket", 21th Annual Network and Distributed System Security Symposium (NDSS), February 2014

[48] Michael Spreitzenbarth, Florian Echtler, Thomas Schreck, Felix C. Freling, Johannes Hoffmann, "MobileSandbox: Looking Deeper into Android Applications", 28th International ACM Symposium on Applied Computing (SAC), March 2013

ABOUT THE AUTHORS

Sapna Malik is a Ph.D. candidate at the School of Engineering and Technology, Ansal University, India. She holds M.Tech in IT from GGSIPU, New Delhi, India and B.Tech in CSE from Maharishi Dayanand University, India. She is working as Assistant professor in department of Computer Science Engineering in Maharaja Surajmal Institute of technology, Delhi, India. Her research interest includes Cloud computing, Network Security and Virtualization.

Dr. Kiran Khatter has done M.Tech(IT) from Punjab University,Patiala and Ph.D in Computer Science from Himachal Pradesh University,Summer Hill.She has number of publications in journal & international conferences proceeding of repute.She is an Oracle Certified Java SE 6 Programmer.She has done training in ABAP programming and handled ABAP Projects.Her research interests span over Software Engineering, Image Processing,Data Mining and Big Data Analytics.

View publication stats