Research Journal of Computer and Information Technology Sciences _____ISSN 2320 – 6527
Vol. **1(6),** 12-19, November **(2013)** *Res. J. Computer and IT Sci.*

Review Paper

Review on Android and Smartphone Security

Tiwari Mohini, Srivastava Ashish Kumar and Gupta Nitesh
NRI Institute of Information Science and Technology, Bhopal,Madhya Pradesh, INDIA

Available online at: www.isca.in, www.isca.me
Received 25th October 2013, revised 4th November 2013, accepted 19th November 2013

Abstract

Android has the biggest market share among all Smartphone operating system. Security is one of the main concerns for Smartphone users today. As the power and features of Smartphone's increase, so has their vulnerability for attacks by viruses etc. Perhaps android is more secured operating system than any other Smartphone operating system today. Android has very few restrictions for developer, increases the security risk for end users. In this paper we have reviewed android security model, application level security and security issues in the Android based Smartphone.

Keywords: Android security, smartphone security, malware.

Introduction

Android is a modern mobile platform that is designed to be truly open source. Android applications can use advanced level of hardware and software, as well as local and server data, exposed through the platform to bring innovation and value to consumers. Android platform must have security mechanism to ensure security of user data, information, application and network[1].

Open source platform needs strong and rigorous security architecture to provide security. Android is designed with multi-layered security that provides flexibleness needed for an open platform, whereas providing protection for all users of the platform designed to a software stack, android includes an operating system, middleware and core application as a complete[2]. Android powers hundreds of millions of mobile devices in more than 190 countries around the world.

Android architecture is designed with keep ease of development ability for developers. Security controls have designed to minimize the load on developers. Developers have to simply work on versatile security controls. Developers are not familiar with securities that apply by defaults on application.

Android is also designed with focused on user's perspective. Users can view how applications work, and manage those applications.

Android Platform Security Architecture

Android seeks to be the most secure and usable operating system for mobiles by re-purposing classical operating system security controls to protect user data, system resources and provide application isolation.

Android provides following security features to achieve these objectives are first robust security at the operating system level through the Linux kernel, second compulsory application sandbox for all applications, third secure interposes communication, fourth application signing, and sixth application defined permission and user have to grant permissions.

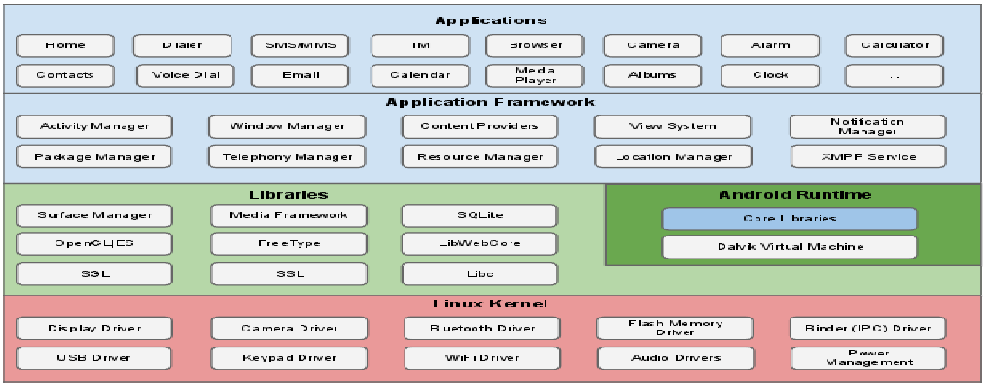

Figure-1
Android Architecture

Figure 1 summarizes security components and considerations at the various levels of the Android. Every component assumes that component below is properly secured. With exception some Android operating system code running as root, all process run above the Linux Kernel is restricted by the Application Sandbox.

Security in Android: i. Android is open source platform, developers will work along to enhance it[1]. ii. Android platform is multitasking software; therefore no application will gain critical access to the components of OS[3]. iii. Android platform is UNIX based operating system that is the most secure operating system[1]. iv. The developers need a unique signature to publish their application on market[4]. v. Users will report a possible security flaw through their Google account. vi. All applications on android need permission from the user at the time of installation.

Security Issues faced by Android

Android is not secure as it appear, even when such robust security measures. There are several security problems faced by the android, some of them are mentioned below. i. Android has no security scan over the apps being uploaded on its market. ii. There are some apps which can exploit the services of another app without permission request. iii. Android's permission security model provides power to user to make a decision whether an app should be trusted or not. This human power introduces a lot of risk in Android system. iv. The Open Source is available to legitimate developers as well as hackers too. Thus the Android framework cannot be trusted when it comes to develop critical systems. v. The Android operating system developers clearly state that they are not responsible for the security of external storage. vi. Any app on the android platform will access device data just like the GSM and SIM marketer Ids while not the permission of the user.

Android platform provides all security features, but there will always be a risk if the user will install suspicious apps or allow permission to an app without paying attention.

Literature survey

W. Enck, D. Octeau, P. McDaniel and S. Chaudhuri present 'a study of Android application security'. They introduce the ded decompiler, which generate android application source code directly from its installation image. They design and execute a horizontal study of smartphone applications based on static analysis of 21 million lines of recovered code. Their analysis uncovered pervasive use / misuse of personal / phone identifiers, and deep penetration of advertizing and analytics networks[5].

S. Powar, Dr. B. B. Meshram, surveyed on 'Android security framework', in this paper, they described android security framework. Increased exposure of open source Smartphone is increasing the security risk. Android provide a basic set of permissions to secure phone. The technique to make Android security mechanism more versatile, the current security mechanism is too rigid. User has only two options at the time of application installation first allow all requested permissions and second deny requested permissions leads to stop installation[6].

S. Kaur and M. Kaur presented review paper on 'implementing security on Android application'. In that paper, they described how security can be improved in android based system so that users can safely use the android smart phones[2].

S. Smalley and R. Craig presented 'Security Enhanced (SE) Android: Bringing Flexible MAC to Android'. The android software stack for mobile devices defines and enforces its own security model for apps through its application-layer permissions model. However, at its foundation, android depends upon the UNIX operating system kernel to shield the system from malicious or imperfect apps and to isolate apps from each other. At present, android leverages UNIX operating system discretionary access control (DAC) to enforce these guarantees, despite the notable shortcomings of DAC. In this paper, they motivate and describe their work to bring flexible mandatory access control (MAC) to Android by enabling the effective use of Security Enhanced Linux (SELinux) for kernel-level MAC and by developing a set of middleware MAC extensions to the Android permissions model[7].

P. Gilbert, W. Enck, L.P. Cox, B.G. Chun, J. Jung, A.N. Sheth and P. McDaniel presented 'TaintDroid: An Information-Flow Tracking System for Real-time Privacy Monitoring on smartphones'. Now days smartphone operating systems often fail to provide users with adequate control over and visibility into how third-party applications use their private data. They address these shortcomings with TaintDroid, system-wide dynamic taint tracking and analysis system capable of at the same time tracking multiple sources of private data. TaintDroid display real-time analysis by leveraging Android's virtualized execution environment and Monitoring private data to inform use of third-party applications for phone users and valuable input for Smartphone security service firms seeking to identify misbehaving applications[8].

B. J. Berger, M. Bunke, and K. Sohr presented an android security case study with Bauhaus. In this paper, they discovered that firms and corporation now uses security software for code analysis to discover security problems in application. They carried out a case study on android based mobile in cooperation with a security expert and employed the reverse engineering tool-suite Bauhaus for security assessment. During the investigation they found some inconsistencies in the implementation of the Android security concepts. Based on the case study, they propose several research topics in the area of reverse engineering that would support a security analyst during security assessments[9].

M. Ongtang, S. McLaughlin, W. Enck and P. McDaniel study on 'Semantically Rich Application-Centric Security in Android'. In this paper, they augment the existing android operating system with a framework to meet security requirements. They proposed secure application interaction (Saint), an improved infrastructure that governs install-time permission assignment and their run-time use as dictated by application provider policy. Saint provides necessary utility for applications to assert and control the security decisions on the android platform[10].

A.D. Schmidt, H. G. Schmidt, J. Clausen, A. Camtepe, S. Albayrak, K. Ali Yüksel and O. Kiraz study on 'enhancing security of Linux-based android devices'. In this paper they present an analysis of security mechanism in Android Smartphones with a focus on Linux. The results of their analysis can be applicable to Android as well as Linux-based Smartphones. They analysed android framework and the Linux-kernel to check security functionalities. They surveyed well-accepted security mechanisms and tools which could increase device security. They provided details on how to adopt these security tools on Android platform, and overhead analysis of techniques in terms of resource usage[11]. Their second contribution focuses on malware detection techniques at the kernel level. They tested applicability of existing signature and intrusion detection methods in android platform. They focused in observation on the kernel, that is, identifying critical kernel event, log file, file system and network activity events, and making efficient mechanisms to monitor them in a resource restricted setting[11]. They presented a simple decision tree for deciding the suspiciousness of the application[11].

C. Marforio, A. Francillon, S. Capkun study on 'application collusion attack on the permission-based security model and its implications for modern smartphone systems'. In this paper they show technique in which permission based mechanisms are used on mobile platforms allows attacks by colluding applications that communicate over explicit and covert communication channels. These security bugs allow applications to indirectly execute operations that those applications, based on their declared permissions, should not be able to execute. Example operations include disclosure of user's private data (e.g., phone book and calendar entries) to remote parties by applications that do not have direct access to such data or cannot directly establish remote connections. They further showed that on mobile platforms users are not aware of possible implications of application collusion quite the contrary users are implicitly lead to believe that by approving the installation of each application independently, based on its declared permissions, will limit the damage that an application can cause [12]. In this title, they show that this is not correct and that application permissions should be displayed to the users differently, reflecting their actual implications.

A. Lackorzynski, M. Lange, A. Warg, S. Liebergeld, M. Peter presented 'L4Android: a generic operating system framework for secure smartphones'. In this title they present a generic operating system framework that overcome the need of hardware extensions to provide security in smartphones. They encapsulate smartphone operating system in a virtual machine, this framework allows highly secure applications to run side-by-side with the virtual machine. It is based on a state-of-the-art micro-kernel that ensures isolation between the virtual machine and secure applications[13].

T. Luo, H. Hao, W. Du, Y. Wang, and H. Yin work on 'attacks on WebView in the android system'. Web-View is an important element in android platforms, enabling smartphones and tablet apps to embed a simple but powerful browser within them. To achieve a much better interaction between apps and their embedded browsers, WebView provides variety of APIs, permitting code in apps to invoke the JavaScript code within pages, intercept their events, and modify those events, using these features; apps will become customized browsers for their required web applications. Currently, within the android market, 86 % of the top twenty most downloaded apps in ten various classes use WebView[14]. The design of WebView changes the landscape of the web, particularly from the security perspective. Two essential component of the Web's security infrastructure are weakened if Web-View and its APIs are used: the Trusted Computing Base (TCB) at the client aspect, and therefore the sandbox protection enforced by browsers. As results, several attacks may be launched either against apps or by them[14].

D. Barrera, H. Güne¸ S. Kayacık, P.C. van Oorschot, A. Somayaji study on 'a methodology for empirical analysis of permission-based security models and its application to android'. In the paper, the proposed methodology is of independent interest for visualization of permission based systems beyond current Android-specific empirical analysis. They provide some discussion identifying potential points of improvement for the android permission model, trying to increase quality where required without increasing number variety of permissions or overall complexity[15].

C. Gibler, J. Crussell, J. Erickson and H. chen case study on 'AndroidLeaks: automatically detecting potential privacy leaks in android applications on a large scale'. Under this paper, they have presented a static analysis framework for automatically searching potential leaks of sensitive data in android applications on a large scale. AndroidLeaks drastically reduces the number of applications and the number of traces that a security auditor must verify manually[16].

I. Burguera, U. Zurutuza, S. Nadjm study on 'Crowdroid: behaviour-based malware detection system for Android'. In this title they used earlier approaches for dynamic analysis of application behaviour for detecting malware in the android platform. The detector is embedded in framework for assortment of traces from limitless number of real users supported crowd sourcing. This framework has been demonstrated by analysing information collected in the central

server using two sorts of data sets: those from artificial malware created for test functions, and people from real malware found in the world. The technique is shown to be an effective means of analytic the malware and alerting the users of a downloaded malware. This method is avoiding the spreading of a detected malware to a larger community[17].

M.L. Polla, F. Martinelli, and D. Sgandurra presented 'a survey on security for mobile devices'. This title surveys the vulnerabilities and security solutions over year 2004-2011, by targeting on high-level attacks on user applications. They cluster existing approaches aimed to securing mobile devices against these kinds of attacks into many categories, on the basis of the detection principles, architectures, collected information and operating systems, particularly focusing on IDS-based models and tools. This categorization aim to provide a simple and concise view of the underlying model adopted by each approach[18].

W. B. Tesfay, T. Booth, and K. Andersson presented 'reputation based security model for android applications'. They have proposed a cloud based reputation security model as a solution which greatly mitigates the malicious attacks targeting the Android market[19]. This security solution uses unique user id (UID) which is assigned to each application in the android platform. This model stores the reputation of Android applications in an anti-malware providers cloud (AM Cloud). The experimental results witness that the proposed model can identify the reputation index of a given application and its potential of being risky or not[20].

T. Blasing, L. Batyuk, A. D. Schmidt, S. A. Camtepe, and S. Albayrak studied 'an android application sandbox system for suspicious software detection'. They have projected an AASandbox (Android Application Sandbox) that performs static and dynamic analysis on android apps to detect suspicious apps. Static analysis scans the software package for malicious patterns without installing it and the dynamic analysis executes the app in an isolated environment, i.e. sandbox, that intervenes and logs low-level interactions with the system. Both the sandbox and the detection algorithms can be deployed in the cloud, providing a quick and distributed detection of suspicious app in an app store similar to Google's Play Store. AASandbox might be used to improve the anti-virus apps available for the android devices[21].

T. Vidas, D. Votipka, N. Christin study on 'all your droid are belong to us: a survey of current android attacks'. In this title they look to Android as a specific instance of mobile computing. They first discuss the Android security model and some potential weaknesses of the model. They provide taxonomy of attacks to the platform demonstrated by real attacks that in the end guarantee privileged access to the device[22].

S. Holla, M. M Katti work on 'Android based mobile application development and its security'. In this title, they discuss a layered approach for android application development where they can develop application which downloads data from the server, also an Android Application Sandbox (AASandbox) which is able to perform both static and dynamic analysis on Android programs to automatically detect suspicious applications[23].

D. Feth, A. Pretschner proposed 'flexible data-driven security for android'. They propose an improved security system beyond the standard permission system. It is possible to enforce complex policies that are built on temporal, cardinality, and spatial conditions in this system. Enforcement can be done by means of modification or inhibition of certain events. Leveraging recent advances in information flow tracking technology, policies can also pertain to data rather than single representations of that data[24].

G. Portokalidis, P. Homburg, K. Anagnostakis, and H. bos presented 'Paranoid Android: versatile protection for smartphones'. They propose a solution in which security checks are applied on remote security servers that host exact replicas of the phones in virtual environments. The servers are not subject to equivalent constraints, permitting user to use multiple detection techniques at the same time. They developed a prototype of this security model for android phones, and show that it is each practical and scalable: they generate no more than 2KiB/s and 64B/s of trace data for high-loads and idle operation respectively, and are able to support quite 100 replicas running on one server[25].

A.D. Schmidt and S. Albayrak presented paper on 'malicious software for smartphones'. They present a list of the most common behavior patterns and investigate possibilities how to exploit the given standard Symbian OS API for additional malware functionalities[26].

J. Cheng, S. H.Y. Wong, H. Yang and S. metal 'SmartSiren: virus detection and alert for smartphones'. They presented SmartSiren collects the communication activity informaiton from the smartphones, and performs joint analysis to discover both single-device and system-wide abnormal behaviors. They used a proxy-based design to load process load from resource-constrained smartphones and simplify the collaboration among smartphones. Once a potential virus is detected, the proxy quarantines the natural event causation targeted alerts to those directly vulnerable smartphones. They have demonstrated feasibleness of SmartSiren through implementation on Dopod 577w smartphone, and evaluated its effectiveness victimisation simulations driven by 3-week SMS traces from a national cellular carrier[27].

A.D. Schmidt, R. Bye, H.G. Schmidt, J. Clausen, O. Kiraz, K. Yuksel, A. Camtepe, and S. Albayrak, 'static analysis of executables for collaborative malware detection on android'. As Smartphones become popular for sensitive information and apps, improved malware detection mechanisms are necessary

complying with the resource constraints[28]. The contribution of this title is two fold. First, they perform static analysis on the executables to extract their operate calls in android environment using the command readelf. Method call lists are matched with malware executables for classifying them with part, Nearest Neighbor Algorithms and Prism. Second, they present a cooperative malware detection approach to improve results[28].

G. Dini, F. Martinelli, A. Saracino, and D. Sgandurra, 'MADAM: a multi-level anomaly detector for android malware'. In this paper, MADAM can monitors android at the kernel-level and user-level to notice real malware infections using machine learning techniques to differentiate between normal behaviors and malicious ones. The primary prototype of MADAM is able to notice several real malware found in the world. The device is not affected by MADAM due to the low range of false positives generated after the training phase[29].

A. Shabtai, U. Kanonov, Y. Elovici, C. Glezer, Yael Weiss, 'Andromaly: a behavioral malware detection framework for android devices'. The proposed framework realizes a Host-based Malware Detection System that continuously monitors various features and events obtained from the mobile device and apply Machine Learning anomaly detectors to classify the collected data as normal or abnormal. They developed four malicious applications and check Andromaly's ability to detect new malware based on samples of known malware. They evaluated many combinations of anomaly detection algorithms, feature choice methodologies in order to find out the combination that yields the best performance in detecting new malware on android[30].

Research Finding

Android has two basic methods of security enforcement. Firstly, applications run as Linux processes with their own user IDs and thus are separated from each other. This way, vulnerability in one application does not affect other applications. Since Android provides IPC mechanisms, which need to be secured, a second enforcement mechanism comes into play. Android implements a reference monitor to mediate access to application components based on permission. If an application tries to access another component, the end user must grant the appropriate permissions at installation time [31].

Phone identifiers are leaked through plaintext requests. Phone identifiers used as device fingerprints. Phone identifiers, specifically the IMEI, are used to track individual users. The IMEI is tied to personally identifiable information (PII). Not all phone identifier use leads to ex-filtration. Phone identifiers are sent to advertisement and analytics servers[5].

Using state-of-the-art tools for finding security bugs cannot reveal logical security problems such as undesirable interactions between components. With increasing complexity of software, software companies need to understand the security risks of their code, and tools employing program comprehension functionality will support them with this challenging task[9].

A study of android application security finding of exposure of phone identifiers and location are consistent with previous studies; analysis framework allows observing not only the existence of dangerous functionality, however conjointly how it occurs inside the context of the application. However, the integration of those technologies into an application certification process needs overcoming logistical and technical challenges[5].

Enhancing security of Linux-based android devices, Open source APIs of android may result in benign and malicious research activities hopefully resulting in an excellent safer smartphone platform[11].

L4Android: a generic operating system framework for secure smartphones: In this title they presented a generic OS framework that facilitates the creation of secure smartphone systems. The framework consists of three core components. A microkernel acts as the secure foundation and is accompanied by a user mode runtime environment. The third component is VMs to securely encapsulate existing smartphone operating system.

They implemented the core components of their framework on a mobile x86 and ARM platform. They evaluated framework by showing how it can be applied to available as the open source L4 solve four challenges in smartphone security such as secure software smartcards, and a unified corporate and private mobile phone[13].

Researches identified two fundamental causes of the attacks in WebView: weakening of the TCB and sandbox. They have shown that the condition for launching attacks is already matured, and the potential victims are in the millions. In their on-going work, they are developing solutions to secure WebView[14].

Android users need a way to determine if applications are leaking their personal information. They created a mapping between API calls and the permissions they must have to execute. AndroidLeaks is capable of analyzing 24,350 in 30 hours. AndroidLeaks drastically reduces the number of applications and the number of traces that a security auditor has to verify manually[16].

Android open source software and programmable framework behavior make it vulnerable to virus attacks[20]. The title takes into consideration the fact that Smart phones are memory, battery and speed constrained and hence exploiting the cloud to do the reputation index computation of a given application. By referring to the calculated matrix of reputation built by a given application, the model will notify users on the risk of the application before installation. Applications can be classified as highly risky, medium risk, less risk and genuine all based on

reputation they have built in the cloud. The experimental results show that some application need to be regarded as highly risky and therefore warn users not to install them until they improve their reputation by passing the threshold set by the reputation based security model[19].

An android application sandbox system for suspicious software detection: In this title they presented a sandbox created for analyzing Android applications applicable as cloud service. Unlike other sandboxes, they added a pre-check technique that can analyze Android executables in a fixed manner. This can reports usage of malicious patterns within source code. The dynamic analysis can logged system calls from application. These can be used for further detections, either performed manually or automatically[21].

User can express and enforce fine-grained policies with temporal, spatial, and cardinal conditions that refer to both single representations of data and, via taint tags, to all representations of a data item. There system helps defend against two attacker models: malicious apps and malicious users. Their system considers the information flow in intents and content provider requests, because this alone is not sufficient, additional hooks were placed to observe the information flow between apps and the file system, the network and remote services (IPC). The Security-Manager is deployed as an integral part of Android and cannot be uninstalled by the user. Authentication between the Security-Manager and the monitor is achieved by Android's IPC mechanism Binder[23].

They evaluated security and performance of their system. The security evaluation showed that the system can be considered as secure, in a sense that it is not possible for attackers to circumvent the monitor under the stated assumptions. The performance overhead was shown to be in an acceptable range for realistic end-user scenarios[24].

Android devices are complex, vulnerable, and attractive targets for attackers because of their broad application domain. The need for strong protection is apparent, preferably using multiple and diverse attack detection measures. Their security model performs attack detection on remote servers in the cloud where the execution of the software on the phone is mirrored in a virtual machine[25].

The evaluation of a user space implementation of our architecture Paranoid Android, shows that transmission overhead can be kept well below 2.5KiBps even during periods of high activity (browsing, audio playback), and to virtually nothing during idle periods. Battery life is reduced by about 30%, but they show that it can be significantly improved by implementing the tracer within the kernel. They conclude that our architecture is suitable for protection of mobile phones. Moreover, it offers more comprehensive security than possible with other models[25].

There is danger of malware for smartphones. Publicly available APIs can lead to new malwares that are able to extract various private data as well as to perform harmful action on infected devices. Private information is the number one data on mobile phones, and hence, a loss or modification will harm every affected person. But, as less and less critical malwares appear, security consideration seems to lose their importance. This is a big mistake and underestimating smartphone malware can cause serious problems not only concerning privacy issues [26].

SmartSiren: virus detection and alert for smartphones: The era of smartphone is on the horizon, and so is smartphone virus. The smartphones are particularly vulnerable to viruses due to their versatile communication capabilities, yet are difficult to harness due to their resource constraints and intermittent network connectivity. As a result, the viruses can easily spread out and cripple both the smartphone users and the cellular and telephony infrastructures[27].

SmartSiren requires limited assistance from the cellular infrastructure and poses minimal processing overhead to the smartphones. While the users can enjoy the targeted virus alert services, their privacy is also protected. The feasibility and effectiveness of SmartSiren have been confirmed by both real implementations and trace-driven simulations[27].

Static analysis of executables for collaborative malware detection on android using static ELF analysis turned out to be an efficient way to observe malware on Android using simple classifiers. These results can be improved once applying collaborative measures which can minimize the false-negative rate. Real resource consumption are major indicator whether this method can be extended to more advanced tasks, e.g. adding more semantically data to the collaborative approach or using additional advanced classifiers[28].

Smartphone can be monitored in order to transmit feature vectors to a remote server. The gathered data is intended to be used for anomaly detection methods that analyze the data for distinguishing between normal and abnormal behavior. Abnormal behavior can indicate malicious software activity. Furthermore, even unknown malware can be detected, since no signatures are used. Most of the top ten applications preferred by mobile phone users affect the monitored features in different ways. This strengthens the approach of using anomaly detection in order to detect malware on smartphone[32].

MADAM framework allows early detection of intrusion attempts and malicious actions performed by real malware on Android platform. The framework exploits a multi-level approach i.e. that combines features at the kernel-level and at the application level, and is based upon machine learning techniques. The first prototype of MADAM for Android smartphone has managed to detect all the 10 monitored real malware, with an impact on the user experience due to the few false positives issued per day. To the best of our knowledge,

these results are a noticeable improvement to solutions presented in previous work, both for detection rate of real malware on current Android-based smartphones, and occurrences of false positives[29].

Conclusion

Now days more than 1 million Android device activated [33]. Android has very few restrictions for developer, increases the security risk for end users. In this paper we have reviewed security issues in the Android based Smartphone. The integration of technologies into an application certification process requires overcoming logistical and technical challenges. Android provides more security than other mobile phone platforms. Kirin will help mold Android into the secure operating system needed for next-generation computing platforms.

References

1. Android Open Source Project. Android Security Overview. http://source.android.com/devices/tech/security/index.html. **(2013)**

2. Kaur S. and Kaur M., Review Paper on Implementing Security on Android Application, *Journal of Environmental Sciences, Computer Science and Engineering & Technology*, **2(3)**, **(2013)**

3. Android Open Source Project. Security and permissions. http://developer.android.com/guide/topics/security/permissions.html. **(2013)**

4. Android Open Source Project. Publishing on GooglePlay. http://developer.android.com/distribute/googleplay/publish/preparing.html. **(2013)**

5. Enck W., Octeau D., McDaniel P. and Chaudhuri S., A Study of Android Application Security, *The 20th USENIX conference on Security*, 21-21, **(2011)**

6. Powar S., Meshram B. B., Survey on Android Security Framework, *International Journal of Engineering Research and Applications*, **3(2)**, (**2013**)

7. Smalley S. and Craig R., Security Enhanced (SE) Android: Bringing Flexible MAC to Android, www.internetsociety.org/sites/default/files/02_4.pdf. **(2012)**

8. Enck W., Gilbert P., Chun B.G., Cox L.P., Jung J., McDaniel P. and Sheth A.N., TaintDroid: An Information-Flow Tracking System for Realtime Privacy Monitoring on Smartphones, *9th USENIX Symposium on Operating Systems Design and Implementation*. **(2010)**

9. Berger B.J., Bunke M., and Sohr K., An Android Security Case Study with Bauhaus, *Working Conference on Reverse Engineering*, 179–183 **(2011)**

10. Ongtang M., McLaughlin S., Enck W. and McDaniel P., Semantically Rich Application-Centric Security in Android, *Computer Security Applications Conference*, 340–349 **(2009)**

11. Schmidt A.D., Schmidt H.G., Clausen J., Camtepe A., Albayrak S. and Yuksel K. Ali and Kiraz O., Enhancing Security of Linux-based Android Devices, http://www.dai-labor.de/fileadmin/files/publications/lk2008-android_security.pdf **(2008)**

12. Marforio C., Francillon A. and Capkun S., Application Collusion Attack on the Permission-Based Security Model and its Implications for Modern Smartphone Systems, ftp://ftp.inf.ethz.ch/doc/tech-reports/7xx/724.pdf **(2013)**

13. Lackorzynski A., Lange M., Warg A., Liebergeld S., Peter M., L4Android: A Generic Operating System Framework for Secure Smartphones, *18th ACM Conference on Computer and Communications Security*, 39-50 **(2011)**

14. Luo T., Hao H., Du W., Wang Y. and Yin H., Attacks on WebView in the Android System, *27th Annual Computer Security Applications Conference*, 343-352 **(2011)**

15. Barrera D., Güne H., Kayacık S., Oorschot P.C. van and Somayaji A., A Methodology for Empirical Analysis of Permission-Based Security Models and its Application to Android, *17th ACM conference on Computer and communications security*, 73–84 **(2010)**

16. Gibler C., Crussell J., Erickson J. and Chen H., Android Leaks: Automatically Detecting Potential Privacy Leaks In Android Applications on a Large Scale, *5th international conference on Trust and Trustworthy Computing*, 291-307 **(2012)**

17. Burguera I., Zurutuza U. and Tehrani S.N., Crowdroid: behaviour-based malware detection system for Android, *1st ACM workshop on Security and privacy in smartphones and mobile devices*, 15-26 **(2011)**

18. Polla M.L., Martinelli F., and Sgandurra D., A Survey on Security for Mobile Devices, *Communications Surveys & Tutorials, IEEE*, **15(1)**, 446–471 **(2013)**

19. Tesfay W.B., Booth T., and Andersson K., Reputation Based Security Model for Android Applications, Trust, Security and Privacy in Computing and Communications, *IEEE Computer Society*, 896-901 **(2012)**

20. Bing H., Analysis and Research of Systems Security Based on Android, *Intelligent Computation Technology and Automation*, 581–584 **(2012)**

21. Bläsing T., Batyuk L., Schmidt A.D., Camtepe S.A. and Albayrak S., An Android Application Sandbox system for suspicious software detection, *Malicious and Unwanted Software*, 55-62 **(2010)**

22. Vidas T., Votipka D. and Christin N., All Your Droid Are Belong To Us: A Survey of Current Android Attacks, *The 5th USENIX conference on Offensive technologies*, 10-10 **(2011)**

23. Holla S. and Katti M.M., Android based mobile application development and its Security, *International Journal of Computer Trends and Technology*, **3(3)**, 486-490 **(2012)**

24. Feth D., Pretschner A., Flexible Data-Driven Security for Android, *The 2012 IEEE Sixth International Conference on Software Security and Reliability*, 41-50 **(2012)**

25. Portokalidis G., Homburg P., Anagnostakis K. and Bos H., Paranoid android: versatile protection for smartphones, *Computer Security Applications Conference*, 347–356 **(2010)**

26. Schmidt A.D. and Albayrak S., Malicious software for smartphones, https://www.dai-labor.de/fileadmin/files/publications/smartphone_malware.pdf **(2008)**

27. Cheng J., Wong H.Y., Yang H. and Lu S., Smartsiren: virus detection and alert for smartphones, Mobile Systems, Applications, and Services, 258–271 **(2007)**

28. Schmidt A.D., Bye R., Schmidt H.G., Clausen J., Kiraz O. and Yuksel K., A. Camtepe, and S. Albayrak, Static analysis of executables for collaborative malware detection on android, *2009 IEEE International Confernece on Communications*, 1-5 **(2009)**

29. Dini G., Martinelli F., Saracino A. and Sgandurra D., MADAM: a multi-level anomaly detector for android malware, http://www.iet.unipi.it/g.dini/research/papers/2012-MMM-ANCS.pdf **(2012)**

30. Shabtai A., Kanonov U., Elovici Y., Glezer C. and Y. Weiss, Andromaly: a behavioural malware detection framework for android devices, *Journal of Intelligent Information Systems*, **38(1)**, 161-190 **(2012)**

31. Enck W., Ongtang M., and McDaniel P., Understanding Android security, IEEE Security Privacy, **7 (2009)**

32. Schmidt A.D., Peters F., Lamour F. and Albayrak S., Monitoring Smartphones for Anomaly Detection, *1st international conference on MOBILe Wireless MiddleWARE, Operating Systems, and Applications*, Article No. 40 **(2008)**

33. Android Open Source Project. What is Android? http://developer.android.com/about/index.html **(2013)**