ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 3, March 2015

MOSES: Supporting and Enforcing Security Profiles on Smart Phones

E.Ramya, A.Nirosha

PG Scholar, Department of Computer Science, Dhanalakshmi Srinivasan College of Arts & Science for Women, Perambalur, Tamilnadu, India

Assistant Professor, Department of Computer Science, Dhanalakshmi Srinivasan College of Arts & Science for Women, Perambalur, Tamilnadu, India

ABSTRACT: Smart phones are very effective tools for increasing the productivity of business users. With their increasing computational power and storage capacity, smart phones allow end users to perform several tasks and be always updated while on the move. Companies are willing to support employee-owned smart phones because of the increase in productivity of their employees. However, security concerns about data sharing, leakage and loss have hindered the adoption of smart phones for corporate use. Present MOSES, a policy-based framework for enforcing software isolation of applications and data on the Android platform. In MOSES, it is possible to define distinct Security Profiles within a single smart phone. Each security profile is associated with a set of policies that control the access to applications and data. Profiles are not predefined or hardcoded, they can be specified and applied at any time. One of the main characteristics of MOSES is the dynamic switching from one security profile to another. To run a thorough set of experiments using our full implementation of MOSES. The results of the experiments confirm the feasibility of our proposal

KEYWORDS*:* Moses, Smart phone, Security Profile Manager, Moses Policy Manager

I. INTRODUCTION

Despite this positive scenario, since users can install third-party applications on their Smartphone, several security concerns may arise. For instance, malicious applications may access emails, SMS and MMS stored in the Smartphone containing company confidential data. Even more worrying is the number of legitimate applications harvesting and leaking data that are not strictly necessary for the functions the applications advertise to users. One possible solution to this problem is isolation, by keeping applications and data related to work separated from recreational applications and private/personal data. Within the same device, separate security environments might exist: one security environment could be only restrict to sensitive/corporate data and trusted applications; a second security environment could be used for entertainment where third-party games and popular applications could be installed. As long as applications from the second environment are not able to access data of the first environment the risk of leakage of sensitive information can be greatly reduced. Such a solution could be implemented by means of virtualization technologies where different instances of an OS can run separately on the same device. Although virtualization is quite effective when deployed in full-fledged devices (PC and servers), it is still too resource demanding for embedded systems such as smart phones. Another approach that is less resource demanding is virtualization. Unlikely full virtualization where the guest OS is not aware of running in a virtualised environment, in virtualization it is necessary to modify the guest OS to boost performance.

II. LITERATURE SURVEY

2.1. *Survey android leaks: Automatically Detecting Potential Privacy Leaks in Android Applications on a Large Scale.*

Android provides the core smart-phone experience, but much of a user's productivity depends on third-party applications. To this end, Android has numerous marketplaces where users can download third-party applications. In contrast to the market policy for OS, in which every application is reviewed before it can be published, most Android

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 3, March 2015

markets allow developers to post their applications with no review pro- cess. This policy has been criticized for its potential vulnerability to malicious applications. Google instead allows the Android

Market to self-regulate, with higher-rated applications more likely to show up in search results and reported malicious applications removed. Android sandboxes each application from the rest of the system's resources in an ort to protect the user . This attempts to ensure that one application cannot tamper with another application or the system as a whole. If an application needs to access a restricted resource, the developer must statically request permission to use that resource by declaring it in the application's manifest _le. When a user attempts to install the application, Android will warn the user that the application requires certain restricted resources (for instance, location data), and that by installing the application, she is granting permission for the application to use the specied resources. If the user declines to authorize these permissions, the application will not be installed .However, statically requiring permissions does not inform the user how the resource will be used once granted. A maps application, for example, will require access to the Internet in order to download updated map tiles, route information and trac reports. It will also require access to the phone's location in order to adjust the displayed map and give real-time directions. The application's functionality requires sending location data to the maps server, which is expected and acceptable given the purpose of the application. However, if the application is ad-supported it may also leak location data to advertisers for targeted ads, which may compromise a user's privacy. Given the only information currently presented to users is a list of required permissions, a user will not be able to tell how the maps application is handling her location information. To address this issue, we present Android Leaks, a static analysis framework designed to identify potential leaks of personal information in Android applications on a large scale. Leveraging WALA , a program analysis framework for Java source and byte code, we create a call graph of an application's code and then perform a reachability analysis to determine if sensitive information may be sent over the network. If there is a potential path, we use data analysis to determine if private data reaches a network sink. We have created a set of mappings between Android API methods and the permissions they require to execute using static techniques. We use a subset of this mapping as the sources and sinks of private data for our data analysis. To present Android Leaks, a static analysis framework forming potential leaks of private information in Android applications. We evaluated Android Leaks on 24,350 Android applications, Forming potential privacy leaks involving uniquely identifying phone information, location data, Wi-Fi data, and audio recorded with the microphone. Android Leaks Identies APKs and provides a set of leaks most likely to be of interest to a security researcher. We designed and implemented taint-aware slicing and an approach for identifying taint sources in call backs, which is used extensively in Android applications.

2.1.1. DISADVANTAGE

- Static analysis framework
- Depends on third-party applications
- Location data, Wi- Fi data, and audio recorded with the microphone.

2.2. Taint Droid: An Information-Flow Tracking System For Real Time Privacy Monitoring On Smart Phone.

Resolving the tension between the fun and utility of running third-party mobile applications and the privacy risks they pose is a critical challenge for smart phone platforms. Mobile-phone operating systems currently provide only coarse-grained controls for regulating whether an application can access private information, but provide little insight into how private information is actually used. For example, if a user allows an application to access her location information, she has no way of knowing if the application will send her location to a location-based service, to advertisers, to the application developer, or to any other entity. As a result, users must blindly trust that applications will properly handle their private data. This paper describes Taint Droid, an extension to the Android mobile-phone platform that tracks the flow of privacy sensitive data through third-party applications. Taint Droid assumes that downloaded, third-party applications are not trusted, and monitors–in real time–how these applications access and manipulate users' personal data. Our primary goals are to detect when sensitive data leaves the system via un trusted applications and to facilitate analysis of applications by phone users or external security services. Analysis of applications' behavior requires sufficient contextual information about what data leaves a device and where it is sent. Thus, Taint Droid automatically labels (taints) data from privacy-sensitive sources and transitively applies labels as sensitive data propagates through program variables, files, and inter process messages. When tainted data are transmitted over the network, or otherwise leave the system, Taint Droid logs the data's labels, the application responsible for transmitting the data, and the data's destination. Such real time feedback gives users and security services greater insight into what mobile applications are

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 3, March 2015

doing, and can potentially identify misbehaving applications. We evaluated the accuracy of Taint Droid using 30 randomly selected, popular Android applications that use location, camera, or microphone data. Taint Droid correctly flagged 105 instances in which these applications transmitted tainted data; of the 105, we determined that 37 were clearly legitimate. Taint Droid also revealed that 15 of the 30 applications reported users' locations to remote advertising servers. Seven applications collected the device ID and, in some cases, the phone number and the SIM card serial number. In all, two-thirds of the applications in our study used sensitive data suspiciously. Our findings demonstrate that Taint Droid can help expose potential miss behavior by third-party applications.

2.2.1 DISADVANTAGE

- Fail to provide users
- Visibility into how third-party applications

2.3. Performance Evaluation Of Para-Virtualization On Modern Mobile Phone Plate Form

Currently in order to address the security challenges in mobile phones, several different methods are used, such as implementing complete open devices (including open OS),separating application domain and cellular domain, using strict API level certification policies and restricting run-time environm(e.g. Java). However, all these solutions have their own limitations and restrictions. To overcome these Virtualization technology can date back to IBM's VM/370system in 1960's, which is the first commercial virtual machine system on the world. The initial motivation to use a virtual machine system is to support multiple operating systems and multiplex expensive mainframe hardware. A virtual machine (VM) is a duplicate of a real computer system, whose resources are fully controlled by a virtual machine monitor (VMM). The VMM provides users with an efficient, isolated processing environment, which is essential to allow more than one operating system running on one single machine. At present, virtualization technology is primarily applied on servers and workstations to help system administrators reduce management overhead. In the future, virtualization will be a solution for security and software reliability The requirements for virtualization on mobile phones are quite different from virtualization on high performance systems. Some suitable virtualization technologies for high performance systems may become un applicable on mobile phones due to hardware resources and power consumption limitations. For example, in full virtualization by DBT, the executed instructions are intercepted and replaced in real time. This is computationally intensive and unsuitable for mobile systems. And the hard wares dedicated for virtualization, which are usually available for PC and server systems are not yet available at the embedded market. Para-virtualization is currently the emerging solution for virtualization in mobile phones. Examples are Tango (current VMW are MVP)[9], Virtual Login virtualization technology and L4 microkernel virtualization technology. This is substantially determined by the fact that on mobile phones, high performance efficiency is preferred because of limited resources. Among all these available embedded system virtualization solutions, only the L4 microkernel virtualization approach is open-source, which is a big advantage in research work. It gives us the possibility to deeply understand the med system virtualization approach and allows us to do detailed analysis and evaluation. Furthermore it provides us more space for optimization in the future.L4 microkernel was designed and optimized for 486 and Pentium architectures. It has been proved quite efficient on x86 systems. However, on ARM processors, which are usually used in mobile phones, the efficiency of L4 microkernel, especially used as a VMM, has not been extensively investigated yet. In order to narrow this gap and to get exact performance data of L4 microkernel based VMM on mobile phone, we evaluated the performance of L4 Fiasco microkernel (Re-implementation of L4 microkernel from TU Dresden. In the rest of this paper, we call it L4 for short) as a VMM on a modern mobile phone platform by comparing the performance of L4Linux and native Linux.

2.3.1DISADVANTAGE

- Virtualization overhead
- Running on one single machine
- Higher development cost

III. PROBLEM DESCRIPTION

3.1 EXISITING SYSTEM

Companies are willing to support employee-owned smart phones because of the increase in productivity of their employees. Despite this positive scenario, since users can install third-party applications on their smart phones, several security concerns may arise..For instance, malicious applications may access emails, SMS and MMS stored in the smart phone containing company confidential data. Moreover, a permission cannot be revoked at run time. As long as

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 3, March 2015

applications from the second environment are not able to access data of the First environment the risk of leakage of sensitive information can be greatly reduced.

3.1.1 DISADVANTAGE

- Security concerns a but data sharing,
- Data leakage of our smart phone
- Data loss of out smart phone
- Malicious applications may access emails
- Malicious application SMS and MMS stored in the smart phone containing company confidential data

3.2. PROPOSED SYSTEM

Security environment could be only restricted to sensitive/corporate data and trusted applications Each security profile is associated with a set of policies that control the access to applications and data. A second security environment could be used for entertainment where third-party games and popular applications could be installed. As long as applications from the second environment are not able to access data of the first environment the risk of leakage of sensitive information can be greatly reduced. MOSES implements soft virtualization through controlled software isolation.Each security profile (SP) can be associated to one or more contexts that determine when the profile become active. Both contexts and profiles can be easily and dynamically specified by end users. MOSES provides a GUI for this purpose. Switching between security profiles can require user interaction or be automatic, efficient, and transparent to the user. Such a solution could be implemented by means of virtualization technologies where different instances of an OS can run separately on the same device.

3.2.1 ADVANTAGE

- Applications and private/personal data. Data and trusted applications.
- Each security profile set application and data.
- Leakage of sensitive information can be greatly reduced.
- Different OS can run separately on the same
- Third person is not access data sharing device
- No leakage, loss.
- Application and data full security

IV. SYSTEM ARCHITECTURE

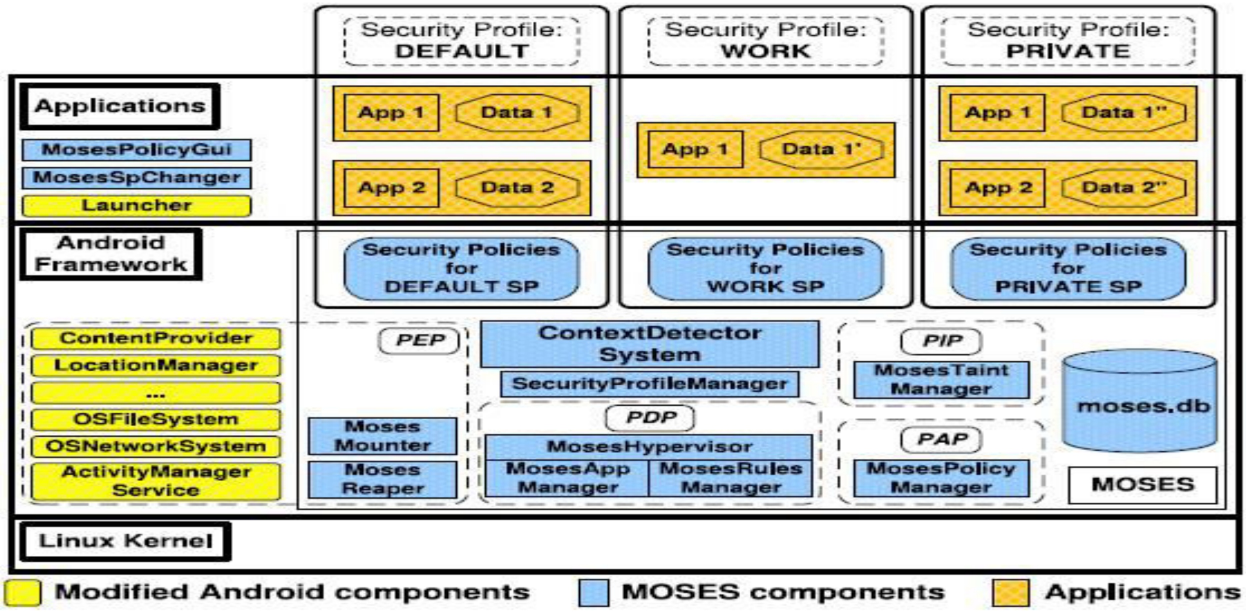

Figure No.4.1.System Architecture

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 3, March 2015

V. MODULES DESCRIPTION

5. 1.USER REGISTRATION AND AUTHENTICATION

Authentication is the process of verifying the credentials such as username and password of the user and then allows that user to access an application or server. The proper identification of a person, device, or group is vital for safeguarding and maintaining the confidentiality, integrity, and availability of the application. Access controls can be created for authenticated users and information.

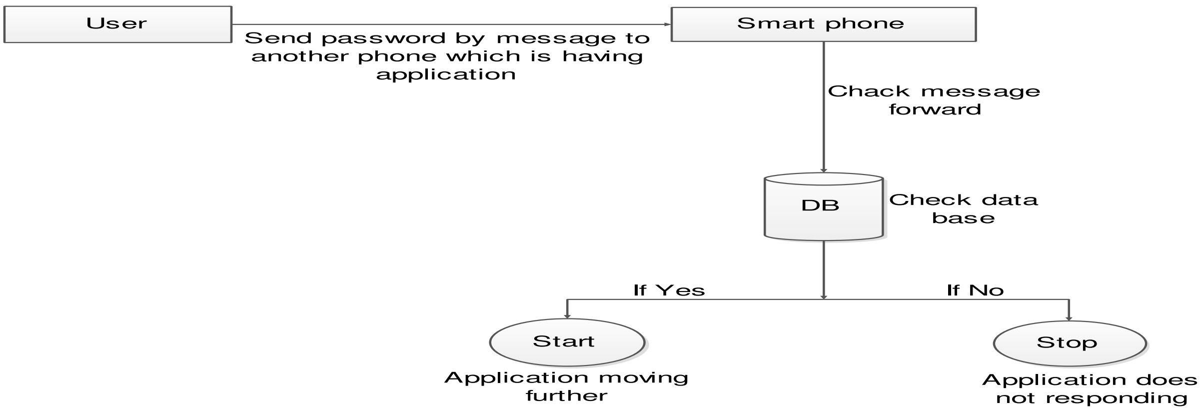

Figure No.5.1.1 User Registration And Authentication

5.2. SEC URITY PROFILE CREATE

New Security Profile Create in Security profile name Security Profile location, Security Profile time and Security Profile Priority. Present MOSES, a policy-based framework for enforcing software isolation of applications and data on the Android platform. In
MOSES, it is possible to define distinct Security Profiles within a single smart phone.

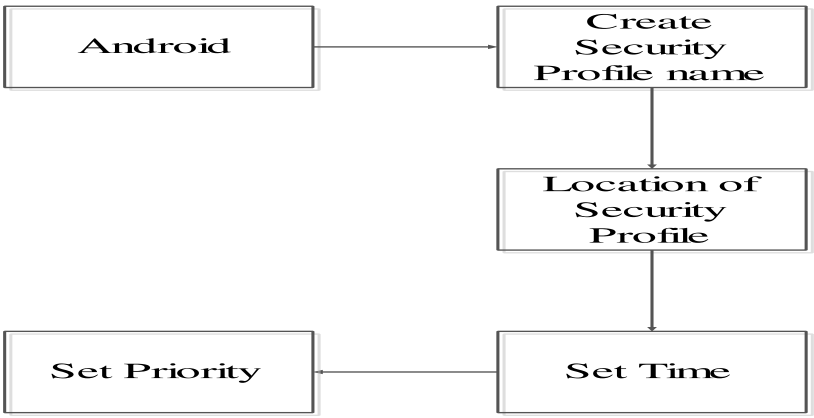

Figure: 5.2.1 Security profile create

5.3. MOSES SECURITY PROFILE MANAGER

The Security Profile Manager holds the information linking SP with one or more Context. The Security Profile Manager is responsible for the activation and deactivation of SPs. If a newly activated Context corresponds to the active SP then the notification is ignored. If the SP corresponding to a newly active Context has a lower or equal priority to the currently running SP, then the notification is ignored;

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 3, March 2015

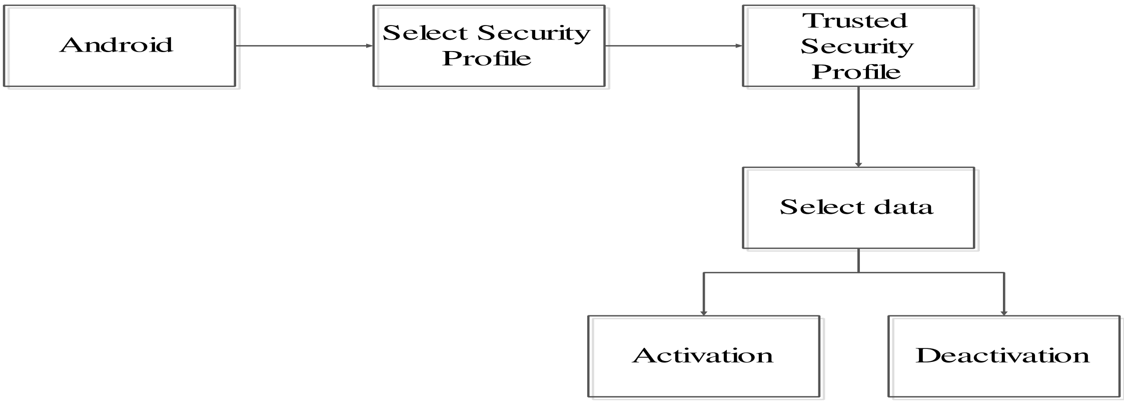

Figure: 5.3.1.Moses Security Profile Manager

5.4. CHECK ERROR PROCEESS

If these applications are running during the profile switch, then we need to stop their processes. The Moses Reaper is the component responsible for shutting down processes of applications no longer allowed in the new

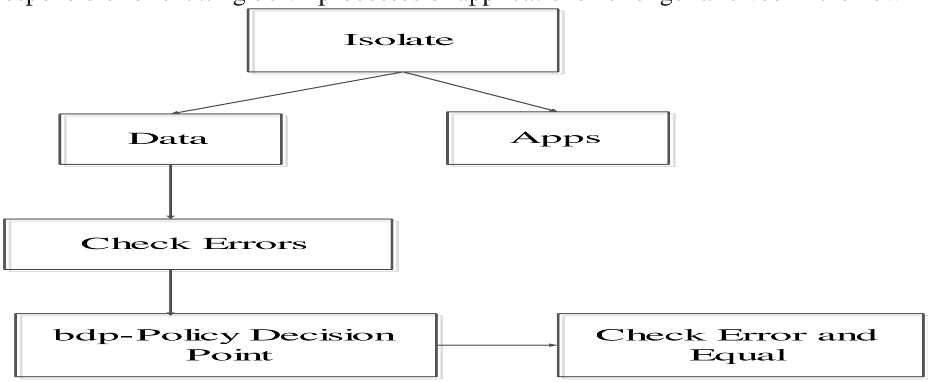

Figure: 5.4.1.Check Error Process

5.5. MOSES MANAGER PROCESS

The Moses Hypervisor delegates the policy checks to its two managers: the Moses App Manager and the Moses Rules Manager. The former is responsible for deciding which apps are allowed to be executed within a SP. The latter takes care of managing Special Rules. The Moses Policy Manager acts as the policy administrator point (PAP) in MOSES. It provides the API for creating, updating and deleting MOSES policies. It also allows a user to define, modify, remove monitored Contexts and assign the
SPs

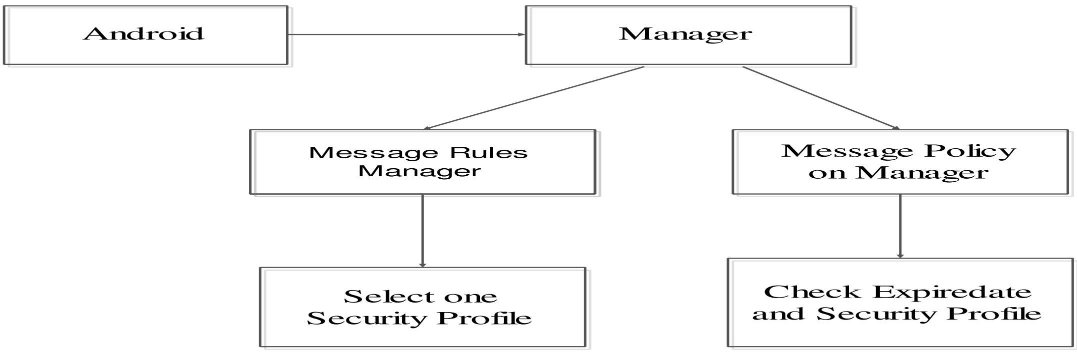

Figure: 5.5.1.Moses Manager Process

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 3, March 2015

5.6. MOSES APPLICATION PROCESS

To allow the user of the device to interact with MOSES, we provide two MOSES applications: the Moses Sp Changer and the Moses Policy Guy. The Moses Sp Changer allows the user to manually activate a SP. It communicates with the Moses Hypervisor and sends it a signal to switch to the profile required by the user. The Moses Policy

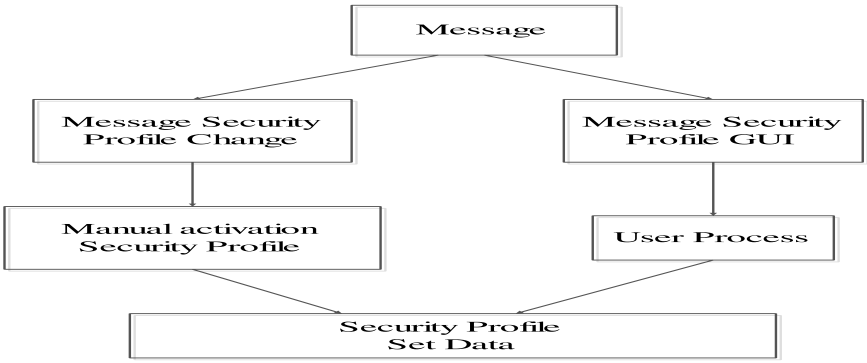

Figure: 5.6.1.Moses Application process

VI.CONCLUSION

MOSES is the first solution to provide policy-based security containers implemented completely via software. By acting at the system level we prevent applications to be able to bypass our isolation. However, at the present moment MOSES has also some limitations. At first, fine-grained policies and allowed applications are specified using the UID of an application. Meanwhile, in Android it is possible that some applications share the same UID. Thus, if we apply MOSES rules and restrictions to one application they automatically will be extended to the other ones with same UID. Furthermore, some fine-grained policies in MOSES are built on top of Taint droid [4] functionality. Thus, MOSES inherits the limitations of Taint droid explained in Section 3. It should be also mentioned that the applications that have root access to the system can bypass MOSES protection

VII.FUTURE ENHANCEMENT

MOSES can also be improved in several aspects. For instance, to make the policy specification process easier, a solution could be to embed into the system policy templates that can be simply selected and associated to an application. It should be also mentioned that currently MOSES does not separate system data (e.g., system configuration files) and information on SD cards. In the future we plan to add this functionality to the system. Moreover, performance overheads are also planned to be reduced considerably in the future versions.

REFERENCES

1. Gartner Says Smartphone Sales Accounted for 55 Percent of Overall Mobile Phone Sales in Third Quarter of 2013, http://www. gartner.com/newsroom/id/2623415, 2014.
2. Are Your Sales Reps Missing Important Sales Opportunities? http://m.sybase.com/files/White_Papers/Solutions_SAP_Reps.
3. pdf, 2014.
4. Unisys Establishes a Bring Your Own Devic (BYOD) Policy, http://www.insecureaboutsecurity.com/2011/03/14 unisys
5. Establishes_a_bring_your_own_device_byod_policy/,2014
6. W. En ck, P. Gilbert, B.-G. Chun, L.P. Cox, J.Jung, P. McDaniel, and A.N. Sheth, “Taint droid: An Information-Flow Tracking
7. System for Real time Privacy Monitoring on Smart phones,” Proc. Ninth USENIX Conf. Operating Systems Design an
8. Implementation (OSDI ’10), pp. 1-6, 2010.
9. C. Gibler, J. Crus sell, J. Erickson, and H. Chen, “Android Leaks: Automatically Detecting Potential Privacy Leaks in Android
10. Applications on a Large Scale,” Proc. Fifth Int’l Conf. Trust and Trustworthy Computing (TRUST ’12), pp. 291-307, 2012.
11. Y. Xu, F. Bruns, E. Gonzalez, S. Traboulsi, K. Mott, and A. Bilgic, “Performance Evaluation of Para-Virtualization on Modern
12. Mobile Phone Platform,” Proc. Int’l Conf. Computer, Electrical, and Systems Science and Eng. (ICCESSE ’10), 2010.
13. M. Lange, S. Liebergeld, A. Lackorzynski, A. Warg, and M. Peter, “L4Android: A Generic Operating System Framework for Secure
14. Smart phones,” Proc. First ACM Workshop Security and Privacy in Smart phones and Mobile Devices (SPSM ’11), pp. 39-50, 2011.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 3, March 2015

BIOGRAPHY

Ms. E.RAMYA is a PG Scholar M.Sc., CS in Department of Computer Science Dhanalakshmi Srinivasan College Of Arts & Science For Women Perambalu,Tamilnadu. Received her B.Sc., Computer Science in Department of Computer Science Dhanalakshmi Srinivasan College Of Arts & Science For Women Perambalu, Tamilnadu. Affiliated by Bharathidasan University Trichy .Her area of interest is Mobile Computing, Networking.

Ms.A.NIROSHA, Received M.C.A, M.E Degree in Computer Science & Engineering. Currently working as Assistant Professor in Department of Computer Science, Dhanalakshmi Srinivasan College of Arts & Science for Women, Perambalur, Tamilnadu, India. 3 Papers are Published in International Journal and Published a Book named “A Small Pick up From Computer Concepts” and 2 Papers are presented in International Conference, 1 Paper is presented in State Level Seminar. Her Research areas are Networking, Web Technology, and Mobile Computing.