Scientific Journal Impact Factor (SJIF) : 1.711
International Journal of Modern Trends in Engineering and Research
ISSN (ONLINE) : 2349 - 9745
ISSN (PRINT) : 2393 - 8161

AN EFFECTIVE FRAMEWORK FOR PROVIDING SECURITY ON SMARTPHONES

M.manasa[1] , N.lakshmi chaitanya [2]
[1]*Department of CSE, Audisankara College of Engineering and technology, Gudur,manasa*
[2] *Department of CSE, Audisankara College of Engineering and technology, Gudur,*

Abstract-Now-a-days Smartphone is turning into one in all the essential features in the form of many exclusive options that it's providing within the kind of varied application storage, computational power. As a result of this several company corporations provides mobile versions of their desktop applications to their workers so as to be updated forever. As Smartphone providing such options to end-user there are some security issues that require to be handled. This manuscript deals with the design and implementation of the SecurePhoneApp application that run on the Smartphone in association with MOSES application. Here, Moses provides internal security to the information on the Smartphone whereas SecurePhoneApp prevents the information from external threat.

Keywords: isolation, security profile, virtualization, context, Smartphone

I. INTRODUCTION

In today's era the usage of Smartphone is increasing day by day. Within the Smartphone domain android OS is gaining more popularity because of its openness to the third-party developer [1] [2]. Smartphones enable user to perform many task whereas being on the move and remains updated by that time. As a result, several company corporations are willing to support the usage of Smartphone's to increase the productivity of business users. Despite of this positive situation, many security issues might arise regarding data present on the device. As several firms support the utilization of Smartphone's, it's very essential to produce security. I may happen that the third-party application .Unknowingly tampers the user information. One possible resolution to this problem is isolation, by keeping application and information involving work separated from recreational applications and private/personal information within the same device, separate security surroundings would possibly exist: one security surroundings may well be solely restricted to

Sensitive/corporate data and trustworthy applications; a second security environment could be used for
entertainment where third-party games and widespread applications could be installed. As long as applications from the second surroundings are not able to access the information of the first surrounding, the risk of leakage of sensitive data is greatly reduced. This can be done by means of virtualization technique [3]. This manuscript deals the implementation of two applications that runs on android Smartphone and permits the user to avoid loss of their data internally and externally. Two applications primilarly are MOSESApp and SecurePhoneApp. We have slightly modified the implementation of MOSES from its original work because of some limitations that are given within the following sections. We have to done several experiments to know the compatibility of both the applications that is mentioned within the experimental analysis section.

II. IMPLEMENTATION OF PROPOSED WORK

Throughout the worldwide the sales of Smartphone's is up to 455 million and there is a 46 percent of

increase from the year of 2012 to 2013. There is 77 percent increase in Smartphone's and 11 percent increase in mobile phones. As the number of users increased to use the Smartphone's and as the average selling price of the Smartphone's is almost relatively similar to the old featured phones so there was gradual decrease in the use of old featured phones. In the Smartphone's world the most used OS is an Android OS where as the Android OS has an 82 percent share in market [1]. The above figures shows the easily acceptance of Android because of its easy to used by third party developers. By observing the MOSES architecture we understood that it provides internal security to the data present in the Smartphone but problem arises when we talk about the theft concept. Suppose the person is working in some software company and using the MOSES architecture on his/her Smartphone at that time someone has stolen his Smartphone, at that time he lost confidential data. To solve this problem certain solutions are there so that the person gets back his Smartphone and also data presented in the phone. The solution is shown in the form of following diagram.

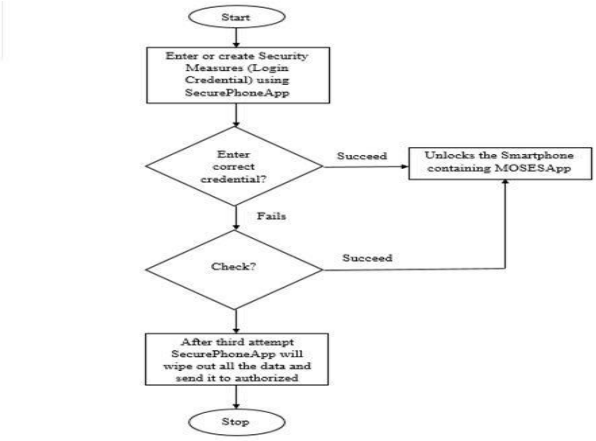

Fig. 1 Flowchart of Proposed Work

III. MOSES FRAMEWORK

The MosesTaintManager component manages the "shadow database" which stores the taint values used by Taintroid [5]. In MOSES, we can taint specific rows of a content provider: to be able to perform per row filtering when an app access data in the content provider. For instance, it is possible to filter out from the query result data the rows which contain the information about device identifiers or user contacts. Given the fact that the enforcement of policies depends on the information provided by the MosesTaintManager, this component acts as a policy information point (PIP).

Moses provides abstraction so as to separate the apps and data related to different contexts installed on the single device. For example the data and apps related to work separated from the personal data. In short MOSES provides separate compartments for apps and data. Here the compartments are referred to as environment or security profiles (SP). In general sp's are set of policies or rules that control which apps to be executed and what data need to used.

A. Architecture

Main part of MOSES is development of context. The Context Detector System is liable for activating/deactivating of context. Once it happens the element ContextDetectorSystem notifies regarding this to the SecurityProfileManager handle this information linking a sp with the context.

The element SecurityProfileManager is employed for the activation and deactivation of SecurityProfiles. The SecurityProfileManager works as follow:

The Moses Hypervisor is the element that acts as the Policy decision point (PDP) in Moses. MosesHypervisor provides a central purpose for MOSES security checks against the policies outlined for the active SP to control access to resources. The MosesHypervisor delegates the policy checks to its two managers: the MosesAppManager and also the MosesRulesManager. The previous is liable for choosing which apps are allowed to be performed within a SP. The latter takes care of managing special rules [4].

The MosesPolicyManager operate as the policy administrator purpose (PAP) in MOSES. It provides the API for creating, updating, deleting MOSES policies. It also permits the user to outline, modify, remove the monitored contexts and assign them to SPs. Moreover this element conjointly controls access to MOSES policy database (moses.db) permitting only apps with special permissions to act with this element.

The MosesTaintManager element manages the "shadow database" that stores the taint values utilized by tantroid [5]. In MOSES we able to taint specific rows of a content supplier:to be able to perform per row filtering once an app access data in the content provider. It is possible to filter out from the query result data the rows contain the knowledge regarding the device identifiers or user contacts. Given the actual fact that the social control of policies depends on the knowledge provided by the MosesTaint Manager, this component acts as a policy information purpose (PIP).

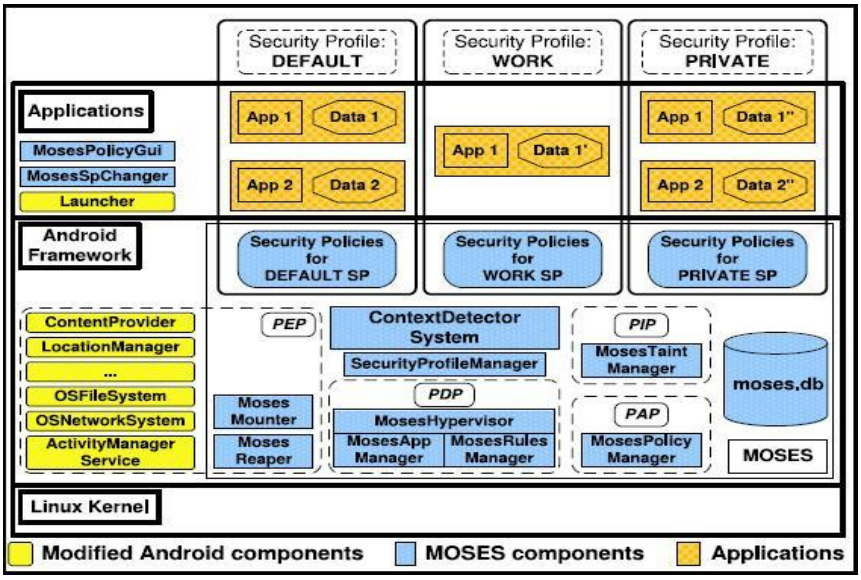

FIG. 2 Moses Architecture

The decision taken by the Moses Hypervisor ought to be employed by the policy social control purpose (PEP). MOSES affects many components among android middleware wherever choices ought to be implemented. For this reason, the life includes several android components giving system services like LocationManager and ActivityManagerService. More over some android core categories (such as the OSFileSystem and OSNetworkSystem) are changed to enforce choices concerning the access to the file system and network, respectively.

The social control of separated SPs needs special components to manage application processes and file system views. Once a new SP is activated, it would deny the execution of some applications allowed within the previous profile. If these applications are running throughout the profile switch, then we want to prevent their processes. The MosesReaper is the component liable for closing down processes of applications now not allowed within the new SP after the switch. In MOSES,

applications have access to different data depending the active profile. To separate knowledge between profiles totally different file system views are supported. This practicality is provided by the MosesMounter. To permit the user of the device interact with the MOSES, we offer two MOSES applications: the MosesSpChanger and therefore the MosesPolicyGui. The MosesSpChanger permits the user to manually activate a sp. It communicates with the MosesHypervisor and sends it a signal to modify to the profile required by the user. The Moses Policy Gui permits the user to manage SPs [4].

B. Implementation

We have slightly modified the existing work of MOSES as a result of the services needed for it is very expensive. In base work author has created context definition on that the activation or deactivation of security profiles depends. Every security profile OS related to one or more definition of context. A context definition may be a Boolean expression outlined over any data that may be obtained from the smart phones raw sensors (e.g./ GPS sensor) and logical sensors. Logical sensors are functions that combine raw data from physical sensors to capture specific user behaviors (such as detection whether or not the user is running). Once a context definition evaluates to true, the SP related to such a context is activated. It is possible state of affairs once several contexts, that area unit related to totally different SPs, could also be active at same time. To resolve such conflicts, every SP is additionally appointed with apriority permitting MOSES to activate the SP with the very best priority. If SPs have same priority the SPs that has been activated first, can stay active. This behavior needed Google Maps service that is cheap. Therefore what we have done is we are activating the profile based on the password that is given whereas making security profile. If this evaluates to true it will activate the actual profile in keeping with it apps are executed and its information will be accessed.

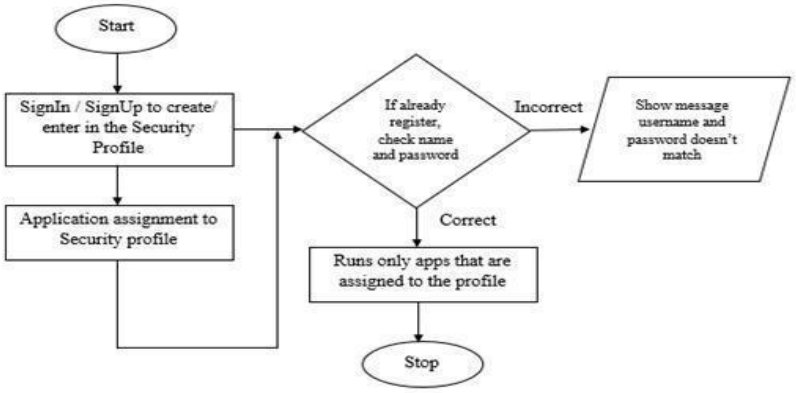

FIG.3 Flowchart for MOSESApp

C. Performance Analysis of MOSESApp

Because of some limitations that are mentioned in previous section, we have a tendency to do not seem to be implementing Moses in terms of context definition. Therefore what we have got done, we have control the activation and deactivation of profiles in terms of passwords. If login name i.e. context name and passwords matches then solely explicit profile gets activated else it will pop-up the toast like user name and password does not match. Let's have a detail look on the app. The first step is to form context definitions which provide security environments or profiles to applications so as to try their executions in safe mode. Here context definition is outlined in terms of username and password which cans ne also termed as "Login Credentials for Moses security environments". Security profiles or environments are the set of protocols to rules that regulates the app behaviors. Following steps offers detail description regarding Moses working.

1) As presently as application launches, it shows two buttons i.e. sign-in and sign-up.
2) User has to create context definitions by clicking on signup button assigns varied applications thereto so as to provide them in safe environment for execution. Here we tend to distribute all apps to HOME profile for our convenience. User will assign anyone among these
3) Again we have created another definition referred to as WORK to that none of those applications are allotted.
4) Once it is created, user will sign-in with any one of this. If user sign-in with HOME then all the applications allotted with it will be executed. On other if sign-in work then none of the applications will get executed.
5) With the help of sign-in button user will manually switched between the profiles at any time. If user fails to enter correct username and password then it will show the message username and password does not match.

D. *MOSES* Evaluation

To measure the energy overhead created by MOSES and SecurePhoneApp, we have a tendency to performed the subsequent tests. We have a tendency to run seven third-party applications(sequentially) via a monkey runner script: Caller, Bluetooth, Email, Message, Media-player, Video-camera, and Camera. For each of them the script performed common operations representative for the applications (making calls, permitting and denying Bluetooth, sending-emails and SMS, enjoying music, capturing video and pictures).

Each experiment lasted for a complete of a hundred and twenty minutes. We performed this experiment for three forms of systems: Stock android, MOSES while not SP changes (the system switched between two professional for each twenty minutes). Throughout every experiment, each ten seconds our service measured the amount of the battery and notes this worth. For every of the three thought of systems, we execute the check ten times and averaged the obtain values. The results of this experiment area unit reportable in fig. we note that the curves for three thought of systems behave equally. This shows that the actual fact that MOSES is simply running, or even switch between contexts does not incur an understandable energy overhead.

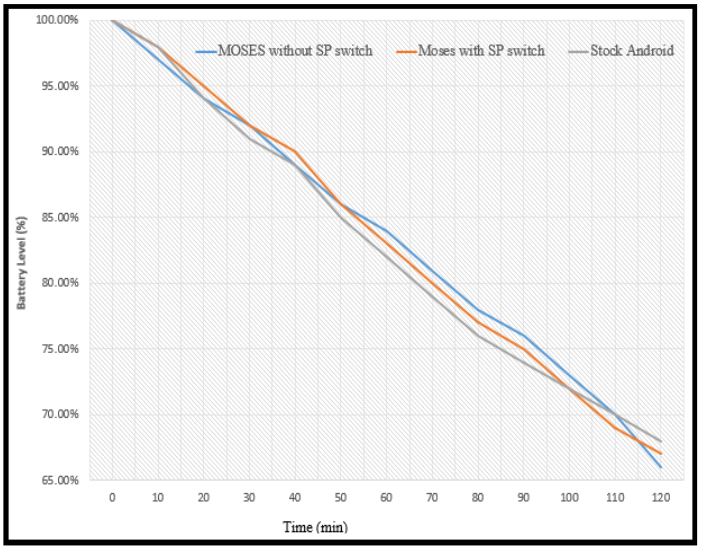

FIG .4 Energy overhead of MOSESApp

IV. SECUREPHONEAPP APPLICATION

As mentioned earlier, external security is provided by the SecurePhoneApp it works as follows as

before long as we have a tendency to install the application we have to activate all the service of Device Manager. At that time user should set password for the device and additionally provides the quantity of makes an attempt thus we will go for. Because it is given within the fig. user should enter correct password if fails, after third attempt information will get washed out. It is going to happen that unauthorized user is handling the device or device get purloined. Therefore the initiative of the theft is to require out the SIM card thus once taking away SIM all the information on the device can get format. However it additionally a chance that certified user needs to vary the SIM. Thus avoid formatting of info user will check in as a certified user additionally daily backup are sent to authorize email-address. So, the processed work is all concerning the way to secure the info on the device that is extremely sensitive. Many corporations willing to support worker-owned smart phones due to the increasing productivity of their employee. Hence it is necessary to produce such reasonably security to the information so that no one except authorize user will tamper with it accordingly.

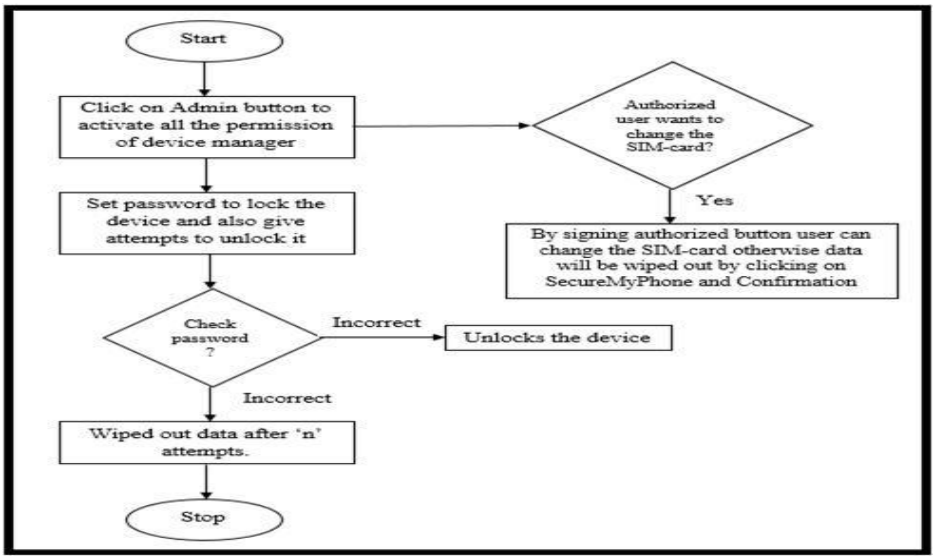

Fig. 5 Flowchart for SecurePhoneApp

A. Operating of SecurePhoneApp

1) As we tend to install the applying it shows home screen
2) User should click on "Admin" button so as to activate all the permissions that is given within the following figure,clicking on activate button allow the application to figure as DeviceManager for the device that controls all the permission regarding the device.
3) User can set screen security by clicking on "screen security" button so as to lock home screen of the device. It permits to line any of the following.;

- None
- Slide
- Face unlock
- Voice unlock
- Pattern
- PIN
- Password

FIG. 6 Home-screen of SecurePhoneApp

Also able to offer number of attempts to unlock it. Normally, android mobile phone works on three makes an attempt.

a)Another button is "Secure My Phone" which is employed to administer indication to the Device Manager that the SIM-card has been modified and it is time to wipe out the device information.Also,if the user fails to enter correct screen security, after "n" range of your time all the information are going to be tired. By the time SecurePhoneApp creates ".zip" file
to authorize user. this can be done by clicking on "confirmation" button that tells the Device Manager to start out the activity for clearing. Necessity to do so therefore is that it should happen unfortunately user has lost the device and somebody else except the owner has it.

b)Most people do not assume to come it back. Therefore what they are doing. They takeout the SIM-card 1st and stop working the device a few month when start using it. And nobody needs that somebody else is change of state with their sensitive information. in short everybody during this world need to stay their information safe from Now-a-days, mobile device is not as vital because the information gift on the device. This problem is solved by "Secure My Phone" and "confirmation" button. when beginning the activity, the projected app endlessly check for the presence of SIM-card and will the action accordingly

c)It could happen that owner of the device needs to alter the SIM-card and to forestall from clearing out information "Authorized Sign In" button is thereby clicking on it, user will change the SIM-card while not wiping the information.

FIG.7 Screenshots of SecurePhoneApp: (a) Device Administrator Activation, (b)Screen security to unlock device, (c) Confirmation to start activity of clearing data, (d)Authorised sign-in to remove SIM-card

B. User Rating based on Smartphone

The User Rating is given based upon the correct execution of both the application. The user rating shows that the applications ready to work well in several environment of android software.

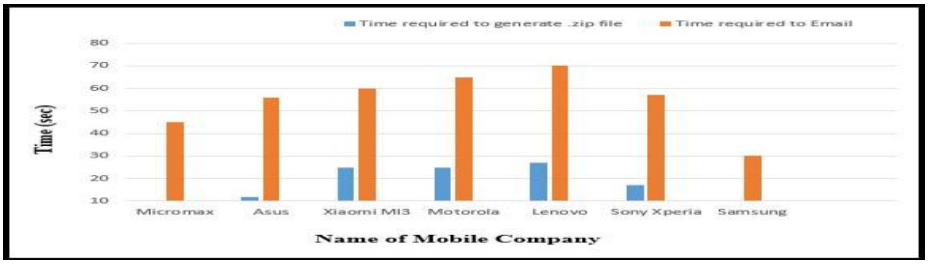

FIG. 8 User rating for SecurePhoneApp

V. CONCLUSION

To solve the problem of information being tampered by malicious app and by un-authorized user, the planned application ensures that end-user will send daily backup to their approved email- id. If they fails to enter the correct login credentials then data will be automatically emailed even within the second try of login credentials. Additionally if somebody stolen the mobile phone, theft can remove the SIM-card initially then while taking out the SIM data will get wiped out. The proposed work related to how can we secure the data gift on Smartphone internally and outwardly not a way to get mobile phone back. In today's world Purchasing a Smartphone is sort of easier than losing the valuable knowledge.

VI. ACKNOWLEDGMENT

We would like to show our sincere gratitude to them who directly or indirectly support us in completion of the manuscript.

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] http://www.s21.com/smartphones.html accessed on 17th Feb 2015.
[3] Y. Xu, F. Bruns, E. Gonzalez, S. Traboulsi, K. Mott, and A. Bilgic, "Performance Evaluation of Para-Virtualization on Modern Mobile Phone Platform," Proc. Int'l Conf. Computer, Electrical, and Systems Science and Eng. (ICCESSE '10), 2010.
[4] Yury Zhauniarovich and et. al., "MOSES: Supporting and Enforcing Security Profiles on Smartphones", IEEE Transaction on Dependale and Secure Computing, Volume-11, Issue-3, May-June-14, pp. 211-221.
[5] W. Enck, P. Gilbert, B.-G. Chun, L.P. Cox, J. Jung, P. McDaniel, and A.N. Sheth, "Taintdroid: An Information-Flow Tracking System for Realtime Privacy Monitoring on Smartphones", Proc. Ninth USENIX Conf. Operating Systems Design and Implementation (OSDI '10), 2010, pp. 1-6.

