International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2013): 4.438

Efficient Integrity Protection for Open Mobile Platform

Alankrita Ladage[1], H. Gurav[2]

Student, Dept. of C.E., SKNCOE, Pune University, India

Professor, Dept. of C.E., SKNCOE, Pune University, India

Abstract: ***As in these days mobiles are handy devices to use for computer application like internet surng, downloading songs, video etc. The security of mobile devices such as cellular phones and smart phone's has gained extensive attention due to their increasing usage in people daily life. The problem is challenging as the computing environments of these devices have become more open and general-purpose. We implement SEIP, Simple but yet Effective Solution for to protect trusted services of resource and maintain integrity level of cellular device using simple integrity protection rules.***

Keywords: Integrity Protection, SEIP, Open Mobile Platform

1. Introduction

Organizations implement security in accordance with their needs. An organization creates a security policy and uses security mechanisms to enforce the policy. A security policy is a statement that partitions the states of the system into a set of authorized or secure states and a set of unauthorized or unsecured states. The goal of an information system is to control access to the subjects and objects in the system. A security policy governs a set of rules and objectives needed by an organization .Like this person using mobile also needs a security. Mainly computer security is concerned with three aspects: confidentiality, integrity, and availability.

- **Confidentiality**: Preventing unauthorized users from gaining access to critical information of any particular user.
- **Integrity:** Ensures unauthorized modification, destruction or creation of information cannot take place.
- **Availability:** Ensuring authorized users getting the access they require

Mobile device security mainly focuses on porting PC counterpart technologies to mobile devices, such as signature- and anomaly-based analysis. Anomaly based detects the abnormal behaviour in the computer systems and computer networks. The deviation from the normal behaviour is considered as attack. Signature based matches the signatures of already known attacks that are stored into the database to detect the attacks in the computer system. Security of these devices has gained extensive attention due to their increasing usage in people daily life. But it has some limitations.

2. Related Work

Biba [5] is a hierarchical integrity policy, similar to Bell LaPadula but, interestingly, the exact opposite. It allows processes to both read and write to objects of the same integrity, no surprises there. Next it allows high integrity processes to write to low integrity objects, but not read them and last it allows low integrity processes to read high integrity objects but not write them. Though the use of Biba is very limited. The Clark-Wilson integrity model provides a different view of dependence [7]. Security-critical processes may accept low integrity information flows, but the program must either discard or upgrade all the low integrity data from all input interfaces. The Clark-Wilson model differs from the other models that are subject and object oriented by introducing a third access element programs resulting in what is called an access triple, which prevents unauthorized users from modifying data or programs. In addition, this model uses integrity verification and transformation procedures to maintain internal and external consistency of data. Low Water-Mark Mandatory Access Control (LOMAC) [4] is a Mandatory Access Control model which protects the integrity of system objects and subjects by means of an information flow policy coupled with the subject demotion via floating labels. In LOMAC, all system subjects and objects are assigned integrity labels, made up of one or more hierarchical grades, depending on their types. SEIP [1] is simple and efficient but yet effective solution for the integrity protection of cellular phone platforms. As all above all models have some disadvantages and limitation, the SEIP is now considering for integrity protection which has protection rules based on open mobile platform and application behaviour. It provides a set of rule which control flow of information according to different mobile systems. After studying all these techniques and comparing with each other come up with result, all techniques sanitize the low integrity data while SEIP do not.

Table 1: Comparison between different integrity models

Parameters	*Load Time*	*Biba*	*CW-Lite*	*Clark Wilson*	*LOMAC*	*SEIP*
High integrity process can read low integrity Process	No	No	Yes	No	No	Yes
High integrity process can read low integrity Network Data	Yes	No	Yes	No	No	No
Sanitation of low integrity data	No	Yes	Yes	Yes	Yes	No
Downgrade process integrity level	No	No	No	Yes	Yes	Yes
Upgrade process integrity level	No	No	No	No	No	Yes

3. Proposed System and Architecture

Traditional approaches for security and mobile platforms either prohibit information flow from low integrity process to high integrity process or sanitize low integrity data. This is not flexible for maintaining integrity level of device. Our general approach is based upon information flow control between trusted (e.g., customer, device manufacturer, and service providers) and untrusted (e.g., user downloaded through browser or received through Bluetooth and MMS) domains. First to preserve the integrity of a mobile device, we need to identify the integrity level of applications and resources. Basically, subjects are active entities that can access objects, which are passive entities in a system such as les and sockets. Subjects are mainly active processes and daemons, and objects include all possible entities that can be accessed by processes, such as, directories, network objects, and program. Note that a subject can also be an object as it can be accessed by another process, for example, being launched or killed. Second, many trusted processes on a mobile device provides functions to both trusted and untrusted applications, mainly the framework services such as telephony server, message service, interprocess communications, and application configuration service so for that we distinguish three types of trusted subjects on mobile platforms, according to their functionalities and behaviours.

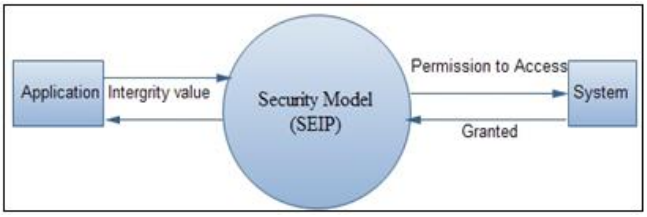


Figure 1: System Architecture

For integrity protection, it is critical to control how information can flow between high- and low-integrity entities. For providing integrity protection we use the following rules to identify integrity of application.

Rule 1: create$(s,o) \leftarrow L(o) = L(s)$when an object is created by a process, it inherits the integrity level of the process.

Rule 2: create $(s_1, s_2, o) \leftarrow L(o) = MIN((L(s_1),L(s_2))$: when an object is created by a trusted process s_1 with input/ request from another process s_2, the object inherits the integrity level of the lower bound of s_1 and s_2.

Rule 3: $can\_read(s,o) \leftarrow L(s) \leq L(o)$: a low-integrity process can read from both low- and high-integrity entities, but a high-integrity process can only read from entity of the same level.

Rule 4: $can\_write(s,o) \leftarrow L(s) \geq L(o)$: a high-integrity process can write to both low- and high-integrity entities, but a low-integrity process can only write to entity of the same level.

Rule 5: $can\_read(s,o_{1)} \leftarrow L(s) \geq L(o_1) \wedge writes\ (s,o_2) \wedge L(o_1) \geq L(o_2)$: a high-integrity process can receive information from low-integrity entity o1, provided that the information is separated from that of other high-integrity entities, and it flows to low-integrity entity o2 by the high-integrity process s.

Rule 6: $change\_level: L'(s) = L(o) \leftarrow read(s,o) \wedge L(s) > L(o)$: when a trusted subject reads low-integrity object, its integrity level is changed to that of the object. This rule is dedicated for the Type II trusted subjects, which usually read untrusted data (e.g., Internet content or media files)

4. Implementation

We have implemented SEIP on an application level for Android platform. Our implementation is built using simple security rules based on mobile operating system environment and application behaviours. Android classifies application permission into four protection level namely Normal, Dangerous, Signature and signature or System.

The SEIP firstly distinguishes the applications into three types trusted subjects on device according to their functionalities and behaviours. And then we present a demo application like call dialler or sim stealer to demonstrate integrity protection for open mobile platform through SEIP application. Figure represents an example of SEIP application.

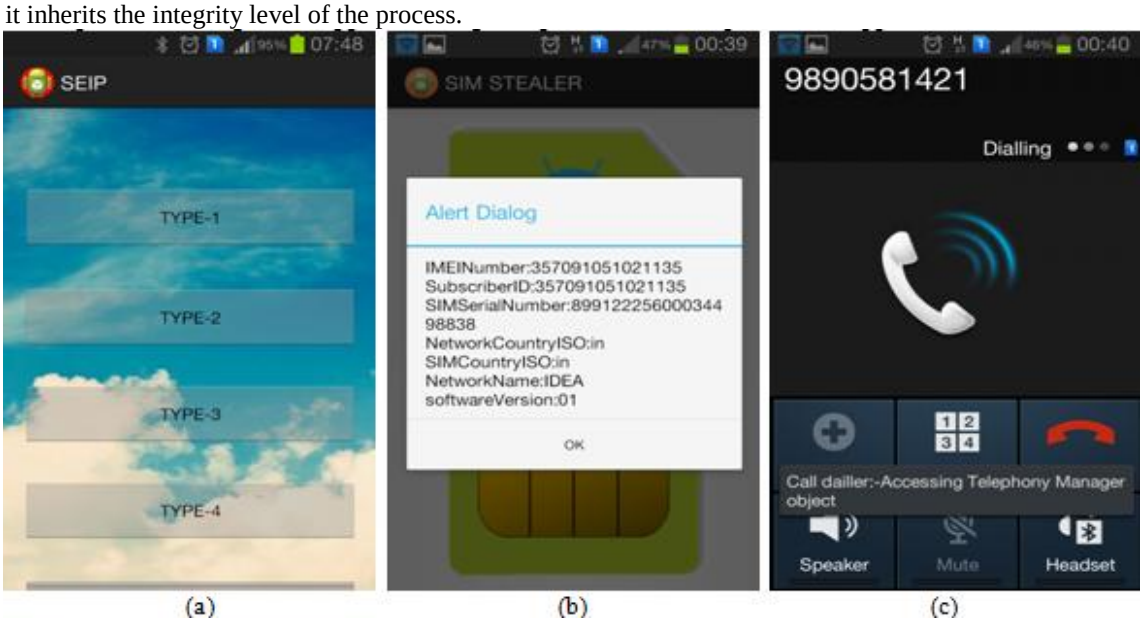


Figure 2: Here is an example demo application of SEIP. (a) SEIP (b) Sim Stealer (c) Call dialler.

5. Evaluation

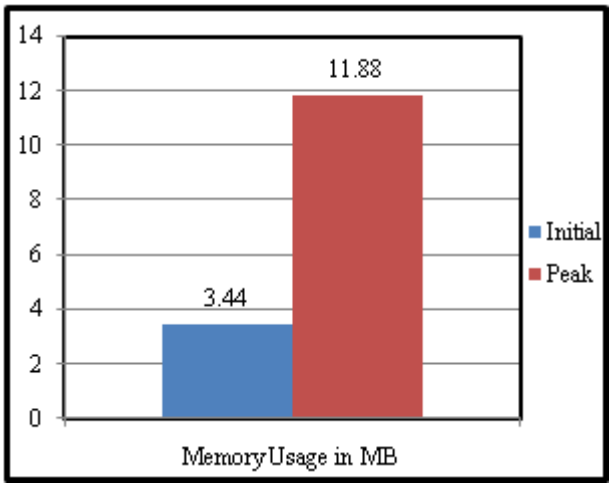
A. Security Evaluation

As this is implemented on android platform, it is challenging to test security mechanism completely. Instead, we test three types of attacks: typical attacks toward mobile platform integrity

- **Sim Stealer:** In this we send sim information to premium numbers so that he can steal the sim and this attack is detected on SEIP.
- **Call Dialler:** In this we send call to premium numbers so when mobile screen is off this attack is detected using SEIP.
- **Fake Downloader:** In this when application is downloading from any site and if on back end more files get downloaded then it is detected using SEIP.

B. Performance Evaluation

For evaluating performance of SEIP on application level of android platform, we consider the RAM usage of device. If you have minimum 512MB RAM and your application uses more than 480 MB then it is of no use. So it is require to use minimum RAM to broadcast the message through SEIP application. We measure it at initial and peak level. So as most it uses less than 13MB at peak level. Hence it proves that it uses efficient and optimal RAM.



6. Conclusion

In this paper, we present simple but yet effective and efficient solution for maintaining integrity protection for device. It easily distinguishes trusted and untrusted domains on both filesystem and memory space In this we do not sanitize low integrity data. It detects major the major threats from user downloaded or unintentionally installed applications. And proves that uses efficient and optimum memory to perform.

References

[1] Xinwen Zhang, Jean-Pierre Seifert and OnurAciicmez, "Design and Implementation of Efficient Integrity Protection for Open Mobile Platforms", IEEE Transactions on Mobile Computing, Vol. 13, no. 1, January 2014

[2] W. Enck, P. Traynor, P. McDaniel, and T.L. Porta, "Exploiting Open Functionality in SMS-Capable Cellular Networks," Proc.12th ACM Conf. Computer and Comm. Security (CCS), 2005.

[3] C. Heath, Symbian OS Platform Security. Symbian, 2006.

[4] T. Jaeger, R. Sailer, and U. Shankar, "PRIMA: Policy-Reduced Integrity Measurement Architecture," Proc. 11th ACM Symp. Access Control Models and Technologies (SACMAT), 2006.

[5] N. Li, Z. Mao, and H. Chen, "Usable Mandatory Integrity Protections for Operating Systems," Proc. IEEE Symp. Security and Privacy, 2007.

[6] P. Loscocco and S. Smalley, "Integrating Flexible Support for Security Policies into the Linux Operating System," Proc. USENIX Ann. TechnicalConf., pp. 29-42, June 2001.

[7] D. Muthukumaran, A. Sawani, J. Schiffman, B.M. Jung, and T. Jaeger, "Measuring Integrity on Mobile Phone Systems," Proc. 13th ACMSymp. Access Control Models and Technologies (SACMAT),2008.

[8] U. Shankar, T. Jaeger, and R. Sailer, "Toward Automated Information-Flow Integrity Verification for Security-Critical Applications," Proc. Network and Distributed Systems Security Symp. (NDSS), 2006.

[9] D. Venugopal, G. Hu, and N. Roman, "Intelligent Virus Detection on Mobile

[10] Devices," Proc. Int'l Conf. Privacy, Security and Trust, 2006