ISSN 23499842(Online), Volume 1, Special Issue 1(NCFCSIT 15), March 2015
International Journal of Innovative Trends and Emerging Technologies

# A STREAMLINED AND EFFICIENT INTEGRITY PROTECTION FOR MOBILE DEVICES

MS.J.MARY PRAVEENA[1], M.Tech., MS.P.SUGANYA[2], B.Tech.,
[1]Assistant professor, [2]PG Scholar(M.E), [1,2]CSE Department,
[1,2]Aksheyaa College of Engineering, (Anna University)kanchipuram, Tamil Nadu, India
[1]pravinajoseph@gmail.com, [2]suganyaprakasam@gmail.com

***Abstract*—Mobile phone security has become more important in the mobile computing applications. Most data breaches on the mobile devices are typically due to basic security failure, such as weak passwords, data encryption failure and application software vulnerabilities. This paper proposes the Streamlined and Efficient Integrity Protection (SEIP) for the effective protection of mobile devices. The SEIP is implemented by utilizing the malware detection using security policy, permission viewer of mobile apps, memory read and write options and secure SMS transactions. The proposed SEIP can prevent major types of attacks towards system integrity through mobile malware. The experimental result of the proposed SEIP, shows low overhead when compared to PC counterpart technology. The size of the security policy is less than 20 KB and requires less than 10 domains and types. The SEIP can be used within a commercially available Android-based smart phone.**

***Index Terms*—Mobile Computing, Mobile Phone Security, Integrity Protection, Security Policy, Smart Phone, Integrity Rules, and Malware Detection**

## I. INTRODUCTION

With the increase in the usage of mobile phones and smart phones more to surf the web, update social networking sites, shop & bank online, as well as cyber criminals and malware are increasingly targeting the mobile devices. Nowadays, the mobile phones are more prone to the malicious attacks. During the previous year, 5.6 million smart-phone users experienced some issues such as sending of unauthorized text messages or the fraudulent accessing of accounts without their permission. The rate of the presence of malicious software on the mobile devices seems to be increasing nowadays. Due to the increase in the storage amount of data on the mobile devices, the vulnerability of the mobile device to unauthorized access to the information is increased. Along with the data loss, the mobile devices carry the risk of introducing malware. The threat of cyber-attacks on mobile devices results in data loss, security breaches and compliance/regulatory violations.

The mobile Apps and messaging connect us, entertain us, and enrich our lives. But malware and aggressive permissions complicate the digital experience. Before the installation of many apps, they ask for permission to perform various actions. But all the permissions are not essential to the installation and usage of the apps. Small screens and lengthy privacy notices can make it tough to find out the security level and access control of the personal information collected by the mobile app developers and advertisers. The increase in the installation and usage of the number of apps on the mobile device increases the probability that some of the mobile apps may contain malicious code. With reference to the Kasper sky report, 98% of the mobile malware target the Android operating platform, and the number of variations of malware for the Android platform seems to increase to 163% in 2012. This Kasper sky report highlights the requisite for malware protection and security control over the mobile devices running the versions of the Android operating system and the concerns of traditional technology and non-uniform platforms.

More than 370 different malware, including viruses, root kits, adware, spyware and Trojans has been detected on various mobile devices, by the end of 2007. Most of the infection mechanisms comprise the integrity of the mobile device, by maliciously modifying the data or code on the device. The digital signature and anomaly based analysis are implemented in the mobile device security applications. But the signature-based security measures are not enough to defend against today's malware threats. The major objective of this paper is to confine the influence of the user installed applications, for securing the mobile device. The objective of this paper is to implement a simple and efficient security functionality that requires minimum interactions from the mobile user and the end user need not configure individual security policies. The security measures should be optimized effectively, when leveraged collectively through a centralized platform. This objective requires restricting the permissions of applications to access sensitive resources and functions of mobile operating system (OS), customers, device manufacturers and remote service providers, thus maintaining high integrity of

the mobile device, which usually indicates its expected behavior. This security policy establishes the rules for the proper use of mobile devices, to protect the confidentiality of sensitive data and integrity of the data and applications, to secure the mobile device.

This paper proposes a Streamlined and Efficient Integrity Protection (SEIP), a mandatory access control (MAC) based integrity protection mechanism of mobile devices. The SEIP is implemented by utilizing the malware detection using security policy, permission viewer of mobile Apps, memory read and write options and secure SMS transactions. The integrity threats on the mobile device are analyzed based on the infection mechanisms. Then the salient requirements for the security measures on the mobile devices are identified. A set of integrity protection rules is proposed to protect the mobile devices from the untrusted applications downloaded by the users. The proposed SEIP is deployed on the real-time Linux-based mobile devices. The size of the security policy is less than 20 kB in binary form and requires less than 10 domains and types. The SEIP can prevent major attacks toward the integrity of mobile device, through mobile malware. The experimental result of the proposed SEIP shows low overhead when compared to PC counterpart technology. This paper also describes about the implementation of memory read and write option for the storage disk, and implementation of the module to view the permissions of the installed applications.

The rest of the paper is organized as follows. Section II describes about the related work and section III shows an overview of SEIP. Section IV shows the evaluation of the SEIP and section V describes about the conclusion of this paper.

## II. RELATED WORK

Zhang et al [1] suggested a simple and efficient solution for the integrity protection of real-time cellular phone platforms, to overcome the disadvantages of the traditional integrity models on the performance and user experience controlled mobile devices. The prime security objective of the simple and efficient integrity protection (SEIP) is to secure the trusted services and resources from the third-party code. A set of simple integrity protection rules is recommended, based on the application behaviors and open mobile operating system environments. The proposed design controls the limited permissions and service convergence of user installed applications, and easily recognizes the borderline between trusted and untrusted domains on the mobile device platforms. High assurance of the platform integrity is achieved, while simplifying the policy specifications.

Bugiel et al [2] proposed a generic security architecture that serves as a effective and flexible ecosystem to instantiate different security solutions, for the android operating system (OS). The proposed Flaskdroid offers mandatory access control simultaneously on both middleware and kernel layers of android. Due to their completely different semantics, the alignment of policy enforcement on middleware and kernel layers is vital. An efficient policy language tailored to the specifics of Android's middleware semantics is presented. Kostiainen et al [3] described about the typical hardware-based security mechanism that provides the foundation for the mobile platform security. Many features are adapted from the older platform security architectures, by comparing the currently most prominent security architectures of the open mobile platforms.

Finally, a number of open problems is identified in designing the effective mobile platform security. Bugiel at al [4] proposed a trusted execution environment to secure the login credentials of the user. The design and implementation of an authentication agent and a wallet like password manager TruWalletM on the mobile phone are presented, without the need to depend the whole OS software. The compatibility of the proposed design and implementation to the existing standard web authentication mechanisms are preserved. Lange et al [5]presented a generic operating system framework that matches with the need for such hardware extensions. The proposed generic operating system framework allows the highly secure applications to run side-by-side with the virtual machine. The separation between the virtual machine and secure applications is ensured, based on the state-of-the-art microkernel. The proposed framework is evaluated by outlining the methods to solve the problems in current smartphone security. Roland et al [7] discussed about the overview of the new threats imposed by untrusted mobile phone applications and mobile connectivity. The security aspects of the mobile devices are explained, after performing the analysis of various application programming interfaces (API) for providing secure element access to the access control mechanisms of various mobile phone platforms. Two practical attack scenarios such as process for performing a denial of service (DoS) attach against a secure element and a process for the remote utilization of the applications on the secure element of victim without the knowledge of the victim, are highlighted finally.Becher et al [8] recommended a brief overview of the attack vectors for smartphone, using the mobile web browser and back end system, but also proposed the overview of the mobile user and hardware layer as attack enabler. The differences and similarities between "normal" security and mobile security is shown in this paper.

To overcome the problems in the traditional techniques, this paper proposes a set of simple integrity protection rules to leverage the limited permissions and service convergence of the user installed applications. The rules can easily identify the

border between trusted and untrusted domains on the mobile devices. The policy specifications are simplified by the proposed approach, while achieving high assurance of the platform integrity of the mobile devices. As the proposed SEIP can effectively prevent certain malware on the mobile device, the SEIP can be utilized within a commercially available Linux-based Smartphone.

## III. Streamlined And Efficient Integrity Protection

Traditional integrity models either prohibit the flow of information from low-integrity sources to high-integrity processes, or degrade the integrity level of high-integrity processes once they receive low data. However, the traditional approaches are not flexible enough for the security and performance requirements of mobile platforms. Specifically, the important feature of the mobile devices is that resources are maintained by individual framework services which run as daemons and accept requests from both trusted and untrusted processes, due to function convergence. In the LiMo platform [6], a message framework controls all the message channels between the platform and base stations and implements all message-related functions.Any program that needs to receive/send the SMS or MMS messages has to call the interfaces provided by this framework, instead of directly interacting with the wireless modem driver.

Many frameworks need to accept input from both trusted and untrusted applications during runtime. However, due to performance reasons they cannot frequently change their security levels. Thus, traditional integrity models are not flexible enough to support such security and performance requirements. Similarly, domain transitions used in SELinux are not feasible for mobile platforms. Toward this issue, this paper proposes some particular trusted processes can accept untrusted information, while maintaining their integrity level.

The untrusted applications can be installed and launched through a trusted package management tool and application launcher, during the installation and launching of applications on mobile platforms, while the installer and launcher still maintain their respective integrity. A set of integrity rules is defined to label the data, when these data are processed or handed over to these subjects. The general principles are identified to isolate untrusted access requests in typical service frameworks on the mobile device. An important feature that distinguishes our approach from the traditional information flow-based integrity models is that, we separate the data from different integrity levels within a trusted subject.

The SEIP is proposed to effectively achieve the integrity protection goals. This section presents design details and integrity rules of SEIP for the mobile platform, based on discussed security threats and our strategies. The main goal is to prevent software-based attacks from the application level.

### *A. MAC-based integrity model*

Logically, it is nearly impossible to specify and verify the policies for fundamental security properties such as system integrity. In the proposed design, permissions can be assigned different processes from the user, using the MAC-based security model. Furthermore, our proposed approach enables deeper security checks than simply allowing/denying the API calls. Finally, instead of considering extremely fine-grained permission control, thus requiring a complete and formal verified policy, this paper focuses on read- and write-like permissions that affect the system integrity status, thus make integrity verification feasible.

### *B. Malware detection using security policy*

The proposed SEIP approach simplifies the security policy specifications significantly and prevent certain malware efficiently, while achieving a high assurance of platform integrity. So that the proposed SEIP is utilized within a commercially available Linux-based smartphone. As the size of the security policy is less than 20 kB, it can be applied to all users of information assets, including Organization-Name employees, employees of temporary employment agencies, vendors, business partners, and contractor personnel and functional units irrespective of the geographic location. This security policy includes all information systems environments operated by organization-name or contracted with a third party by organization-name.Although this security policy obviously covers the responsibilities of users, it does not cover the matter exclusively.

The additional responsibilities are defined by other organization-name information security policies, standards, and procedures. All users are required to read, understand and comply with the other information security policies, standards, and procedures.

### *C. Permission viewer of installed apps*

The usage of the mobile apps has become increasing prevalent across the mobile users. Before downloading and installing the mobile apps, a list of permissions will be shown to the users. A list of the most commonly used permissions will be shown in the permission section. The list will explain the importance and prime objective of the permission viewer. The permission viewer enables a better understanding about the allowable levels of the mobile apps. In this module, the permissions of all installed mobile apps are checked, so that the user can easily view the abstract of the current state of the installed application.

*D. Securing SMS services*

To secure the SMS service, Rivest-Shamir-Adleman (RSA) algorithm is utilized for encryption and decryption. If there is some malware in the middle and it tries to intercept or view the body of the SMS, it will get some random bytes. So, the message will be secured during the transaction.

*1) RSA algorithm*

Step: 1 Compute m= a.b

Step: 2 Compute $\phi(m) = (a-1)(b-1)$

Step: 3 Choose an integer i, such that $1 < i < \phi(m)$ and gcd $(i, \phi(m)) = 1$

Step: 4 Compute $j = i^{-1} \bmod [\phi(m)]$

Step: 5 Publish the public encryption key: (i,m)

Step: 6 Keep secret private decryption key: (j,m)

To encrypt a message, the sender has to:
Obtain the public key of the recipient (i,m)
Represent the message as an integer p, in [0, m-1]

Compute: $s = p^e \bmod m$

To decrypt the ciphertext s, the recipient:
Uses his private key (j,m)

Compute: $p = s^j \bmod m$

Where a and b are two distinct random prime numbers.

*E. Read/Write options of file systems*

The proposed SEIP controls the ability to view or make changes to the contents of the file system, so that the file system can be mounted easily for the read/write operations. In the read only option, the data from the third party cannot affect the data of the file system. The read or write operations of the file system can be changed easily at any time.

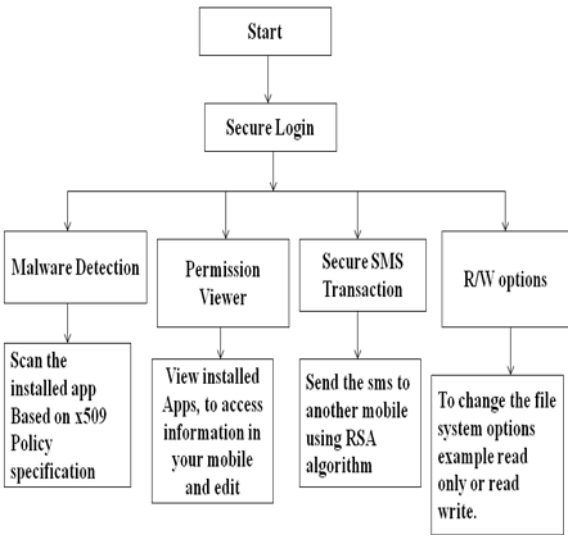


Fig.1 System Architecture of the proposed methodology

## IV. EVALUATION

*A. Security Evaluation*

It is a challenging task to test a security mechanism completely. Several Symbian malware that maliciously install files into the system directories and disable application manager has been found, such that applications cannot be launched and the device cannot be rebooted. In the proposed implementation, a user application can only be installed on untrusted side of the filesystem and it cannot write to trusted side.

Therefore, this type of attacks can be easily prevented. Wireless fidelity (Wi-Fi) faker is another malware which rapidly exhausts the battery of a mobile phone. With the power management functions of DevicePower-Notify() and SetDevicePower() in Symbian system, this malware sets the WiFi radio operating in the high power mode but shows an inactive status icon in system tray. We simulate this attack using similar APIs provided by the system framework in our platform. This attack causes the phone to run out of battery in less than 1 hour even it is not used at all. By controlling that only trusted processes can set the WiFi status, our proposed implementation prevents this attack successfully.

*1) Cross-service attacks:*

With multiple network interfaces on a single mobile device, there are multiple network connections and services available for mobile applications. Attackers can exploit from one network interface and get unauthorized access to another network service. For example, some existing mobile malware leverage vulnerability in Bluetooth stacks to infect devices and thus send SMS/MMS messages via their wireless interface such as Bluebug and Commwarrior. This attack is simulated with a dummy echo server which accepts inputs from Ethernet via universal serial bus (USB). With a simple strcpy buffer-over-flow vulnerability, a client can execute a shellcode which dials a premium phone number. However, when our security mechanism is enforced, as the echo server is untrusted process, the shellcode's phone call is denied by the telephony server.

*2) Bypassing security mechanism*

A common way to circumvent an access control mechanism is to exploit privileged applications. This implies two methods in our system: exploit trusted processes and hijack their security level to obtain privileged accesses, or exploit trusted service daemons which enforce access control policies. A code-injection attack is implemented with a user installed application. This untrusted application maliciously injects a shellcode to any trusted process withptrace() and changes the program register to the address of injected shellcode. The shellcode is then executed and the malicious process writes back the original code and registers thus the trusted process resumes. The shellcode simply disables the SELinux

en-forcement in the kernel, therefore, after this attack any untrusted process can get full privileges of writing to trusted side. Further, we implement a code-replacement attack to Type III trusted subject gconfd, which bypasses the security hook in database_handle _set()and always returns true (allowed) to any access request.

Thus, the untrusted process can set new key values to any high-integrity object such as phone unlock password and data network connection profile. We implement similar attack to the telephone server by replacing the permission check hook function in tapi_call_setup() with a trivial one, thus an untrusted process can make phone calls to any phone number including premium rate ones. All these attacks work as both trusted and untrusted applications run with the same uid in our evaluation thus untrusted process can ptrace to trusted process. When our security mechanism is enabled, both attacks failed with denied write operations (ptrace) from untrusted processes to trusted processes.

*B. Performance Evaluation*

Our policy size is less than 20 KB including the genfscon rules for file system labeling. Our policy footprint is tiny, when compared to the typical desktop Linux distributions such as Fedora Core 6. The small footprint of our security mechanism results from the simple way to identify the borderline between trusted and untrusted domains, and the efficient way to control their communications. The performance of our Android-based implementation with micro benchmarks is studied, to investigate the overhead for various low-level system operations such as file, process and socket accesses.

The performance evaluation results show that our security enforcement has much better performance than the counterpart technology on PC.The proposed method provides efficient malware detection using the simple and efficient security policy. The permissions of all installed mobile apps can be checked, so that the user can easily view the abstract of the current state of the installed application. The security of the SMS is ensured, using the efficient encryption and decryption process.If there is some malware in the middle and it tries to intercept or view the body of the SMS, it will get some random bytes.The proposed SEIP controls the ability to view or make changes to the contents of the file system, so that the file system can be mounted easily for the read/write operations. Fig 3-9 show the performance evaluation of the proposed SEIP for mobile devices.

Fig.2 shows the proposed SEIP for mobile devices. The SEIP is implemented by utilizing the malware detection using security policy, permission viewer of mobile apps, memory read and write options and secure SMS transactions.

Fig.2.SEIP security

Fig.3. SMS encryption using RSA

Fig.3 shows the SMS encryption using RSA algorithm.To secure the SMS service, the RSA algorithm is utilized for encryption and decryption. If there is some malware in the middle and it tries to intercept or view the body of the SMS, it will get some random bytes. So, the message will be secured during the transaction.

Fig.4.Encrypted process

Fig.4 shows the encryption process.The SMS message is encrypted using a public key. The encrypted

message is again decrypted to receive the original message.


Fig.5.Decryption process

Fig.5 shows the decryption process. The SMS message is decrypted using a private key.After decryption, the original message can be received without any errors.


Fig.6.Decrypted text

Fig.6 shows the decrypted text.The original message is obtained without any errors, after the decryption using private key. If there is some malware in the middle and it tries to intercept or view the body of the SMS, it will get some random bytes. So, the message will be secured during the transaction.

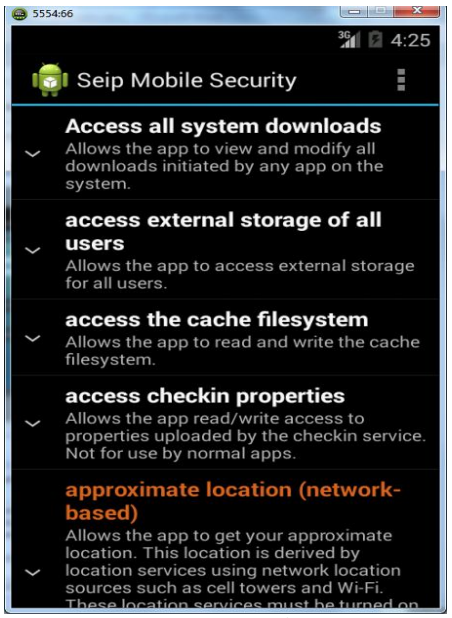


Fig.7.Permission Viewer of mobile apps

Fig.7 shows the permission viewer of mobile apps.The permission viewer enables a better understanding about the allowable levels of the mobile apps. In this module, the permissions of all installed mobile apps are checked, so that the user can easily view the abstract of the current state of the installed application.

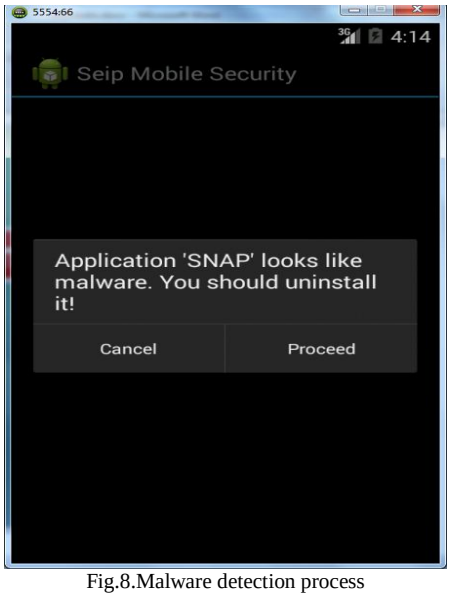


Fig.8.Malware detection process

Fig.8 illustrates the malware detection process. Detection of malware can be performed efficiently, using the simple and effective security policy specifications.

## V. CONCLUSION

A simple but yet effective and efficient security solution for integrity protection of the mobile devices is presented in this paper. Our design captures the major threats from the user downloaded applications or unintentionally installed applications, including

codes. Our solution enables very simple security policy development. Our design is implemented on a Android platform and effectiveness of the proposed design is demonstrated by preventing a set of attacks. The performance study shows that our solution is efficient, when compared to the counterpart technology on desktop environments. Furthermore, it is planned to provide more secure by creating profile manager and context provider for creating security profiles on android platforms and develop an intuitive tool for the policy development.

## REFERENCES

[1] X. Zhang, J.-P. Seifert, and O. Aciicmez, "Design and Implementation of Efficient Integrity Protection for Open Mobile Platforms," *IEEE Transactions onMobile Computing*,, vol. 13, pp. 188-201, 2014.

[2] S. Bugiel, S. Heuser, and A.-R. Sadeghi, "Flexible and Fine-grained Mandatory Access Control on Android for Diverse Security and Privacy Policies," in *Usenix security*, 2013, pp. 131-146.

[3] K. Kostiainen, E. Reshetova, J.-E. Ekberg, and N. Asokan, "Old, new, borrowed, blue--: a perspective on the evolution of mobile platform security architectures," in *Proceedings of the first ACM conference on Data and application security and privacy*, 2011, pp. 13-24.

[4] S. Bugiel, A. Dmitrienko, K. Kostiainen, A.-R. Sadeghi, and M. Winandy, "TruWalletM: Secure web authentication on mobile platforms," in *Trusted Systems*, ed: Springer, 2011, pp. 219-236.

[5] M. Lange, S. Liebergeld, A. Lackorzynski, A. Warg, and M. Peter, "L4Android: a generic operating system framework for secure smartphones," in *Proceedings of the 1st ACM workshop on Security and privacy in smartphones and mobile devices*, 2011, pp. 39-50.

[6] "LiMo Foundation," https://www.limofoundation.org, 2013.

[7] M. Roland, J. Langer, and J. Scharinger, "Practical attack scenarios on secure element-enabled mobile devices," in *2012 4th International Workshop onNear Field Communication (NFC)* 2012, pp. 19-24.

[8] M. Becher, F. C. Freiling, J. Hoffmann, T. Holz, S. Uellenbeck, and C. Wolf, "Mobile security catching up? revealing the nuts and bolts of the security of mobile devices," in, *2011 IEEE Symposium onSecurity and Privacy (SP)*, 2011, pp. 96-111.

[9] M. Ongtang, S. McLaughlin, W. Enck, and P. McDaniel, "Semantically rich application-centric security in Android," *Security and Communication Networks*, vol. 5, pp. 658-673, 2012.

[10] Z. Sanaei, S. Abolfazli, A. Gani, and R. Buyya, "Heterogeneity in mobile cloud computing: taxonomy and open challenges," *IEEECommunications Surveys & Tutorials*, vol. 16, pp. 369-392, 2014.

[11] C. Gehrmann, H. Douglas, and D. K. Nilsson, "Are there good reasons for protecting mobile phones with hypervisors?," in *2011 IEEEConsumer Communications and Networking Conference (CCNC)*2011, pp. 906-911.

[12] N. Santos, H. Raj, S. Saroiu, and A. Wolman, "Using ARM trustzone to build a trusted language runtime for mobile applications," in *Proceedings of the 19th international conference on Architectural support for programming languages and operating systems*, 2014, pp. 67-80.

[13] R. Mayrhofer, "An architecture for secure mobile devices," *Security and Communication Networks*, 2014.

[14] S. Srivastava and G. Nandi, "Self-reliant mobile code: a new direction of agent security," *Journal of Network and Computer Applications*, vol. 37, pp. 62-75, 2014.

[15] C. Mulliner, S. Liebergeld, M. Lange, and J.-P. Seifert, "Taming Mr Hayes: Mitigating signaling based attacks on smartphones," in *2012 42nd Annual IEEE/IFIP International Conference onDependable Systems and Networks (DSN)*, 2012, pp. 1-12.