Dynamic Change Reporting of Platform Configuration

Hoda Ghazaghi[1]
[1]Shahed university, Iran, Tehran,

Mohammad-Ali Doostari[2]
[2]Shahed university, Iran, Tehran,

ABSTRACT

TCG group introduced the Remote Attestation Protocol, which has a weak point that makes it vulnerable to a masquerade attack. In this paper, a new method is introduced for improving the security of this protocol against masquerading attacks. The security of the improved protocol is analyzed using AVISPA tools. Advantages of the improved protocol include a reduced number of messages and lower cost, which prevents useless communication. Furthermore, an improved mechanism for measuring and reporting the changes is recommended. Combining the above mentioned, improved protocol with the improved integrity measurement and reporting mechanism can solve the existing problem in certain critical applications.

Keywords

remote attestation, Trusted Platform Module, integrity measurement, masquerading attack, formal analysis

1 INTRODUCTION

With the widespread use of remote and electronic interactions, the need to protect systems and their integrity against existing threats has become evidently more important. Electronic interactions such as e-banking service, e-government service, interaction of organizational servers that contain information of sensitive government sites are examples of such interactions. In this context, securing communication channels is not sufficient for the overall security since endpoints, which contain plain data, are more and easier exposed to malicious treatment. Therefore, it is necessary to find some solutions for securing the endpoints in their communications and the related access control.

On this basis, TCG1 group has introduced one security hardware module called TPM2 that has functional cryptography and some other special capabilities. This module can be used as a root of trust in digital systems. One of the main capabilities of the module is facilitating remote attestation. For the sake of privacy, remote attestation can be done in an anonymous manner. For this, the remote entity can only investigate and find out the genuineness of the platform which its integrity information is received without understanding the identity of that platform.

The purpose of the remote attestation is so a platform can be able to report its status to the challenger in a manner that the challenger can evaluate the integrity of the platform and make a decision based on its policies and reference configurations. In realization of remote attestation, another module cooperates with TPM so that the module measures the component and stores the measurement in TPM’s configuration register, called PCR3. Also, the results of the measurement are stored in a memory structure called SML4. The values of SML and PCR register are used for integrity reporting of remote attestation. TCG group has introduced a boot process that measures every loaded component before passing control to it. In the case that results match the expected values, the boot process continues.

The remainder of the paper is organized as follows. In section 2, remote attestation is defined and its components and challenges. Section 3 surveys the works done so far in this context. Section 4 describes one of the prevailing challenges in remote attestation (masquerading attack), and proposes the improved protocol. The security of the proposed protocol and accuracy of our claim is verified by the AVISPA5 tool. Section 5 presents an improved integrity measurement and reporting mechanisms, which elaborate the reports of changes that occurred during platform configuration. Finally, in section 6 we elaborate on the usages of improved protocol along with improved integrity measurement and reporting mechanism as a solution for problems in some current applications.

2 REMOTE ATTESTATION

In remote attestation, a platform wants to prove to a remote party that its integrity is good enough for interaction and communication. So the remote party verifies whether it is safe or not. Integrity is a binary attribute that indicates whether the program and its environment are modified illegally or not. This process is done based on TPM. TPM module has a unique asymmetric key pair called EK[6], which often the manufacturer generates it and gives it in TPM. If TPM module uses this key in general interactions, the TPM and thus the platform would be traceable. So TPM generates alias key called AIK[7] and uses it in the interactions. The AIK certificate is issued by TTP called Privacy-CA and assures that the identity key is indeed TPM hosted and does not contain any information that links the certificates to the specific platform hosting the AIKs.

2.1 Integrity Measurement and Reporting Mechanism

All of the questions like which parts of runtime environment are measured, how and when they are measured, how the measurements are stored securely and the way the integrity reporting are done, what structure and component are needed, would be answered in measurement architecture. In fact, Integrity measurement mechanism provides the needed data for remote attestation protocol and integrity reporting mechanism uses the protocol.

2.2 Receiving AIK Certificate

As explained earlier a Privacy CA takes the role of a trusted third party and must be trusted by both parties in the remote attestation process. The Privacy CA has an operational mode which is adjusted by its policy and defines two important attributes: who can gain an AIK certificate and how much information can be saved about AIK certificate request and the issued certificate. With using the presented structure of TCG group [18], the protocol for requesting an AIK certificate is summarized and shown in Fig. 1.

1. Owner→ Privacy CA : E (session key1)$_{\text{privacy CA public key}}$, E (TCPA Version, new pub key, identity label, sign (new pri key, TCPA_IDENTITY_CONTENTS), endorsement credential, platform credential, conformance credential)$_{\text{session key1}}$

2. Privacy CA→ owner : E (hash (new pub key), session key2)$_{\text{EK pub key}}$, E (identity Credential)$_{\text{session key2}}$

Fig. 1 Protocol of receiving AIK certificate

In Fig. 1, TCPA_IDENTITY_CONTENTS structure contains identity key's public key and hash (identity label, privacy-CA public key). Also, the Privacy CA verifies the TCPA_IDENTITY_CONTENTS structure to determine whether it was signed by a private key matching with public key in identity request or not. The target of the certificate request (which privacy CA) is determined by Privacy CA's public key Hash which is in the TCPA_IDENTITY_CONTENTS and prevents the sending of one certificate request to several Privacy CAs.

2.3 Integrity Reporting Protocol

The integrity reporting protocol presented by TCG group [20], which is shown in **Error! Reference source not found.**, has some stages: A Challenger party requests PCR registers from another platform. Then an agent on the platform collects SML entries and asked PCR values from the TPM. The TPM signs PCR values using an AIK. The Platform Agent collects credentials about the TPM and sends the signed PCR value, SML entries and Credentials to the Challenger. Finally, The Challenger verifies the received response by computing the measurement digest and comparing it with PCR value. The platform credentials are evaluated and the signatures are checked.

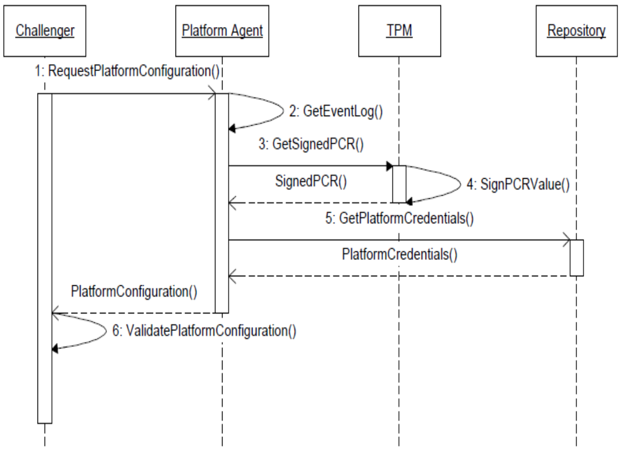

Fig. 3 Remote attestation protocol of TCG group [20]

Currently, there are challenges in actualization of remote attestation and researchers are working on the challenges. There are two categories for the Challenges: integrity measuring architecture and integrity reporting protocol. Some of the challenges in area of integrity reporting protocol are enumerated: security analysis and existing attacks on the protocol such as privacy aspect in sending data, lack of openness, and static integrity reporting. The static integrity reporting belongs to a loading time (not runtime). This time discrepancy is a limitation since the code can be compromised in runtime and the process can run in unsecure environment.

2.3.1 *Integrity Reporting Protocol Attacks*

In integrity reporting protocol, freshness of data and sending of data over a secure and authentic channel must be assured. The integrity reporting protocol of TCG is vulnerable against masquerading attacks. Such attacks are shown in **Error! Reference source not found.**Fig.. Therefore, malicious entity can exploit another platform's configuration and present itself as a safe entity and interact with part A. The malicious entity can be in two forms: part A attests part B and after doing attestation, part C inserts itself in interaction with part A. This attack can be prevented by establishing a secure channel between attestation parties so that only the two parts know the key of secure channel.

In Second form, part B is a malicious entity and forwards the attestation massages to a safe part C and then forwards receiving attestation data from part C to part A. After doing attestation, part B interacts with part A. We enhance the protocol to avoiding such attacks.

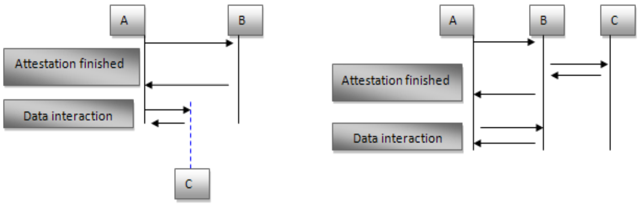

Fig.4 Two states of masquerading attack

2.3.2 *AVISPA Tool*

AVISPA is one of the model checking tools which constructs a great but finite number of possible protocol behaviors and checks them against a set of correctness conditions. This method often constructs a complete state diagram of the protocol behaviors and then performs an exhaustive search on it to find a path containing a state where correctness conditions are violated.

TCG attestation protocol is written the in HLPSL[8] language and analyzed with the AVISPA tool. Security goal in the written code contains a challenger authenticates the attester on a nonce with value n. AVISPA shows the above attack for the code and its security goal.

3 RELATED WORK

Reiner Sailer et al. [13] expressed the integrity reporting protocol used by challenging party C to validate integrity claims of entity AS in Fig. 2. To prevent masquerading attack, Frederic Stumpf et al. [6] proposed to integrate a key agreement protocol into integrity reporting protocol. In fact, he used Diffie-Hellman for establishing a shared key and consequently authentic channel between two parties. Based on this idea, each part generates a Diffi-Hellman pair and attester signs public part of Diffi-Hellman along with nonce and PCR by its AIK private key. Therefore the attacker cannot send other's configuration as itself and cannot be in their interactions, because it cannot generate the shared key. If entity B be a man in the middle attacker, it will be failed because it cannot modify the response of AS.

1. C : create non-predictable 160bit nonce
2. C → AS: ChReq(nonce)

3a. AS: load protected AIK$_{priv}$ into TPM

3b. AS: retrieve Quote=sig{PCR, nonce}AIK$_{priv}$

3c. AS: retrieve Measurement List ML

4. AS → C : ChRes(Quote, ML)

5a. C: determine trusted cert(AIK$_{pub}$)

5b. C: validate sig{PCR, nonce}AIK$_{priv}$

5c. C: validate nonce and ML using PCR

Fig. 2 Integrity reporting protocol [13]

Song Cheng et al. [17] used TLS/SSL protocol to establishing an authentic channel. The Pre_Master_Secret value is signed by AIK private key and the remote attestation data is sent over TLS handshake. For efficiency and privacy aspect, SML will be sent to challenger after establishing Master key. Bounding the Pre_Master_Secret value to integrity reporting protocol ensures that integrity reporting protocol describes the system in TLS/SSL channel.

Xinwen Zhang et al. [20] proposed that TPM measures booting system and secure kernel in the boot time and stores the hash results in PCR registers. After that, the secure kernel measures the application's code (Before application starts) and stores hashes locally. TRM is a component of application or a running service in OS user space. Secure kernel generates a key pair for that and its public key will be certified by the AIK. For remote attestation, TPM signs the PCR values and secure kernel signs the TRM integrity value. Then both are sent to a remote party. Challenger verifies signature and AIK certificate and secure kernel certificate. If all of them are valid, TRM and secure kernel will be trusted.

Trent Jaeger et al. [19] expressed that the application can be isolated in an information-flow sense from most other applications on the system. Without having such dependency information, the remote party must conclude that any unknown and untrusted application that is loaded, may compromise the target application. By using the information flow, it is possible to find out where the runtime inputs come and which target application has dependency to them. So the measurement process can limited to related elements.

Reiner Sailer et al. [13] proposed a Linux measuring architecture called IMA based on integrity reporting of TCG. He extends the measure-before-load principle of TCG and supports Automatic measurements of executable files and Manual (application-induced) measurements of important input files. This architecture maintains a list of hash values covering all executable content loaded to runtime from booting time. IMA integrates the measurements of executable content and configuration files with measurements of BIOS, Boot loader, OS and retrieves the hashes for the challenger party.

David Safford and Mimi Zohar proposed a Trusted Linux Client [3]. For protection of integrity, TLC has loadable kernel modules called SLIM, EVM and hardware module TPM. TPM module is used for hardware based public key management and boot time measurement and secures storage of data. The master key that the other keys are generated based on it is generated and sealed by TPM. EVM module checks security characteristics including the authenticity, integrity of an application but cannot determine they will operate properly with any given input data (possibly malicious). SLIM module is considered for managing access control to process and program, and limit executable privileges based on the executable trust attributes.

Reiner Sailer et al. [14] proposed that a running system has a fresh aggregate fingerprint and measurement list at any time, reflecting any software and booting hardware that was involved in the current system state since the last reboot. Also, he expressed how the process of measurement and remote attestation was done and improved the process via caching the address of files in measurement request.

4 PROPOSED REMOTE ATTESTATION PROTOCOL

For the sake of privacy in trusted computing applications, the attester will be anonymous in its transactions. Anonymity means that it does not reveal the real identity for other party and generates an alias name (AIK key pair). Based on definitions of TCG group [19], the field of subject in AIK certificate must be empty and certificate issuer (Privacy-CA) must use alternative name subject. The alternative name contains three values: TPM manufacture and model and version numbers from EK certificate, platform manufacture and model and version numbers from platform certificate, TPM identity Label provided to the Privacy-CA by the TPM owner.

One of the security challenges of the protocol is establishing authentic channel between two parties. In other words, it should be assured that the sender of the trigger is the entity which reports its configurations and is the same entity in data interaction. The integrity reporting protocol of TCG is unable to establish the authentic channel and is vulnerable against masquerading attack. In this article, two proposed protocols are recommended that will help address this issue. In the first proposed protocol, using the AIK key pair for establishing shared key is suggested. In the second proposed protocol, the shortcoming in the first protocol will be overcome. In both proposed protocols, when the challenging party receives a request from an entity, corresponding to its policy, it sends a nonce value for it.

4.1 First Proposed Protocol

This protocol uses AIK key for establishing shared key. The advantage of the protocol is having lower cost compared with previous works because there is no need to generate new keys for each side and there is no need to use TLS/SSL protocol. The TPM that has AIK private key can do decryption and achieve value n2 and compute shared key. The first proposed protocol is shown in **Error! Reference source not found.**.

1. C→ A : trigger (request message)

2. A→ C: challenge Request (n)

3.a C : loadkey (AIK$_{pri}$)

3.b C: retrieve quote=sig{PCR, n}AIK$_{pri}$

3.c C: get stored measurement log SML

3. C→ A : challenge Response(quote, SML), cert(AIK$_{pub}$)

4. A → C : E(n2)$_{AIK\text{-}pub}$, generate key-sym=Hash(n, n2)

5. C → A : E(n2)$_{key\text{-}sym}$, generate key-sym= Hash(n, n2)

6.a A : verify n2 and key-sym

6.b A: valiadate cert(AIK_{pub})

6.c A: validate sig{ PCR, n}AIK_{pri}

Fig. 3 First proposed attestation protocol

Part A verifies that the other side in communication is the integrity reporter. If the verification is successful, then A verifies the integrity reporting data. After attestation is done, the two parties interact with each other with key-sym key. SML can be encrypted by key-sym and sent in line 5 for privacy aspect.

4.1.1 Formal Analysis with AVISPA

The proposed protocol is written in HLPSL[9] language and is analyzed with AVISPA tool. The defined Security goal contains that part A authenticates part C via n and n2 (verify that part C is the same that it claims) and only two sides know key-sym. The results illustrate that this protocol is safe and masquerading attack is removed.

4.1.2 Analysis of First Proposed Protocol

Following attestation, the malicious entity C' cannot insert itself into communication, because it does not have the shared key. In masquerading attack which entity C is malicious and wishes to bypass the messages to another safe entity F, the attack will be removed because n2 is encrypted by AIK public key and only entity F can do decryption and compute shared key and can interact with entity A. The entity C only forwards the messages.

Suppose that malicious entity C' is man-in-the-middle attacker and after party C sends the message 3, generates message 4 and sends it to C. Since entity C' cannot have n2 and consequently shared key, it cannot generate message 5 for party A. So party A understands and attack fails. Because of remote attestation concept, the attacker does not want to send its configuration and does not be a complete man-in-the-middle. This proposed protocol has a problem that an AIK key is used for both signing and decryption and this is not desirable for TPM. So another protocol is suggested that has more complexity regarding this but solves the problem and eliminates the threat of masquerading attack.

4.2 Final Proposed Protocol

Several goals are considered in this protocol that contains: first. Integrity reporting is done based on the adjusted policy of attester, so attester can determine how much information will be disclosed for the remote entity. Second. There is a secure and authentic channel between two parties which prevents masquerading and man-in-the-middle attack. Third. Challenging party only interacts with sender of trigger.

Proposed protocol has three parts: 1- handshaking.2- integrity reporting.3- data interaction (if the configurations match with reference values).

The basic idea of the protocol, which is shown in **Error! Reference source not found.**, is establishing needed parameter between two sides in early handshake. Specially, establish the AIK certificate securely so a malicious entity C is forced to report its configuration by its AIK. In handshaking, AIK certificate is sent and attester signs its Diffie-Hellman key. Also, a secure channel is established that can be used in the future interactions. The final protocol is robust against masquerading attack.

1. C→ A: trigger, Kc, cert(AIK_{pub}) , sig{Kc}$_{AIKpri}$

2.a A: validate Kc

2.b A: generate K_{ac}=$(Kc)^a$

Handshake phase

3. A→ C: Ka, cert(K_{Apub}), E{nonce1}$_{Kac}$

4.a C: loadkey(AIK_{pri})

4.b C: retrieve quote=sig{PCR, nonce1}AIK_{pri}

4.c C: get stored measurement log SML

Integrity Reporting phase

4.d C: generate K_{ac}=$(Ka)^c$

5. C→ A: Challenge Response(quote, E{SML}$_{Kac}$)

6.a A: validate nonce1 by Kac

6.b A: validate cert(AIK_{pub})

6.c A: validate sig{PCR, nonce1}AIK_{pri}

6.d A: validate SML using PCR

Integrity Reporting phase

Fig. 4 Final proposed attestation protocol

Ka= g^a mod m is the public part and "a" is the private part for side A. cert(K_{Apub}) is a certificate for public key belong to side A. Side C makes a decision based on the amount of information disclosed for side A. Since the Diffi-Hellman public part signature by AIK cannot replaced and modified, in line 1 side A assures that public part Kc belongs to a platform that has the AIK private key. Then side A computes shared key and generates the encrypted random number in line 3 to ensure that side C has computed the same shared key.

4.2.1 Formal Analysis of Proposed Protocol

The proposed protocol has written in HLPSL language and analyzed it with AVISPA. The security goals were defined so that side A authenticates side C via nonce1 and the established shared key is only known for two parties. The results show that the goals were achieved and no attack was found on the protocol.

4.2.2 Protocol Analysis of Proposed Protocol

Suppose that M is man-in-middle attacker. In exchange of public part keys, it can replace its public key Km but cannot sign it with AIK private. If the attacker wants to replace AIK certificate and generate signature on Km with itself AIK (in line 1), it will be enforced to report itself to configurations. So the attack will fail.

For masquerading attack, suppose that side C is malicious entity and wishes to send another's configuration as its own configuration. If side C sends its own data (Diffi-Hellman public part, AIK cert, signature on Diffi-Hellman public part) in line 1, it must report its configuration. Otherwise, the AIK certificate does not match with the signed AIK. If side C sends the data of side B in line 1, C only forwards the massages and cannot insert itself in interaction. So the masquerading attack will fail. The possibility of sending data of side B in line 1 is low because synchronic interaction with side B is needed. In such situation in related works, side A interacts with side B and does not know that side B is not the initial requester (trigger sender). The advantage of this protocol is avoiding useless communication and using processing power of side A.

The value of nonce1 is used as a challenge in remote attestation and caused side A assures that side C has the correct key. Since the SML contains the configuration data for the entity, it is encrypted by Kac then sent.

The proposed protocol has two advantages toward related works: the number of messages in the protocol is reduced (the number of messages in the proposed protocol is three and in the previous work were four). The cost of the protocol is lesser than related work, because doing hash function and generating second random number is not needed. The other advantage is that the useless interactions are prevented.

4.2.3 Removing Concern about Symmetric Cryptography

A symmetric key is established between two sides in proposed protocol. TPM has functionality of symmetric cryptography but based on definitions of TCG group [18], CPU does symmetric cryptography of general interactions. There, when the challenging party verifies that attester is safe and begins to communicate with it, the shared key is used by CPU. Since the attester is known safe, CPU usage of the key is not a critical problem. But if integrity of the attester is changed during the communication, malicious software can get the key and plaintext data. This problem existed in related previous works, but it is solved it in this article via reporting dynamic changes during the interaction. The challenging party verifies it again and if the verification be successful, the interaction will continue and the key will be used.

5 PROPOSED MECHANISM FOR INTEGRITY MEASURING AND REPORTING

The advantage of the proposed mechanism is that attester reports the changes in its integrity during communication, thus it removes the concern about doing symmetric cryptography by CPU. So, the proposed mechanisms satisfy the following characteristics:

- The integrity reporting data provided for the verifier is new and complete. So that it reports the current platform state and contains all the measurements from booting time to the time that attestation is performed.

- The attester is allowed to identify the verifier and to determine based on its policy that how much information discloses for the verifier.

- It provides the dynamic changes reporting of attester platform during the communication to the verifier.

All of executable codes which are loaded via OS or dynamic loader or application should be verified. The kernel is modified so that loaded executable codes and static configuration files are measured. The measurements are done before virtual memory maps.

In the proposed mechanism, measurement of booting time (BIOS to kernel) is done by TCG measurement principle and a kernel module is introduced for measuring the run time. This kernel module is loaded along with kernel and its trustworthiness is verified. Consequently, a Chain of Trust is established and the kernel and the module are measured reliably. The measurements during boot are stored in TPM. In fact, the root of trust is extended from TPM step by step. After boot is done, the kernel and the kernel module are trusted and kernel module measures and extends the results of measuring to TPM’s registers.

Afterwards, the loaded kernel module is responsible to response the measurement requests. The module writes the measurements in measurement list in kernel and extends PCR registers. This measurement list is used for integrity reporting and is used as a cache to improve the measuring process. Integrity measurement requests can come from user space (by application) and kernel. Integrity measurement request contains a pointer to location of the file should be measured (measuring agent informs about the location should be hashed).

Upon receiving a random number from challenger (attestation request contain a challenge), the attester provides the response. Therefore, the integrity reporting component communicates with TPM module and retrieves a signature on random value and related PCR. The response contains collected measurement list and a signature on PCR and random value. In the response, the running environment is described since startup until current time for challenger.

5.1 Mechanism of Reporting Dynamic Changes and Verification

Before data interaction, platform’s integrity is attested and verified. The change in integrity and configuration means that the software components affected the running environment are changed (executable file or sensitive file requested by applications) and new file is loaded or means that old loaded file is changed. In this situation, the challenger should be informed about the changes. So the attester can decide based on its discretion to continue interaction or not. So, a mechanism for reporting the changes in integrity of platform is proposed.

The measuring module in kernel is always ready to respond to measurement requests and has a list of measurements since booting to current time. This list reflects the files that are loaded to platform. Upon receiving measurement request, the module measures the file and searches the produced measurement in the list. If the measurement is new (does not exist in the list), the module will add it to the end of the list and will extend it to PCR register. If the measurement exists in the list, the module concludes that the file has been measured previously and it does not change. Since the measurement does not contain any new information about integrity of the platform, the module does not add it to list and does not extend it to PCR register in order to prevent increase of operations cost and volume of list.

Integrity reporting component does the proposed remote attestation protocol and collects the needed data. This component differs between remote attestation request before interaction and remote attestation request during interaction. Therefore, the component operates in two states and stores the list (SML) in every report.

Upon receiving the request from communication software in the same platform, the integrity reporting component operates in state 1 and carries out the proposed protocol. If remote attestation is successful, the communication will start. The state 2 of integrity reporting component is activated when the measuring module calls TPM Extend command during the interaction. Then, the notification message is sent to challenger to inform the changes. The challenger sends random value. The component of attester compares current SML against stored SML from last reporting and achieves new parts of SML. As shown in Fig. 5, attester platform only sends new parts of SML (instead of sending total measurement list) in order to improve the amount of sending data.

1. Attestor → Challenger: notification
2. Challenger → Attestor : n
3. Attestor → Challenger: sig(n, PCR)$_{AIKpri}$, sequence of new hash of files

Fig. 5 Reporting dynamic changes

In the proposed mechanism, the integrity reporting component can be a part of application layer or a part of TCB[10] as a kernel module.

If integrity reporting component is in application layer, it will communicate with TPM via TSS interface and retrieve the needed values from it. The component is not trusted. If the component manipulates the integrity reporting protocol, the challenger party will become aware. The integrity reporting component sends the changes and the communication software uses the established key. We eliminate the concern about key generation and key maintenance in this component via reporting the changes. There, it is proposed that notification massage (which has a random value) is triggered with TPM-Extend command and is generated with the trusted measuring module. So that the measuring module waits to receive a signal contains the random value signed with private key of challenger. The received signal shows that challenger party received the changes and assures that integrity reporting component sends the changes. There, the certificate of the challenger party and related private key is needed.

If integrity reporting component is in TCB as a kernel module, its trustworthy will be checked in trust chain and key generation and maintenance will be protected. In fact, TCB contains the components that their trustworthy are verified during boot and it is needed to be trusted. This component does attestation protocol and if remote attestation process is successful in state 1, the reporting module derives a key from established key and gives it to communication software (to have a secure channel). When the reporting component is activated in state 2, it generates a notification message that contains a random number. Upon receiving a signal from the challenger that contains the encrypted random number with Kac, the reporting module is assured that the challenger takes the changes. In proposed mechanism, it is preferred that the reporting component be in TCB.

The challenger party receives an ordered sequence of hashes and adds them to the end of the last SML with the same order. Then verifies the new SML via new PCR to assure that it has not been modified. The challenger compares the hashes against reference values and decides based on its policy to continue to interaction. Fig. 6 shows the view of proposed integrity reporting mechanism and process of proposed remote attestation protocol.

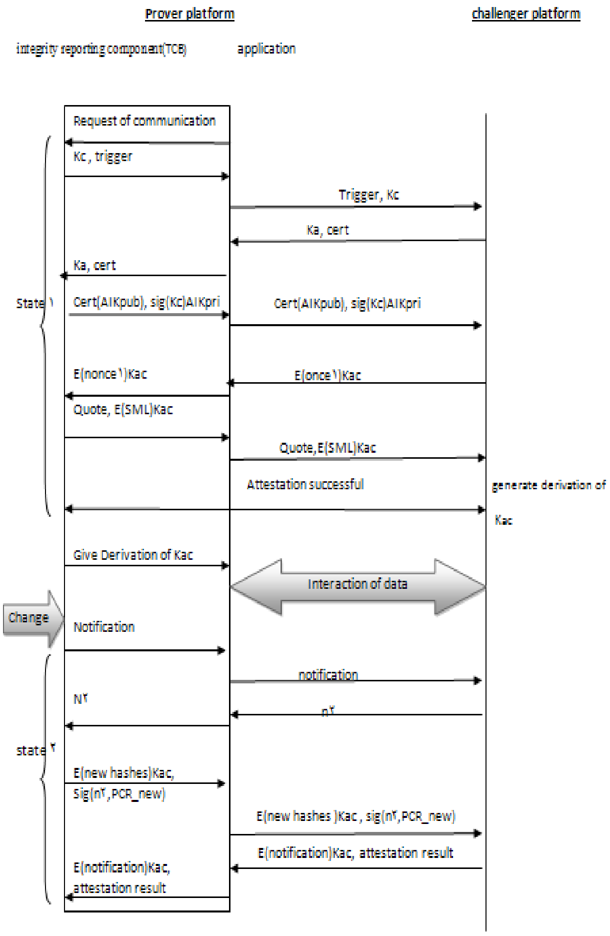

Fig. 6 High level view of proposed solution

6 USING THE PROPOSED SOLUTION TO SOLVE PROBLEMS IN SOME APPLICATION

Nowadays in various interactions between individual nationals and organizations such as interactions of e-banking, TLS/SSL protocol is used. This protocol establishes a secure connection among applications in two platforms and satisfies confidentiality and authenticity attributes. If both or one of the platforms is infected and there is malicious software in them, the malicious software can exploit plaintext data and keys so the secure channel will be useless. Besides, the machines are easier polluted than the channels. Attesting the other platform and verifying its integrity (before interaction) can prevent the misuse of plaintext data and keys. In other words, it is possible to have secure channel between two trusted platforms and extend the channel between applications to trusted platforms.

A customer that wants to do e-banking operations can trust any platform (even in café net) which does his/her electronic transactions. Since the customer is assured that the platform first reports its integrity to a bank service provider before interacting with it can assure that its private information is not compromised. Moreover, this solution can be useful in interaction between various banks and access control management of them.

7 CONCLUSION

In this paper, the proposed method contains the protocol that is robust against masquerading attack along with the measurement and reporting mechanism that measures and reports dynamic changes to challenger and improves this process. The proposed method is effective because it is robust against masquerading attacks, has a reduced number of messages and lesser cost toward related works, prevents useless communication and useless power processing usage of the challenger platform, improves changes reporting during the interaction, and eliminates the concern about symmetric cryptography. The proposed protocols are compared with other works in Table. 1 and the cost column in this table express additional costs toward the TCG protocol.

Table. 1 comparison between the proposed protocols and other works

solution	Number of massages	Cost of operations (basis of the cost is TCG protocol cost)
TCG protocol	2	0
Robust protocol [6]	4	Diffi-Hellman operation + SHA1 operation + symmetric encryption and decryption + random number generation
TLS enhanced protocol [17]	6-8	Cost of TLS protocol
First proposed protocol	4	Asymmetric encryption and decryption + symmetric encryption and decryption + random number generation +SHA1 operation
second proposed protocol	3	Diffi-Hellman operation + sign operation + symmetric encryption and decryption

The result of formal analysis with AVISPA tool for integrity reporting protocol of TCG showed that the protocol is vulnerable against masquerading attacks. A malicious attester can send configuration of another platform as its own configuration and can present itself as safe. We improved the protocol in two ways and removed the possibility of the above attack. To confirm this point, two proposed protocols are analyzed with AVISPA tool. In addition, an integrity measurement and reporting mechanism was proposed that reports the changes in configuration of platform during interaction and removes the concern about symmetric cryptography which is done by CPU. This method can be a solution for problems in security applications such as e-banking applications.

8 REFRENCES

[1] Aarthi Nagarajan, Vijay Varadharajan, Michael Hitchens, Eimear Gallery, 2009, Property Based Attestation and Trusted Computing: Analysis and Challenges, Third International Conference on Network and System Security. *IEEE*.

[2] Ahmad-Reza Sadeghi, 2008, Trusted Computing Special Aspects and Challenges, SOFSEM 2008, LNCS 4910. Springer-Verlag Berlin Heidelberg, 2008; pp. 98–117.

[3] David Safford, Mimi Zohar, A Trusted Linux Client (TLC), T.J. Watson Research Center IBM, Final report.

[4] Elaine Shi, Adrian Perrig, Leendert Van Doorn, 2005, BIND: A Fine-grained Attestation Service for Secure Distributed Systems, *Security and Privacy, IEEE*.

[5] E.Brickell, J.Camenisch, and L.Chen, 2004, Direct Anonymous Attestation, In Proceeding of the 11th ACM conference on computer and communication security, pp. 132-145.

[6] Frederic Stumpf, Omid Tafreschi, Patrick R¨oder, Claudia Eckert, December 2006, A Robust Integrity Reporting Protocol for Remote Attestation, Second Workshop on Advances in Trusted Computing (WATC '06 Fall), Tokyo, Japan.

[7] George Coker, Joshua Guttman, Peter Loscocco, Amy Herzog, June 2011, Principles of Remote Attestation, International Journal of Information Security - Special Issue:10th International Conference on Information and Communications Security, Springer-Verlag Berlin, Heidelberg, 2(10): 63-81.

[8] Reiner Sailer, 2011, Integrity Measurement Architecture (IMA), IBM research.

[9] Jan Camenisch, Better Privacy for Trusted Computing Platforms, Final report, IBM Research. Zurich Research Laboratory, CH-8803 R¨uschlikon, Switzerland.

[10] Liang Gu, Yueqiang Cheng, Xuhua Ding, Robert H. Deng, Yao Guo, Weizhong Shao, 2009, Remote Attestation on Function Execution.

[11] Luca Vigan, 2006, Automated Security Protocol Analysis with the AVISPA Tool, Electronic Notes in Theoretical Computer Science, Elsevier; 155:61-86.

[12] Martin Pirker, Ronald Toegl, Daniel Hein, Peter Danner, 2009 A PrivacyCA for Anonymity and Trust, Trust 2009, LNCS 5471. Springer-Verlag Berlin Heidelberg. pp, 101–119.

[13] Reiner Sailer, Xiaolan Zhang, Trent Jaeger, Leendert van Doorn, August 2004, Design and Implementation of a TCG-based Integrity Measurement Architecture, In 13th USENIX Security Symposium, IBM T. J. Watson Research Center.

[14] Reiner Sailer, Scarsdale, Leendert Peter van Doorn, Xiaolan Zhang, 2011, METHOD AND SYSTEM FOR MEASURING STATUS AND STATE OF REMOTELY EXECUTING PROGRAMS, International Business Machines corporation, assignee, United States Patent: US 7,882, 221 B2.

[15] Shane Balfe, Eimear Gallery, Chris J.Mitchell, Kenneth G. Paterson, 2008, Challenges for Trusted Computing, Final report, Royal Holloway: University of Londo.

[16] SHEN ChangXiang, ZHANG HuanGuo, WANG HuaiMin, WANG Ji, ZHAO Bo, YAN Fei, YU FaJiang, ZHANG LiQiang, XU MingDi, 2010, Research on trusted

computing and its development, Science China Press and Springer-Verlag Berlin Heidelberg, 53: 405–433.

[17] Song Cheng, Liu Bing, Xin Yang, Yang Yixian , Li Zhongxian, Yin Han, 2009, A Security-Enhanced Remote Platform Integrity Attestation Scheme, IEEE.

[18] TCG Group, TPM Main Part 1 Design Principle [Internet], Specification Version 1.2, July 2007, Available from: www.trustedcomputinggroup.org.

[19] Trent Jaeger, Reiner Sailer, Umesh Shankar, 2006, PRIMA: PolicyReduced Integrity Measurement Architecture, SACMAT'06, ACM, Lake Tahoe, California, USA.

[20] Xinwen Zhang, Songqing Chen, Ravi Sandhu, 2005, Enhancing Data Authenticity and Integrity in P2P Systems, George Mason University: IEEE internet computing.

[21] Yacine Gasmi, Ahmad-Reza Sadeghi, Patrick Stewin, Martin Unger, N. Asoka, 2007, Beyond Secure Channels, ACM workshop on Scalable trusted computing.

[26] Yan Jianhong, Peng Xinguang, 2010, Protocol for Dynamic Component-Property Attestation in Trusted Computing, Second International Conference on Networks Security, Wireless Communications and Trusted Computing, IEEE computer society, p.369-372.

[1] Trusted Computing Group
[2] Trusted Platform Module
[3] Platform Configuration Register
[4] Stored Measurement Log
[5] Automated Validation of Internet Security Protocols and Applications
[6] Endorsement key
[7] Attestation identity key
[8] High-Level Protocol Specification Language
[9] High-Level Protocol Specification Language
[10] Trusted Computing Base