Implementation of Remote Attestation using TPM

Dai Yuan
National University of Singapore

Dennis Koh Zhi Quan
National University of Singapore

Li Peiru
National University of Singapore

Ridwan Shariffdeen
National University of Singapore

Abstract

Remote attestation is a method by which a host authenticates it's hardware and software configuration to a remote host. The goal of remote attestation is to enable a remote system to determine the level of trust in the integrity of platform of another system. TPM is a proven method for local attestation to guarantee that the runtime environment is not modified. But when it is extended for security over an insecure network, certain additional context need to be considered else primitive network attacks can compromise the core foundation of trusted computing. Our research provides a proof of concept work starting with a systematic analysis of the inherent weakness of remote attestation and proceeding to provide solutions for the limitations. This work also offers some insights into vulnerabilities of certain prior techniques and provides a promising platform for attaining more advanced security services and guarantees, and a feasibility study in implementing remote attestation for embedded computing devices using TPM.

1. Introduction

Embedded systems have been the main target recently for botnets with the ever-increasing landscape of cybersecurity. Previously, general-purpose computers have been the primary targets, but with the kind of specialized tasks performed by embedded computing systems such as controllers, sensors, actuators and all possible hybrids thereof, the incentive has increased over the past few years. Stuxnet incident is a good demonstration of the impact of compromised embedded systems with an impressive scope [13]. The rapid increase of proliferation of embedded devices with the development of internet of things and growing presence of mobile devices in many aspects of mundane tasks, remote attacks have become a clear and present danger. This motivates the necessity for countermeasures, a number of which have been proposed by the research community and some have been implemented by manufacturers.

Trusted computing is becoming more important especially in critical systems like smart grids, financial transaction processing systems etc. Emerging cybersecurity threats such as advanced persistent threats, malware, social engineering etc, has created numerous security problems to the current computing infrastructure due to its openness and interoperability. Inadequate access control systems have caused devastating attacks even at the hand of the expert users. Attempts have been made to make use of cryptographic infrastructure to a mass-market but with limited success [6]. One of the proposed methods to ensure integrity and authenticity of remote systems is the remote attestation protocol by IBM [15]. Attestation allows a program to authenticate itself and remote attestation is a means for one system to make reliable statements about the software it is running to another system.

Remote attestation provides a security guarantee to authenticate a remote host using its hardware and software configurations. The objective of the process is to enable a system to determine the level of trust in the integrity and authenticity of the remotely connected computing platform. Implementing remote attestation using Trusted Platform Module (TPM) is a fairly new security solution proposed by the Trusted Computing Group (TCG). TPM is an international standard for a secure cryptoprocessor, which is a dedicated hardware designed to ensure the security of hardware platform used by a system by integrating cryptographic methods.

The remote attestation protocol proposed by IBM [14] is vulnerable to several primitive cybersecurity attacks. We evaluate the proposed protocol for its weaknesses and propose a modified version of the protocol that is secure against the select threat models. We implemented our solution on the TPM 2.0 simulator and experimented results shows that it can successfully mitigate replay attacks and man-in-the-middle attacks which were vulnerable to the original protocol.

1.1 Motivation

Attestation is designed to allow the remote host to build a trust foundation upon which future computation can be guaranteed for integrity and authenticity. These implementations may be benign or draconian and users will need to assess them carefully. If not properly implemented in a device will result in a certain loss of control. An owner may even be viewed as an adversary on her own machine.

CONF '14, June 22–27, 2014, Baltimore, PA, USA.
Copyright © 2014 ACM 978-1-nnnn-nnnn-n/yy/mm. . . $15.00.
http://dx.doi.org/10.1145/nnnnnnn.nnnnnnn

1.2 Problem Definition

Using TPM for local attestation is a proven to be secure but when it comes to remote attestation certain cryptographic primitives need to be implemented correctly in order to provide security guarantees, even though the hardware itself can be trusts it needs to make sure remote attestation can be achieved within an insecure network. In our work we implement a proof-of-work that remote attestation can be implemented using TPM which ensures security guarantees.

2. Background

2.1 Trusted Platform Module

A trusted platform module is a computing chip with a cryptoprocessor, secure memory, a compute engine and I/O components, attached to a computing platform. A protected capability on a platform configuration register (PCR) called extend is defined in such a way that the current value constitutes of a trust chain (of events). A trust chain begins with a well-known initial state and comprises the sequence of events up to and including the event that brings a platform to its current state. This sequence is bootstrapped by a hardware root for the trust for measurement (RTM) which can be static or dynamic [9].

2.2 Attestation

Attestation is a mechanism for software to prove its identity. The goal of attestation is to prove to a remote party that your operating system and application software are intact and trustworthy. The verifier trusts that attestation data is accurate because it is signed by a TPM whose key is certified by the CA [4].

2.3 Dynamic Root of Trust

Recently a new mechanism was added to TPM specifications [5] which provides a way to perform attestation dynamically i.e. after boot. Many vendors have welcomed this new mechanism and implemented in their own systems e.g. Intel TXT [2] and AMD SVM [3]. This is a technique that allows a specific CPU instruction to reset the state of some PCRs, isolate memory region, hash and atomically execute its content. Several tamper prevention mechanisms such as disabling DMA and resetting the TPM PCRs are included to prevent fraudulent attestation.

3. Implementation

In our research, we have implemented the whole workflow from quoting authentication, remote attestation to encrypted data communication using TPM. Following shows an implementation of trusted runtime environment with TPM which can remotely communicate with another system (PP). In the workflow, we need to make sure the following:

- transmission between TRE and PP are encrypted by using RSA encryption/decryption with keys generated by TPM
- detect the programs in TRE is not modified by using TPM remote attestation
- ensure TRE is not modified by using TPM local attestation

The whole workflow consists of two parts, TRE booting and TRE & PP data transmission, which shows how TRE is working with PP for data transmission and process acquirement. In our work, we used client-server architecture and Java programming language to fulfill the workflow.

TRE Boot

TRE is first running boot TPM, by showing how TRE is working for booting. After TPM boot, TPM will get the unified EK and generate the AIK with RSA private key and public key. Afterwards, TPM will extends the bytecode of running program to PCR and get the latest quote from PCR for attestation.

Encrypted Communication

Following describes encrypted communication establishment, in short, it contains the following steps;

1. TRE will return one time six-digit token to PP to make sure that one the first one who receives this token can use this username / password to authenticate
2. PP will send back the token together with a processing request to TRE
3. TRE check this token is not used before, if not used, send the public key received from AIK to PP
4. PP uses the received public key to use RSA algorithm to encrypt the username, in our simulating, we always use Mike and send back to TRE with encrypted data
5. TRE received the encrypted data and use the AIK private key to decrypt the username
6. TRE run the program query and return the result together with the generated quote in Step 1
7. PP will run the quote part locally to confirm the TRE is not modified, which is the remote attestation step, and if passed, PP will trust the price value received from TRE.

4. Evaluation

For our evaluation, we used the TPM simulator provided by Microsoft and conducted few security attacks to analyze the behavior of the protocol for the following threat models.

Replay Attack: an attack on a security protocol using a replay of messages from a different context intended to fool the honest participant into thinking they have successfully completed the protocol run [12]. In our context consider a passive attacker listening at the network who can capture valid authentication sequence packets and later replay these packets to disguise as the previous user to the remote system.

Man in the middle Attack: an attack when an adversary secretly alters and relays the communication between two parties. In our context, the communication between two hosts is intercepted by an intermediate user by which he can collect confidential information and forge authentic messages to manipulate the communication.

Denial of Service Attack: an attack where the perpetrator seeks to make the system or network unavailable to its legitimate intended users by temporarily or indefinitely disrupting services of the targeted host. In our context, a denial of service can be executed to attack the trusted computing platform to make it unavailable for a certain time.

In our experiments, we orchestrated a man-in-the-middle attack, replay attack and combination of dos and replay attacks. For this purpose, we used packet capturing tools for sniffing traffic and emulating a man in the middle. We were able to successfully thwart replay attacks and MITM attacks but was not able to neutralize the combination of dos + replay attack due to time constraints.

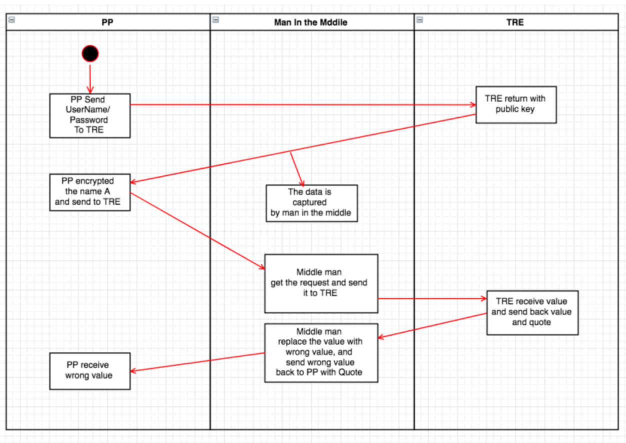
PP
Man In the Mddile
TRE
PP Send UserName/ Password To TRE
TRE return with public key
PP encrypted the name A and send to TRE
The data is captured by man in the middle
Middle man get the request and send it to TRE
TRE receive value and send back value and quote
Middle man replace the value with wrong value, and send wrong value back to PP with Quote
PP receive wrong value

Figure 1. Sequence of the MiTM attack

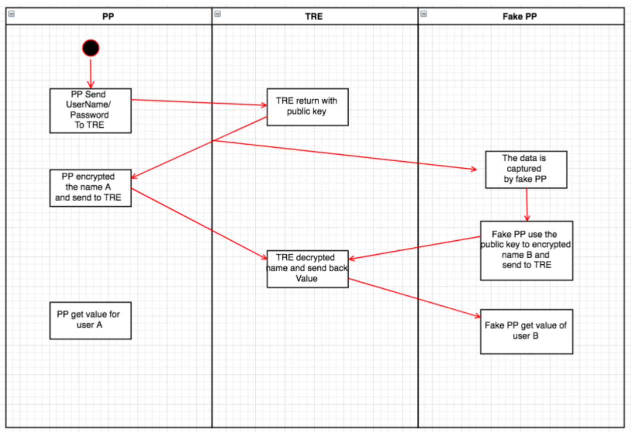
PP
TRE
Fake PP
PP Send UserName/ Password To TRE
TRE return with public key
The data is captured by fake PP
PP encrypted the name A and send to TRE
Fake PP use the public key to encrypted name B and send to TRE
TRE decrypted name and send back Value
PP get value for user A
Fake PP get value of user B

Figure 2. Sequence of the replay attack

Attack Type	Defended
Replay	TRUE
MITM	TRUE
DOS + Replay	FALSE

Table 1. Summary of our evaluation

5. Related Work

5.1 Software based Attestation

There have been several proposals for software-based attestation. One early example is Pioneer [16] that provides device attestation without relying on a secure co-processor or any specialized hardware. It computes a checksum of device memory using a function that includes side-effects (e.g., status registers) in its computation, such that any emulation of this function incurs a timing overhead that is sufficient to detect cheating. Attestation that relies on time-based checksums has also been adapted to embedded devices in [7],[10],[11]. However, some assumptions that form the basis for these solutions have been [17] challenged and several attacks on these (and similar) schemes have been proposed. Moreover, Kovah et al. [8] showed that some time-based attestation schemes may be vulnerable to Time Of Check, Time Of Use(TOCTOU) attack. Alternative (non-time-based) approaches rely on filling the entire memory of the prover with random data to ensure an absence of malicious code. Although the timing is not essential here, this approach is still limited to one-hop attestation since it lacks the means to authenticate a remote prover. In general, all current software-only solutions rely on strong assumptions about the capabilities of the adversary and only work if the verifier communicates directly to the prover, with no intermediate hops. While applicable to specific settings, (e.g., attestation of computer peripherals), this approach is not viable for attestation performed over a network.

5.2 Hardware based Attestation

An early example of this approach is Secure Boot [1]. In it, system integrity is verified at boot time: the root of trust is a small bootloader which computes a hash of the content loaded into memory, and compares this to a signed hash stored in secure ROM. A device is only allowed to boot if the two hashes match. Nowadays, trusted platform modules (TPM) are present in many commercial systems, from phones to laptops. A TPM can compute an integrity checksum over the memory at boot time and send this checksum to be validated by a remote verifier. TPM can also protect a limited amount of data against a compromised operating system, e.g., hide an encryption key unless the Platform Configuration Registers (PCRs) are in a specific state. A TPM can store integrity measurements in PCRs in protected memory. Overall, security is based on two properties: (1) PCRs are accessible only via an API provided by the TPM and (2) measurements in the PCRs can only be extended, each new extension is computed using a cryptographic hash of the previous PCR value and the new measurement. The root of trust is the BIOS that performs the very first extension upon boot. Several concrete architectures have been proposed that rely on a TPM as a foundation [7].

6. Conclusion

In conclusion, in this report, we analyze the remote attestation protocol and successfully built a simulator to analyze the issues of the protocol and vulnerabilities. In our work, we identified that the original protocol was vulnerable to replay attack and man-in-the-middle attack which we presented a formidable solution. However, the modified version is not a complete solution as it doesn't provide protection against more complex attacks such as a combination of DDoS + replay type attacks. In summary, we have done the following:

- We implemented a protocol to achieve remote attestation using TPM which works with authentication and remote attestation
- Empirically evaluated and analyzed replay attacks, man-in-the-middle attacks and denial of service attacks.

References

[1] W. A. Arbaugh, D. J. Farber, and J. M. Smith. A Secure and Reliable Bootstrap Architecture. In *Proceedings of the 1997 IEEE Symposium on Security and Privacy*, SP '97, pages 65–, Washington, DC, USA, 1997. IEEE Computer Society. URL `http://dl.acm.org/citation.cfm?id=882493.884371`.

[2] I. Corporation. Intel Trusted Execution Technology (Intel TXT) – Software Development Guide. 2009.

[3] A. M. Devices. AMD, Secure Virtual Machine Architecture Reference Manual. Technical report, 2005 publication number = 33047.

[4] T. Garfinkel, M. Rosenblum, and D. Boneh. Flexible OS Support and Applications for Trusted Computing. In *Proceedings of the 9th Conference on Hot Topics in Operating Systems - Volume 9*, HOTOS'03, pages 25–25, Berkeley, CA, USA, 2003. USENIX Association. URL `http://dl.acm.org/citation.cfm?id=1251054.1251079`.

[5] T. C. Group. TPM Main Specification. Technical report.

[6] P. Gutmann. PKI: it's not dead, just resting. *Computer*, 35(8):41–49, Aug 2002. ISSN 0018-9162. .

[7] C. Kil, E. C. Sezer, A. M. Azab, P. Ning, and X. Zhang. Remote attestation to dynamic system properties: Towards providing complete system integrity evidence. In *2009 IEEE/IFIP International Conference on Dependable Systems Networks*, pages 115–124, June 2009. .

[8] X. Kovah, C. Kallenberg, C. Weathers, A. Herzog, M. Albin, and J. Butterworth. New Results for Timing-Based Attestation. In *Proceedings of the 2012 IEEE Symposium on Security and Privacy*, SP '12, pages 239–253, Washington, DC, USA, 2012. IEEE Computer Society. ISBN 978-0-7695-4681-0. . URL `http://dx.doi.org/10.1109/SP.2012.45`.

[9] A. Lee-Thorp. Attestation in Trusted Computing: Challenges and Potential Solutions. 04 2018.

[10] Y. Li, J. M. McCune, and A. Perrig. SBAP: Software-Based Attestation for Peripherals. In *Trust and Trustworthy Computing*, 2010.

[11] Y. Li, J. M. McCune, and A. Perrig. VIPER: Verifying the Integrity of PERipherals' Firmware. In *Proceedings of the 18th ACM Conference on Computer and Communications Security*, CCS '11, pages 3–16, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0948-6. . URL `http://doi.acm.org/10.1145/2046707.2046711`.

[12] S. Malladi, J. Alves-Foss, and R. Heckendorn. On Preventing Replay Attacks on Security Protocols. 06 2002.

[13] L. O. M. N. Falliere and E. Chien. W32.Stuxnet Dossier. Symantec,. 2010.

[14] R. Sailer, X. Zhang, T. Jaeger, and L. van Doorn. Design and Implementation of a TCG-based Integrity Measurement Architecture. In *Proceedings of the 13th Conference on USENIX Security Symposium - Volume 13*, SSYM'04, pages 16–16, Berkeley, CA, USA, 2004. USENIX Association. URL `http://dl.acm.org/citation.cfm?id=1251375.1251391`.

[15] R. Sailer, X. Zhang, T. Jaeger, and L. van Doorn. Design and Implementation of a TCG-based Integrity Measurement Architecture. In *Proceedings of the 13th Conference on USENIX Security Symposium - Volume 13*, SSYM'04, pages 16–16, Berkeley, CA, USA, 2004. USENIX Association. URL `http://dl.acm.org/citation.cfm?id=1251375.1251391`.

[16] A. Seshadri, M. Luk, E. Shi, A. Perrig, L. van Doorn, and P. Khosla. Pioneer: Verifying Code Integrity and Enforcing Untampered Code Execution on Legacy Systems. *SIGOPS Oper. Syst. Rev.*, 39(5):1–16, oct 2005. ISSN 0163-5980. . URL `http://doi.acm.org/10.1145/1095809.1095812`.

[17] U. Shankar, M. Chew, and J. D. Tygar. Side Effects Are Not Sufficient to Authenticate Software. In *Proceedings of the 13th Conference on USENIX Security Symposium - Volume 13*, SSYM'04, pages 7–7, Berkeley, CA, USA, 2004. USENIX Association. URL `http://dl.acm.org/citation.cfm?id=1251375.1251382`.