Tsinghua Science and Technology

Volume 21 | Issue 3 Article 7

2016

TPM-Based Remote Attestation for Wireless Sensor Networks

Donglai Fu
the Software School of North University of China, Taiyuan 030051, China.

Xinguang Peng
the School of Computer Science & Technology, Taiyuan University of Technology, Taiyuan 030051, China.

Recommended Citation

Donglai Fu, Xinguang Peng. TPM-Based Remote Attestation for Wireless Sensor Networks. Tsinghua Science and Technology 2016, 21(3): 312-321.

TSINGHUA SCIENCE AND TECHNOLOGY
ISSN 1007-0214 07/11 pp312–321
Volume 21, Number 3, June 2016

TPM-Based Remote Attestation for Wireless Sensor Networks

Donglai Fu* and Xinguang Peng

Abstract: It is essential to design a protocol to allow sensor nodes to attest to their trustworthiness for mission-critical applications based on Wireless Sensor Networks (WSNs). However, it is a challenge to evaluate the trustworthiness without appropriate hardware support. Hence, we present a hardware-based remote attestation protocol to tackle the problem within WSNs. In our design, each sensor node is equipped with a Trusted Platform Module (TPM) which plays the role of a trusted anchor. We start with the formulation of remote attestation and its security. The complete protocol for both single-hop and multi-hop attestations is then demonstrated. Results show the new protocol is effective, efficient, and secure.

Key words: network security; wireless sensor networks; remote attestation; trusted platform module

1 Introduction

Wireless Sensor Networks (WSNs) have been widely applied in critical applications, such as natural disaster detection[1], traffic maintenance[2], and civil infrastructure surveillance[3]. In general, these applications are deployed in unattended or even antagonistic environments. Therefore, security is a prominent issue for these WSNs[4, 5]. Although reputation-based trust management frameworks[6] are a feasible solution to the concern, this approach has shortcomings. For instance, it is difficult to evaluate the trustworthiness of strange nodes without historical data. Consequently, remote attestation[7] is used to solve the problem in the current study. Recently, this measure has been employed to detect malicious nodes in the field of WSNs[8–10]. Generally, these methods can be divided into two categories[11, 12], software-based and hardware-based.

Software-based remote attestation is proposed to verify the integrity of a potentially compromised node without using any hardware devices. The advantage of the technique lies in low cost, because it does not need special hardware, nor physical access to the device. Therefore, the technology can be easily integrated into old devices that are resource-constrained. As simple as this may sound, it is very complicated to correctly design such an attestation scheme in practice[13, 14].

In contrast, hardware-based attestation utilizes additional secure hardware to verify the integrity of a suspicious node. Its merits rely on a special device that provides strong assurance for delivered evidences. Initially the technique was used in traditional computing contexts that are insensitive to cost, but the size, cost, and overhead have to be considered by designers in resource-limited environments. In recent years, researchers deemed that the mechanism was unfeasible for resource-limited environments such as WSNs. But recent developments in hardware technology have made it feasible. Nowadays, the AT97SC* series of devices have been offered with three different interfaces, SPI, LPC, and I2C[15]. Furthermore, all versions are supported in both commercial and industrial grades. It is to be observed that these chips implement version 1.2 of the Trusted Computing Group specifications, and a 2048-bit RSA signature can be processed in 200 ms.

The goal of this study is to design a new approach to verify the integrity of remote nodes distributed in

• Donglai Fu is with the Software School of North University of China, Taiyuan 030051, China. E-mail: hhluci@163.com.
• Xinguang Peng is with the School of Computer Science & Technology, Taiyuan University of Technology, Taiyuan 030051, China.
* To whom correspondence should be addressed.
Manuscript received: 2016-01-08; revised: 2016-03-02; accepted: 2016-03-08

single-hop or multi-hop WSNs. The approach should be effective, efficient, and secure. More to the point, the paper presents a Trusted Platform Module (TPM)-based Security Remote Attestation Protocol (TSRAP), for WSNs. Each sensor node is equipped with one TPM in the protocol. The protocol starts with a challenger sending a challenge to a target. On receiving the challenge, the target constructs a response with the aid of the TPM, and sends it back to the challenger. Finally, the challenger evaluates the trustworthiness of the target according to the response. In the procedure, the TPM ensures the authenticity of the response.

The remainder of the paper is organized as follows. In Section 2, we survey related works in the area of remote attestation. Section 3 describes our network model, and then formulates a TPM-based remote attestation process by a TPM-based Remote Attestation Protocol (TRAP), and defines TSRAP. Assumptions, the threat model, and attack types are presented at the end of the section. In Section 4, the specific TSRAP named TSRAP-I and its security proof are demonstrated. Section 5 analyzes the ability to resist attacks. Performance evaluations are exhibited in Section 6. We summarize the study at the end of the paper.

Relative to the state of the art, the main contributions of this paper are as follows:

(1) The definition of the TPM-based secure remote attestation protocol provides essential prerequisites for the design, usage, or study of such secure-remote-attestation protocols based on TPM.

(2) A new TPM-based secure remote attestation protocol named TSRAP-I is introduced for WSNs. We detailed TSRAP-I, and evaluated its security and performance. Results show our protocol is effective, efficient, and secure.

2 Related Work

2.1 Software-based remote attestation

SWATT[16] is an early software-based remote attestation technique for embedded systems. It allows a trusted external node to verify memory contents of an embedded device without any secure hardware. But the assumption of the clock speed is too rigorous for designers. Shaneck et al.[17] later proposed a software-based remote attestation without requiring additional hardware support for wireless sensor networks. Though the approach is not dependent on the precise measurement of execution timing, it still requires that the challenger receives the response within an expected time. Furthermore, the protocol relies on self-modifying code that is hard to implement, to prevent attackers from tampering with attestation codes. In Ref. [18], the authors introduced a distributed software-based remote attestation protocol to detect compromised nodes. In their scheme, free program memory is filled with random data before deployment to prevent attackers from acquiring enough space to store and run malicious codes. The authors of Ref. [19] presented a software-based attestation procedure that employs a program counter that is not always available to software. Despite the advantage of the cost, some attacks were available to the methods in Refs. [13, 14]. Recently, researchers designed some purely software-based schemes[20–24] to allow a local node to attest to its integrity to a external node. However, these methods do not bind any hardware characteristics to the attestation protocol, and are vulnerable to impersonation attacks.

2.2 Remote attestation based on a trusted platform module

It is commonly agreed that software-based solutions are more vulnerable to attacks than hardware-based solutions, which have been considered unfeasible for the resource-limited environment because of their size, cost, and energy consumption in recent years. However, we assert that the idea changed with the arrival of the TPM (called AT97SC3203S[25]) from the Atmel Corporation.

TPM is a secure chip based on trusted computing specifications[26]. It is a hardware cryptographic module consisting of an execution engine, volatile memory, and non-volatile storage. The engine is designed for SHA-1, RSA key generation, encryption, signing, and random number generation. The endorsement key, EK, and the attestation identity key, AIK, are two important keys for this secure chip. The EK is an asymmetric key pair permanently bound to the platform, and is used in a process for the issuance of AIK credentials and to establish a platform owner. In the old version of TCG specifications, it was generated by the vendor. However, it derives from a common seed in the TPM 2.0 specifications. The AIK is also an asymmetric key pair, and is generated and managed by the TPM with the aid of a privacy certification authority, i.e., a CA. One TPM can have many AIKs. Different AIKs are used to protect privacy when the platform owner is concerned about the consequences of collusion. In this study, we assume that

each TPM owns an AIK, and the key does not change during the attestation process—although the situation may be different in practice. Such an assumption is not critical, because the generation of the AIK is not our main concern. In addition, we point out that the chip has a set of special registers, Platform Configuration Registers (PCRs). These PCRs can be classified into two groups, static and dynamic, according to their initial value and the time that they can be reset. Static PCRs, PCR 0-16, are reset to 0 on system reboot. Dynamic PCRs, PCR 17-23, are initialized as –1 and 0 at reboot and run-time, respectively. In addition, both can only be updated through the *extend* function, which aggregates the current content of a PCR with new content, hashes them, and stores the result back in the PCR. This promising technique can provide two important services, namely, secure storage and platform attestation. In recent years, we have carried out some research[27–31] into platform attestation and its application for a general computing environment with a plenty of resources.

In the area of WSNs, these advantages of TPM were considered for key establishment, distribution, and management in Ref. [32]. Their work shows the TPM can dramatically enhance WSNs' security. In Ref. [33], two TPM-based attestation protocols were constructed to detect compromised nodes in WSNs. In these two protocols, only cluster heads can attest to their integrity to other cluster nodes because only these sensor nodes are equipped with a TPM chip. Moreover, the scheme relies on a trusted system state that never changes. In Refs. [8, 15], authors proposed the first platform with supporting RSA-based functions with the help of the TPM chip. Their contributions show that a trusted sensor node with low energy consumption, low cost, and small size is available. Tan et al.[34] implemented a remote attestation protocol to detect unauthorized tampering with application codes by the aid of the TPM chip. In the implementation, each sensor node is equipped with a TPM chip. Wagner et al.[35] presented a TPM-based code-update protocol that enables one node with the TPM chip to prove its trustworthiness to another node. In Ref. [36], the authors reviewed various types of physical attacks on WSNs and some trusted wireless sensor nodes. Their contributions show that further research on TPM-based attestation is essential to provide enough security for more challenging applications of WSNs.

3 Problem Statements

3.1 Network model

As shown in Fig. 1, WSNs consist of hundreds or thousands of low-cost nodes that have a fixed location or are randomly deployed to gather critical data. The flow of data ends at a special node called a base station or a sink. The sink is used to link one sensor network to another network. In general, these sinks have enough memory, storage, power, and bandwidth to communicate with other sinks, but sensor nodes communicate to the nearest sink in a multi-hop manner because of their limited power, bandwidth, and radio range. The owner of a WSN may set some special nodes that collect readings from surrounding nodes, and forward a single message called an aggregate value to the next node, to save energy.

Sensor nodes are easily compromised by adversaries because they are generally deployed in open and even hostile environments. Therefore, the sensor node must verify the integrity of cooperators before transactions. Remote attestation is an alternative approach.

3.2 TSRAP

TPM-based remote attestation is a process by which a platform can attest to its trustworthiness, i.e., integrity. In the process, the appraiser is called a challenger, and the attesting entity, the platform, is termed a target. Furthermore, a TPM represents a trusted entity which cannot be damaged by malicious software, and can vouch for the accuracy of its messages. Thus, TPM-based attestation is a powerful tool for assessing the trustworthiness of a target.

The attestation process is triggered by the challenger according to TCG specifications. It first sends a

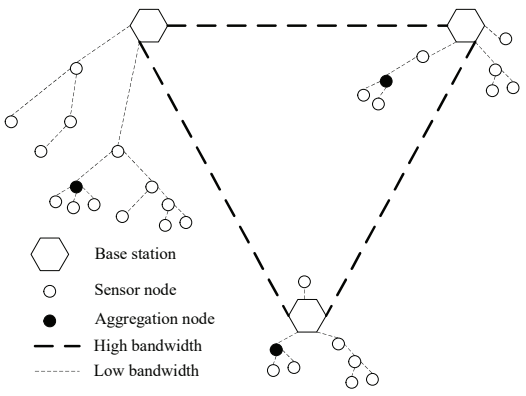

Fig. 1 Wireless sensor network architecture.

challenge to the target. Upon receiving the challenge, the target submits complete platform-configuration data called "the response to the challenger". Finally, the challenger evaluates the trustworthiness of the target by comparing the received Store Measurement Log (SML) and PCR values with reference values.

The goal of the attestation is to allow a benign target to generate an authentication token that convinces the challenger that it is in an expected state. The key issue is that the token must reflect the target's real state. To discuss the problem, we first define the attesting process as a protocol noted as TRAP (see Definition 1), then describe the definition of the TPM-based secure remote attestation protocol, TSRAP, by a secure experiment.

Definition 1 TRAP The TPM-based remote attestation protocol is a triple (Req, Res, Ver) consisting of polynomial-time algorithms:

(1) $c \longleftarrow \mathrm{Req}(n)$

The challenge-generation algorithm Req takes as input a random n, and outputs the challenge c. We write this algorithm as $c \longleftarrow \mathrm{Req}(n)$, since Req may be randomized.

(2) $r \longleftarrow \mathrm{Res}(s, c)$

The response-generation algorithm Res takes as input the target's state s and the challenge c, and outputs the response r.

(3) $\tau := \mathrm{Ver}(r)$

The verification-algorithm Ver takes as input the received response r, and outputs an authentication token $\tau \in \{0, 1\}$. $\tau = 1$ iff the target's state s corresponds to the expected value; else $\tau = 0$. We write this algorithm as $\tau := \mathrm{Ver}(r)$ since Ver is deterministic.

As shown in Definition 2, the experiment $\mathrm{Exp}_{\Pi}^{\Lambda}$ is defined for any remote attestation protocol $\Pi = (\mathrm{Req}, \mathrm{Res}, \mathrm{Ver})$ and any adversary Λ.

Definition 2 $\mathrm{Exp}_{\Pi}^{\Lambda}$ The adversary Λ submits one state $s' \neq s$ or one challenge $c' \neq c$, and accesses $r \longleftarrow \mathrm{Res}(s, c)$. Eventually, it outputs a response r'. The output of the experiment is defined to be 1 if $r' = r$, and 0 otherwise. We write $\mathrm{Exp}_{\Pi}^{\Lambda} = 1$ if the output is 1, and in this case we say that adversary Λ succeeds.

As shown in Definition 3, we define a secure remote attestation protocol named TSRAP, based on Definition 2, to formulate the TPM-based secure remote attestation process.

Definition 3 TSRAP We say a TRAP is secure if there exists a negligible function negl for any polynomial-time adversary Λ and a sufficiently large n such that $\Pr[\mathrm{Exp}_{\Pi}^{\Lambda}(n) = 1] \leqslant \mathrm{negl}(n)$.

3.3 Assumptions

Though TSRAP is valid for general TPM-based remote attestation scenarios, we are mainly concerned with WSNs in this study. Therefore, we make the following assumptions. Firstly, sensor nodes including sinks are equipped with a TPM chip that cannot be compromised by attackers. Furthermore, sinks can be broken by aggressors, and they own unlimited resources, in contrast to ordinary sensor nodes. Secondly, links are insecure among sensor nodes because wireless links are easily accessed by hackers. Finally, each TPM chip has possessed an endorsement key and an attestation identity key, as well as a sealing key.

3.4 Threat model and attack types

In this study, adversaries have the following abilities: (1) Adversaries can eavesdrop on, copy, and replay messages transmitted on channels. (2) Adversaries can either intercept legal messages or inject forged messages. (3) A small number of malicious sensor nodes can be deployed with the same hardware capabilities as benign sensor nodes. (4) Their physical architecture cannot be modified. Moreover, attackers cannot control a large number of sensor nodes because this requires them to be physically present in the deployment region with hardware for a long time. However, to install malicious codes to sensor nodes is possible over the air. (5) Cryptography primitives cannot be broken. In other words, thieves cannot retrieve messages without knowing keys.

As before, the function $r \longleftarrow \mathrm{Res}(s, c)$ must be such that (1) only the function can compute a valid response r; (2) r must accurately capture the target's state s. That is to say, $\mathrm{Res}(s', c') = \mathrm{Res}(s, c)$ is negligible for any $s' \neq s$ or $c' \neq c$. Two types of attack may be launched by adversaries based on the above description. One is that adversaries simulate $\mathrm{Res}(s, c)$, and correctly compute its output r. Another is that r cannot correctly reflect (s, c). Stated briefly, adversaries escape the detection of TSRAP.

4 TSRAP-I

4.1 Notations

Before describing the specific TSRAP called TSRAP-I, notations used in the rest of the paper are illustrated below.

B_i: i-th base station;
N_i: i-th sensor node;
$K_{N_i B_i}$: session key between N_i and B_i;
$(\mathrm{AIK}_i^{\mathrm{pk}}, \mathrm{AIK}_i^{\mathrm{sk}})$: attestation identity key pairs of TPM of the sensor node N_i;
$(S_i^{\mathrm{pk}}, S_i^{\mathrm{sk}})$: sealing key pairs of the sensor node N_i for sealing sensitive data;
C_{b}: content of the bootloader program;
C_{p}: content of the application program;
V_{p}: referenced hash value of the application;
$\|$: concatenation operation;
PCR_i: i-th PCR;
HMAC: hashed message authentication code function;
h: SHA-1 hash function.

4.2 Description of TSRAP-I

TSRAP-I encompasses three phases: an initial phase, a bootloader phase, and an attestation phase.

Initial phase: This phase is invoked before deployment. Therefore, we suppose that sensor nodes are correct in this phase. A node N_i executes the following operations.

(1) N_i sends the command *TPM_Loadkey* to the TPM to load the sealing key $(S_i^{\mathrm{pk}}, S_i^{\mathrm{sk}})$.

(2) N_i sends the command *TPM_SHA1Start* and *TPM_SHA1Complete* to the TPM to compute the hash of C_{b}, noted as h_{b}.

(3) N_i sends the command *TPM_Extend* to the TPM to extend h_{b} to PCR_1.

(4) N_i sends the command *TPM_Seal* to the TPM to seal $K_{N_i B_i}$ and PCR_1 together.

Bootloader phase: This phase occurs before listening to the code-update request and after the initialization of the bootloader. Steps (2) and (3) that ran in the initial phase must be executed again in the bootloader phase, or $K_{N_i B_i}$ cannot be unsealed. During this phase, a node N_i completes the following tasks.

(1) N_i sends the command *TPM_SHA1Start* and *TPM_SHA1Complete* to the TPM to compute the hash of C_{b}, noted as h_{b}.

(2) N_i sends the command *TPM_Extend* to the TPM to extend h_{b} to PCR_1.

(3) N_i sends the command *TPM_SHA1Start* and *TPM_SHA1Complete* to the TPM to compute the hash of $K_{N_i B_i} \| C_{\mathrm{p}}$, noted as h_{p}.

(4) N_i sends the command *TPM_Extend* to the TPM to extend h_{p} to PCR_2.

Attestation phase: This phase occurs whenever the challenger N_1 needs to evaluate the target N_2. In the following processes, N_1 confirms that N_2 has been compromised if the attestation token $\tau = 0$. The interactions among participants can be seen in Fig. 2.

N_1 performs the following steps to evaluate N_2.

(1) N_1 sends the command *TPM_Unseal* to its TPM to unseal $K_{N_1 B_1}$; if the operation fails, the attestation process stops.

(2) N_1 sends the command *TPM_GetRandom* to its TPM to get a random R_1.

(3) N_1 sends $\{N_1, B_1, N_2, R_1, \mathrm{HMAC}(K_{N_1 B_1}, N_1, B_1, N_2, R_1)\}$ to the base station B_1, and waits for its response. If no response is received in time or the response is incorrect, the session will be stopped.

After B_1 receives the above message from N_1, B_1 performs the following steps.

(1) Verify the integrity of the message and the identity of the sender by computing $\mathrm{HMAC}(K_{N_1 B_1}, N_1, B_1, N_2, R_1)$. If the check is not passed, it will inform N_1 that the request is invalid, and stop the session.

(2) B_1 sends the message $\{B_1, N_2, R_1, \mathrm{HMAC}(K_{N_2 B_1}, B_1, N_2, R_1)\}$ to N_2.

On receiving the above message from B_1, N_2 performs the following steps.

(1) Verify the integrity of the message and the identity of the sender by computing HMAC $(K_{N_2 B_1}, B_1, N_2, R_1)$ again. If the check is not passed, it will inform B_1 that the request is invalid, and stop the session.

(2) N_2 sends the command *TPM_Unseal* to its TPM to unseal $K_{N_2 B_1}$. If the operation fails, it will inform B_1 that the response fails, and stop the session.

(3) N_2 sends the command $TPM\_Quote(R_1, \mathrm{PCR}_2)$ to its TPM to get a random R_2 and the cryptographic reporting of PCR_2 called Quote.

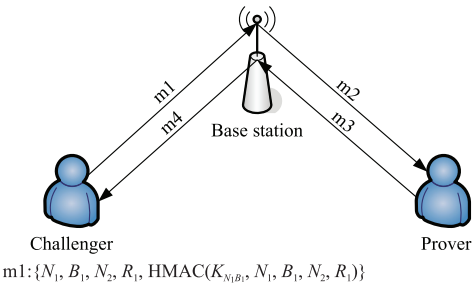

Fig. 2 Interaction diagram of challenger, target, and base station.

(4) N_2 sends the message $\{N_2, R_1, R_2, \mathrm{PCR}_2, \mathrm{Quote}, \mathrm{HMAC}(K_{N_2B_1}, N_2, R_1, R_2, \mathrm{PCR}_2, \mathrm{Quote})\}$ to B_1.

(5) Remove $K_{N_2B_1}$.

Upon receiving the above message from N_2, B_1 performs the following steps.

(1) Verify the integrity of the message and the identity of the sender by computing $\mathrm{HMAC}(K_{N_2B_1}, N_2, R_1, R_2, \mathrm{PCR}_2, \mathrm{Quote})$.

(2) Verify Quote using $\mathrm{AIK}_2^{\mathrm{pk}}$.

If any check is not passed, it will inform N_1 that the response is invalid, and stop the session.

(3) $\tau = 1$ only if $\mathrm{PCR}_2 = \mathrm{Quote} = V_p$, or $\tau = 0$.

(4) B_1 sends the message $\{B_1, N_2, R_1, R_2, \tau, \mathrm{HMAC}(K_{N_1B_1}, B_1, N_2, R_1, R_2, \tau)\}$ to N_1.

On receiving the above message, N_1 performs the following steps.

(1) Verify the integrity of the message and the identity of the sender by computing $\mathrm{HMAC}(K_{N_1B_1}, B_1, N_2, R_1, R_2, \tau)$. If the check is not passed, N_1 deems that the response is invalid, and stops the session.

(2) It checks the attestation token τ, and makes a decision regarding the trustworthiness of N_2.

(3) Remove K_{N_1,B_1}.

4.3 Security proof

Theorem 1 TSRAP-I is a TPM-based secure remote attestation protocol with respect to TSRAP.

Proof

As discussed before, if an adversary constructs a response $r' = r$ by taking as inputs $s' \neq s$ or $c' \neq c$, we say the adversary succeeds, or fails.

In the above construction, $\{N_1, B_1, N_2, R_1, \mathrm{HMAC}(K_{N_1B_1}, N_1, B_1, N_2, R_1)\}$, $(\mathrm{PCR}_2, \mathrm{Quote})$, and $\{N_2, R_1, R_2, \mathrm{PCR}_2, \mathrm{Quote}, \mathrm{HMAC}(K_{N_2B_1}, N_2, R_1, R_2, \mathrm{PCR}_2, \mathrm{Quote})\}$ correspond to c, s, and r, respectively.

Now, we consider two cases.

(1) $c' \neq c$. In fact, c' can only be a stale c since the adversary cannot generate a fresh c without $K_{N_1B_1}$. Yet, it is difficult to extract it from a sensor node without breaking its TPM. Thus, the response r' may be only an old message, which will be detected by B_1. In other words, $r' \neq r$ does not hold.

(2) $s' \neq s$. In the case, although an attacker can replace PCR_2 with $h(C_{\mathrm{p}}')$, Quote cannot be tampered with. Therefore, the attacker cannot construct a fresh $s' = s$. Thus, $r' = r$ also does not hold.

As a consequence, TSRAP-I is secure with respect to TSRAP. ■

5 Security Discussion

5.1 Time-Of-Check-To-Time-Of-Use (TOCTTOU) attacks

TOCTTOU is a popular attack against remote attestation. The concrete process is as follows. The attacker generates a correct attestation response from correct codes when the time-of-check case happens, but it invokes malicious codes when the time-of-use case emerges. There are three kinds of scenarios to be considered in the attack.

The first is that benign and malicious codes swap places. On being attested, the former is used to generate a valid response. When being executed, the latter is invoked to do evil. In our scheme, evildoers have two approaches to launch the attack. One is that they generate a fresh PCR_2 rather than read the value from PCR registers. In this case, the generated response does not contain a correct cryptographic reporting of PCR_2 called Quote because the command *TPM_Quote* only receives the index of the PCR instead of its content. Furthermore, its content is retrieved within the TPM. Therefore, the case will be detected by the base station. Another is that intruders inject malware over the air. The malware may next attempt to write an expected value. However, such an operation is doomed to failure without knowing the session key $K_{N_1B_1}$.

A second case is that correct codes are placed in the correct location, but malicious codes are invoked on being executed. In this case, attackers also have two approaches to launch the attack. One is to inject a jump instruction to malicious codes into the correct application. Another is that attackers construct malicious codes by invoking existing routines in a different order[37]. It should be stressed that the attack must exploit programs to process received broadcast packets. Thus, the attack can be detected only by integrating packet-processing codes into the application software.

The last case is that both are not in the right location, but malicious codes are executed, and correct codes are attested. Obviously, this case is a combination of the above two cases. Therefore, it can be resisted as long as either of them can be found.

5.2 Rootkit-based attacks

Rootkit-based attacks are as follows. On receiving the challenge, the program-memory hook first copies itself to the free data memory, then makes the return address

of the attestation process point to the data-memory hook, and moves malicious codes from the program memory to the data memory. After the attestation is completed, the data-memory hook restores the program-memory hook again.

In our scheme, the above attack can not be mounted. There are two reasons for this. On the one hand, if the removal occurs in the attestation stage, the installation of the new application will result in the execution of the bootloader. As a result, PCR_2 is rewritten. On the other hand, if the removal happens in the bootloader stage, the bootloader's codes need to be modified. However, this will lead to $K_{N_i B_i}$ not being able to be unsealed. Therefore, adversaries are also doomed to failure in this case.

5.3 MITM attacks

Attacks against communication protocols need to be considered in our scheme since we assume that adversaries have complete power over communication channels. Man-In-The-Middle (MITM) is a popular attack against communication protocols. There are three cases to be considered in TSRAP-I.

The first is that the attacker impersonates a middleman between the challenger and the base station. The second is that the attacker plays the role of the middleman between the target and the base station. The third approach is one in which it simulates a base station between the challenger and the target. In the first case, messages on the channel include $\{N_1, B_1, N_2, R_1, \mathrm{HMAC}(K_{N_1B_1}, N_1, B_1, N_2, R_1)\}$ and $\{B_1, N_2, R_1, R_2, \tau, \mathrm{HMAC}(K_{N_1B_1}, B_1, N_2, R_1, R_2, \tau)\}$. In the second case, messages are $\{B_1, N_2, R_1, \mathrm{HMAC}(K_{N_2B_1}, B_1, N_2, R_1)\}$ and $\{N_2, R_1, R_2, \mathrm{PCR}_2, \mathrm{Quote}, \mathrm{HMAC}(K_{N_2B_1}, N_2, R_1, R_2, \mathrm{PCR}_2, \mathrm{Quote})\}$. In the last case, the channel includes all the aforementioned messages.

Clearly, this attack cannot be mounted. There are three reasons for this. The first is that the attacker cannot tamper with messages because the hashed message-authentication code is always included in each message. The second is that it cannot replay outdated messages because of the appearance of random numbers. Finally, it cannot construct legal messages without knowing the shared key. Therefore, the only thing the attacker as a middleman can do is to steal sensitive information. Yet, in TSRAP-I, nothing is kept secret, except for keys, which are not shown on channels.

6 Performance Evaluation

6.1 Storage requirements

Protocols should be kept to a low storage overhead for WSNs because of their limited memory. In this section, we do not analyze the storage overhead of the base station since we assume that it has rich memory resources. As shown in Table 1, sensor nodes must store the storage root key, the attestation identity key, the sealing key, and the session key. The first three keys are 2048-bit RSA keys according to TCG specifications. The last key is a 56-bit symmetric key. Therefore, the storage requirements of the keys are 6200 bits= 775 bytes. Though the overhead seems to be unsuitable for a sensor node with a 4 KB EEPROM, in practice, the TPM chip called AT97SC3203S[25] provides designers with an internal EEPROM, which can store multiple RSA keys. Furthermore, it can also generate 1024-bit RSA keys. What's more, the shared key is stored inside the chip. Accordingly, existing products can satisfy current storage requirements.

Besides the above keys, a sensor node should save its identifier, the identifier of the base station, and cooperators' identifiers. Let L_N denote the length of the identifier, and a cooperator's number is denoted ω. Thus, the storage requirements for identifiers are $L_N \cdot (\omega + 1)$.

6.2 Energy consumption

In this section, we only discuss the challenger's and the target's energy usage after deployment. For our protocol, the energy usage is mainly attributed to three cases: execution of TPM commands, transmission and reception of messages, and HMAC operation. In the protocol, TPM commands encompass *TPM_SHA1Start*, *TPM_SHA1Complete*, *TPM_Extend*, *TPM_Unseal*, *TPM_GetRandom*, and *TPM_Quote*. We denote their energy by $e_{t1}, e_{t2}, e_{t3}, e_{t4}, e_{t5}$, and e_{t6}, respectively. Let e_{HMAC} denote the energy of the HMAC operation. We use e_{1r} and e_{1s} to represent the energy usage for sending and receiving one byte respectively. We write

Table 1 Storage requirements for keys.

Key	Number	Length (bit)
Storage root key	1	2048
Identity key	1	2048
Sealing Key	1	2048
Shared key	1	56
Total	4	6200

the lengths of the identifier, the nonce, the PCR value, the variable Quote, an HMAC, and authentication as l_1, l_2, l_3, l_4, l_5, and l_6, respectively.

As shown in Table 2, the energy usage of operations is $e_{\text{co}} = 2 \times (e_{t1} + e_{t2} + e_{t3}) + e_{t4} + e_{t5} + 2 \times e_{\text{HMAC}}$ for the challenger. Moreover, the challenger sends the message $\{N_1,\ B_1,\ N_2,\ R_1, \text{HMAC}(K_{N_1B_1}, N_1, B_1, N_2, R_1)\}$, and receives the message $\{B_1, N_2, R_1, R_2, \tau, \text{HMAC}(K_{N_1B_1}, B_1, N_2, R_1, R_2, \tau)\}$ during the attestation stage. Therefore, the energy usage is $e_{\text{cs}} = e_{1s} \times (3 \times l_1 + l_2 + l_5)$ for sending, and the energy usage of the reception is $e_{\text{cr}} = e_{1r} \times (2 \times l_1 + 2 \times l_2 + l_5 + l_6)$. Thus, the total energy usage is $e_{\text{ct}} = e_{\text{co}} + e_{\text{cs}} + e_{\text{cr}}$.

In contrast, the energy usage of operations is $e_{\text{po}} = 2 \times (e_{t1} + e_{t2} + e_{t3}) + e_{t4} + e_{t6} + 2 \times e_{\text{HMAC}}$ for the target, according to Table 2. What's more, the target sends one message $\{N_2, R_1, R_2, \text{PCR}_2, \text{Quote}, \text{HMAC}(K_{N_2B_1}, N_2, R_1, R_2, \text{PCR}_2, \text{Quote})\}$, and receives one message $\{B_1, N_2, R_1, \text{HMAC}(K_{N_2B_1}, B_1, N_2, R_1)\}$ during the attestation stage. Therefore, the energy usage is $e_{\text{ps}} = e_{1s} \times (l_1 + 2 \times l_2 + l_3 + l_4 + l_5)$ for sending one byte, and the energy usage for receiving one byte is $e_{\text{pr}} = e_{1r} \times (2 \times l_1 + l_2 + l_5)$. Thus, the total energy usage is $e_{\text{pt}} = e_{\text{po}} + e_{\text{ps}} + e_{\text{pr}}$.

In Ref. [33], the authors demonstrated that the cost of two operations, unsealing and signature, is high, but other operations' cost is negligible. In our protocol, only the unsealing operation is used once for the challenger. In contrast, the two operations are used once for the target. Hence, the target's cost is higher than the challenger's cost in energy usage. The cost is a whisker away from 210 mJ, which is less than the energy usage 221 mJ present in Ref. [33].

6.3 Performance optimization

In the above protocol, the target carries extra poundage. However, it is unnecessary to require the target to generate a fresh response every time the challenge happens. More to the point, it is to allow the base station both to save the response to its cache and to update it in regular intervals. There are three reasons for this. The first is that the base station is creditable. The second reason is that it owns rich resources. Finally, the measure can accelerate the attestation process, and reduce energy consumption. The effect will be more prominent when multiple challenges simultaneously happen to validate the integrity of the same remote node.

Table 2 Operations of challenger labeled 'C' and operations of target labeled 'T'.

Stage	Operation	C	T
Bootloader	*TPM_SH1Start*	2	2
	TPM_SHA1Complete	2	2
	TPM_Extend	2	2
Attestation	*TPM_Unseal*	1	1
	TPM_GetRandom	1	0
	TPM_Quote	0	1
	HMAC	2	2

7 Conclusion

In this paper, we present a new TPM-based remote attestation protocol named TSRAP-I for WSNs. Each sensor node is equipped with a TPM chip which acts as a trust anchor. We started with some formal definitions for TPM-based secure remote attestation. Afterwards, we elaborated on TSRAP-I. In the protocol, the challenger evaluated the target's trustworthiness with the help of the trusted base station and the secure chip called TPM. In theory, we confirmed that the security of TSRAP-I conformed to the definition of TSRAP. Finally, we discussed three kinds of attacks against the new protocol. The storage cost and energy consumption were also examined. We also proposed optimization measures.

Acknowledgment

The research leading to these results was supported by the outstanding graduate student innovation project of Shanxi Province (No. 20123030). Deep gratitude is extended to W. Zhen, W. Ying, and B. Jing for their beneficial discussions and comments. We are also grateful to the anonymous reviewers for their comments and suggestions.

References

[1] A. Zambrano, I. Perez, C. Palau, and M. Esteve, Quake detection system using smartphone-based wireless sensor network for early warning, in *Proc. IEEE International Conference on Pervasive Computing and Communications*, Budapest, Hungary, 2014, pp. 297–302.

[2] J. Yu, J. Yang, and H. Wang, Fault detection for large-scale railway maintenance equipment base on wireless sensor networks, *Sensors & Transducers*, vol. 169, no. 4, pp. 165–169, 2014.

[3] G. Hackmann, W. Guo, G. Yan, Z. Sun, C. Lu, and S. Dyke, Cyber-physical code sign of distributed structural health monitoring with wireless sensor networks, *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 1, pp. 63–72, 2014.

[4] A. Liu, M. Kim, L. B. Oliveira, and H. Tan, Wireless sensor network security, *International Journal of Distributed Sensor Networks*, 2013. doi:10.1155/2013/362385.

[5] Y. Liu and W. Trappe, Topology adaptation for robust ad hoc cyberphysical networks under puncture-style attacks, *Tsinghua Science and Technology*, vol. 20, no. 4, pp. 364–375, 2015.

[6] B. Zhang, Z. Huang, and Y. Xiang, A novel multiple-level trust management framework for wireless sensor networks, *Computer Networks*, vol. 72, no. 7, pp. 45–61, 2014.

[7] G. Coker, J. Guttman, P. Loscocco, A. Herzog, J. Millen, B. O'Hanlon, J. Ramsdell, A. Segall, J. Sheehy, and B. Sniffen, Principles of remote attestation, *International Journal of Information Security*, vol. 10, no. 2, pp. 63–81, 2011.

[8] W. Hu, H. Tan, P. Corke, W. C. Shih, and S. Jha, Toward trusted wireless sensor networks, *ACM Transactions on Sensor Networks (TOSN)*, vol. 7, no. 1, pp. 1–25, 2010.

[9] H. Khiabani, N. B. Idris, and J. L. Ab Manan, Leveraging remote attestation to enhance the unified trust model for WSNs, in *Proc. IEEE International Conference on Cyber Security, Cyber Warfare and Digital Forensic*, Kuala Lumpur, Malaysia, 2012, pp. 139–143.

[10] S. Kiyomoto and Y. Miyake, Lightweight attestation scheme for wireless sensor network, *International Journal of Security & Its Applications*, vol. 8, no. 2, pp. 25–40, 2014.

[11] L. Li, H. Hu, J. Sun, Y. Liu, and J. S. Dong, Practical analysis framework for software-based attestation scheme, in *Formal Methods and Software Engineering*. Springer International Publishing, 2014, pp. 284–299.

[12] J. Valente, C. Barreto, and A. A. Crdenas, Cyber-physical systems attestation, in *Proc. IEEE International Conference on Distributed Computing in Sensor Systems(DCOSS)*, Marina Del Rey, CA, USA, 2014, pp. 354–357.

[13] C. Castelluccia, A. Francillon, D. Perito, and C. Soriente, On the difficulty of software-based attestation of embedded devices, in *Proc. of the 16th ACM Conference on Computer and Communications Security*, Chicago, IL, USA, 2009, pp. 400–409.

[14] A. Francillon, C. Castelluccia, D. Perito, and C. Soriente, Comments on "refutation of *on the difficulty of software-based attestation of embedded devices*", http://www.inrialpes.fr/planete/people/c-castel/, 2015.

[15] Infineon, AT97SC*, http://www.atmel.com/products/security-ics/embedded/default.aspx, 2015.

[16] A. Seshadri, A. Perrig, L. van Doorn, and P. Khosla, SWATT: Software-based attestation for embedded devices, in *Proc. IEEE Symposium on Security and Privacy*, Oakland, CA, USA, 2004, pp. 272–282.

[17] M. Shaneck, K. Mahadevan, V. Kher, and Y. Kim, Remote software-based attestation for wireless sensors, in *Security and Privacy in Ad-hoc and Sensor Networks*. Springer Berlin Heidelberg, 2005, pp. 27–41.

[18] Y. Yang, X. Wang, S. Zhu, and G. Cao, Distributed software based attestation for node compromise detection in sensor networks, in *Proc. 26th IEEE International Symposium on Reliable Distributed Systems*, Beijing, China, 2007, pp. 219–230.

[19] A. Seshadri, M. Luk, A. Perrig, L. van Doorn, and P. Khosla, SCUBA: Secure code update by attestation in sensor networks, in *Proc. of the 5th ACM workshop on Wireless Security*, Los Angeles, CA, USA, 2006, pp. 85–94.

[20] Y. Li, J. M. McCune, and A. Perrig, SBAP: Software based attestation for peripherals, in *Trust and Trustworthy Computing*. Springer Berlin Heidelberg, 2010, pp. 16–29.

[21] T. AbuHmed, J. Kang Jeonil, D. Nyang, and K. Lee, A software-based group attestation for wireless sensor networks, *Ad Hoc & Sensor Wireless Networks*, vol. 13, nos. 1&2, pp. 121–154, 2011.

[22] X. Kovah, C. Kallenberg, C. Weathers, and A. Herzog, New results for timing-based attestation, in *Proc. IEEE Symposium on Security and Privacy (SP)*, San Francisco, CA, 2012, pp. 239–253.

[23] J. W. Ho, Robust detection of malicious nodes in mobile sensor networks using software attestation, *International Journal of Distributed Sensor Networks*, 2013, doi:10.1155/2013/839523.

[24] J. Horsch, S. Wessel, F. Stumpf, and C. Eckert, SobrTrA: A software-based trust anchor for ARM cortex application processors, in *Proc. of the 4th ACM Conference on Data and Application Security and Privacy*, San Antonio, TX, USA, 2014, pp. 273–280.

[25] W. Hu, P. Corke, W. C. Shih, and L. Overs, secfleck: A public key technology platform for wireless sensor networks, in *Wireless Sensor Networks*. Springer Berlin Heidelberg, 2009, pp. 296–311.

[26] Trusted Computing Group, Trusted computing specification, http://www.trustedcomputinggroup.org, 2015.

[27] D. L. Fu and X. G. Peng, Improved remote attestation mechanism of platform configuration based on chameleon hashes, (in Chinese), *Computer Science*, vol. 40, no. 1, pp. 118–121, 2013.

[28] D. L. Fu and G. X. Chen, Remote attestation using chameleon hash and dynamic Huffman Merkle Hash tree, *Journal of Computational Information Systems*, vol. 8, no. 17, pp. 7103–7112, 2012.

[29] D. L. Fu and G. X. Chen, Improved remote attestation mechanism based on group signatures and unbalanced Merkle hash tree, *Journal of Information and Computational Science*, vol. 10, no. 3, pp. 773–781, 2013.

[30] D. L. Fu, X. G. Peng, G. X. Chen, and Q. X. Yang, Remote attestation mechanism of platform configuration based on dynamic Huffman tree, (in Chinese), *Journal of Computer Applications*, vol. 32, no. 8, pp. 2275–2279, 2012.

[31] D. L. Fu, X. G. Peng, and Y. L. Yang, Trusted platform module-based scheme for secure access to outsourced data, (in Chinese), *Journal of Electronics and Information Technology*, vol. 35, no. 7, pp. 1766–1773, 2013.

[32] S. Ganeriwal, S. Ravi, and A. Raghunathan, Trusted platform based key establishment & management for sensor networks, http://www.ee.ucla.edu/saurabh/publications/tpm_sensor_networks.pdf, 2015.

[33] C. Krau, F. Stumpf, and C. Eckert, Detecting node compromise in hybrid wireless sensor networks using attestation techniques, in *Security and Privacy in Ad-hoc and Sensor Networks*. Springer Berlin Heidelberg, 2007, pp. 203–217.

[34] H. Tan, W. Hu, and S. Jha, A TPM-enabled remote attestation protocol (TRAP) in wireless sensor networks, in *Proc. of the 6th ACM Workshop on Performance Monitoring and Measurement of Heterogeneous Wireless and Wired Networks*, Miami, FL, USA, 2011, pp. 9–16.

[35] S. Wagner, C. Krau, and C. Eckert, T-CUP: A TPM-based code update protocol enabling attestations for sensor networks, in *Security and Privacy in Communication Networks*. Springer Berlin Heidelberg, 2012, pp. 511–521.

[36] Y. M. Yussoff, H. Hashim, R. Rosli, and M. D. Baba, A review of physical attacks and trusted platforms in wireless sensor networks, *Procedia Engineering*, 2012. doi:10.1016/j.proeng.2012.07.215.

[37] Q. Gu and R. Noorani, Towards self-propagate mal-packets in sensor networks, in *Proc. of the First ACM Conference on Wireless Network Security*, Washington DC, USA, 2008, pp. 172–182.

Donglai Fu received the master degree in computer software and theory from Guizhou University, China, in 2007. He is currently a PhD candidate at the School of Compute Science and Technology, Taiyuan University of Technology, China. He is also a lecturer at the Software School, North University of China, China. His research interests include trusted computing, cloud computing security, and steganography.

Xinguang Peng received the PhD degree in computer science from Beijing Institute of Technology University, China, in 2002. He is currently a professor at the School of Compute Science and Technology, Taiyuan University of Technology, China. His research interest is computer network and security.