




Article

SNPL: One Scheme of Securing Nodes in IoT Perception Layer

Yongkai Fan ^{1,2,3}, Guanqun Zhao ^{3,*} , Kuan-Ching Li ^{4,*} , Bin Zhang ⁵ , Gang Tan ⁶ , Xiaofeng Sun ³ and Fanglue Xia ³

¹ Institute of Computer Science and Cybersecurity, Communication University of China, Beijing 100024, China; fanyongkai@gmail.com

² Tus College of Digit, Beijing 100084, China

³ Department of Computer Science and Technology, China University of Petroleum, Beijing 102249, China; 2017011316@student.cup.edu.cn (X.S.); 2017011320@student.cup.edu.cn (F.X.)

⁴ Department of Computer Science and Information Engineering, Providence University, Taichung 43301, Taiwan

⁵ Department of Electrical Engineering, University of South Carolina, Columbia, SC 29208, USA; zhangbin@cec.sc.edu

⁶ Department of Computer Science and Engineering, Penn State University, University Park, PA 16802, USA; gtan@cse.psu.edu

* Correspondence: 2017215520@student.cup.edu.cn (G.Z.); kuancli@pu.edu.tw (K.-C.L.)

Received: 20 December 2019; Accepted: 11 February 2020; Published: 17 February 2020



Abstract: The trustworthiness of data is vital data analysis in the age of big data. In cyber-physical systems, most data is collected by sensors. With the increase of sensors as Internet of Things (IoT) nodes in the network, the security risk of data tampering, unauthorized access, false identify, and others are overgrowing because of vulnerable nodes, which leads to the great economic and social loss. This paper proposes a security scheme, Securing Nodes in IoT Perception Layer (SNPL), for protecting nodes in the perception layer. The SNPL is constructed by novel lightweight algorithms to ensure security and satisfy performance requirements, as well as safety technologies to provide security isolation for sensitive operations. A series of experiments with different types and numbers of nodes are presented. Experimental results and performance analysis show that SNPL is efficient and effective at protecting IoT from faulty or malicious nodes. Some potential practical application scenarios are also discussed to motivate the implementation of the proposed scheme in the real world.

Keywords: IoT; security; security framework; IoT nodes

1. Introduction

The Internet of Things (IoT) is well-known for the integration of several technologies with communication systems [1]. The prosperity of IoT is not the reason to neglect its security issues. In fact, the security of IoT is far worse than people know in this respect. There are numerous examples in the real-world about IoT security. For instance, the embedded Radio Frequency Identification (RFID) tags in devices and equipment can transmit or reply to messages [2]. Without appropriate authentication mechanisms, data that are collected by sensor networks may be accessed or distorted by attackers [3]. The sensor system works in unattended status, such that adversaries can modify the information stored in the nodes or decides when the data are delivered to the destination [4]. In the case of node capture attacks, adversaries can capture or control smart devices, by physically replacing or tampering with the nodes and disguising a malicious node like a normal node, to interface with the system [5,6]. In these attacks, malicious nodes can transfer legal identity information, which was received from normal nodes to the target hosts, so that the rogue devices gain trust in IoT networks [7]. Moreover,

attackers can crack and gain the encryption key by using cryptanalysis attacks [8] or timing attacks [9]. Notably, this means that side-channel attacks can be used to illegally change or control intelligent devices such as IoT nodes [8].

To mitigate security risks, several security solutions have been proposed in recent years. Babar et al. [9] proposed an embedded security scheme to strengthen the internal security of a device itself by prevention, diagnosis, and elimination. Pacheco et al. presented a security framework that aims at smart cyberinfrastructures [10]. Besides, they proposed a general threat model that can be used for exploiting new safeguard methodology for IoT devices averting known or unknown attacks. Huang et al. developed a prototype security framework called SecIoT that provides a transparent and robust protection mechanism for relieving security threads [11]. Kalra et al. presented a protocol framework for mutual authentication based on Elliptic Curve Cryptography (ECC), which aims to achieve secure communication between embedded devices and the cloud [12]. Zhou et al. proposed a novel scheme of media awareness to promote the security heterogeneity of multimedia applications within a sensor network [13]. Tao et al. presented a framework of security services based on ontology, for the sake of ensuring confidentiality during an interactive process under an intelligent residential environment [14]. Kang et al. proposed a security framework and combined it with self-signature and access control technologies to avoid attacks and ensure the security of a smart home environment [15]. Meidan et al. used machine learning algorithms to identify and classify IoT devices via network traffic data, and proposed a scheme [16].

With these reported works, however the security of IoT nodes was not thoroughly studied, which leads to weakness in the security of the perception layer. The perception layer is the bottom of IoT, containing various IoT devices for collecting or measuring data from the real physical world, and then transferring to upper layers [17]. It can be said that this layer is of vital importance as it controls the source of the data, playing the role of “the last mile of communications” in IoT. Moreover, IoT nodes are the source of data, which is critical to the security of IoT. To address this problem, this paper develops a Securing Node in Perception Layer (SNPL) scheme. As shown in Figure 1, the primary purpose of the SNPL scheme is to ensure data authenticity from sources. In IoT nodes, the most important ones are sensors as they collect and transfer data for further use. The authenticity and trustworthiness of IoT nodes are critical for data analysis. The gathered data is useless if it is not from true nodes. The worst part of the entire thing is that data from malicious nodes, which are changed on purpose, can cause catastrophic consequences for decision-making systems based on them. By realizing this, SNPL aims to distinguish authentic nodes from malicious nodes to guarantee the reliability of collected information. To achieve this goal, a security scheme is proposed to ensure the trustworthiness of nodes in IoT perception layers. Different from other schemes [9–16], all sensitive operations are implemented in the Trusted Execution Environment (TEE), while general operations are carried out in the Rich Execution Environment (REE). TEE protects susceptible operations during the detection of malicious nodes, which enables isolated execution for key-related operations, access policy design, and identity recognition. Unique values and improved attribute-based signatures are used for the identification of IoT nodes.

The main contributions of this paper are outlined, as the following:

1. Propose a security scheme for the perception layer of IoT to identify and avoid potential hazards caused by unsafe nodes,
2. Combine the TEE technology with the SNPL scheme to provide a safe isolation space for sensitive operations, such as key generation and node identification,
3. The scheme satisfies the requirement of Confidentiality, Integrity, and Availability (CIA), as well as the lightweight property, which offers higher availability and portability for application in IoT devices, and
4. Use unique information of each device node as a kind of identifier such that forgery attacks and substitution attacks can be effectively reduced. Furthermore, appropriate encryption technologies can be integrated to enhance the security of the algorithm.

Note that this paper mainly focuses on the effectiveness and accuracy of the proposed framework based on our algorithm, of which the results are expressed as the success rate. Besides, the performance tests in experiments pay close attention to the various types and quantities of nodes in a small-scale local IoT. The later experiments are designed and simulated with the Raspberry Pi instead of real-time applications on a hardware device, for the processing time aiming at highly time-sensitive applications is not our research focus. Meanwhile, the performance of transmitting periodically nodes, sensors transmitting at fixed periodic time slots, or other conditions are not within the scope of our research either.

The remainder of this paper is organized, as follows. Section 2 introduces technical preliminaries such as TEE, Attribute-Based Signature (ABS) scheme, and a brief introduction of number theory, which are the technical base of the proposed scheme. Section 3 describes the SNPL scheme in detail. Section 4 presents experiments and evaluation metrics to assess the performance of SNPL. Section 5 discusses the possible applications, which is followed by concluding remarks in Section 6.

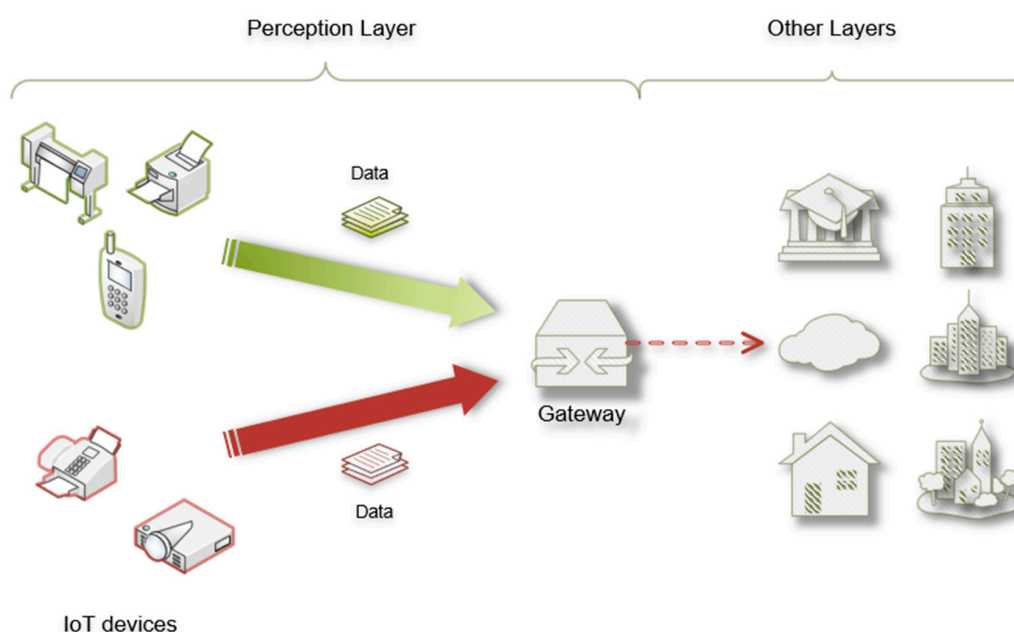


Figure 1. The attacker model in the real scenario.

2. Preliminaries

This section briefly introduces key technologies used in the proposed scheme. These technologies include TEE, ABS scheme, and two mathematical concepts, which are grouped with bilinear pairings and monotone span programs.

2.1. Trusted Execution Environment (TEE)

TEE is a tamper-resistant processing environment, which is independent of the normal environment [18]. It runs on an isolated kernel and has the ability to fight against physical attacks as well as software attacks in the main memory. The substance of TEE is dynamic and updated securely [19]. Usually, TEE is used to execute sensitive operations such as encryption or key generation, and it often has more restricted functions and rooms than REE.

Briefly, TEE is constructed to run security services, while REE is a platform for devices to request services. REE represents a normal processing environment with rich functions. For example, Windows, Linux, Android, and IOS can be referred. The basic interaction process between TEE and REE is supported by client Application Programming Interface (API) and shared memory as shown in Figure 2.

This interaction process provides a safe and feasible way to transfer information between the two isolated execution environments.

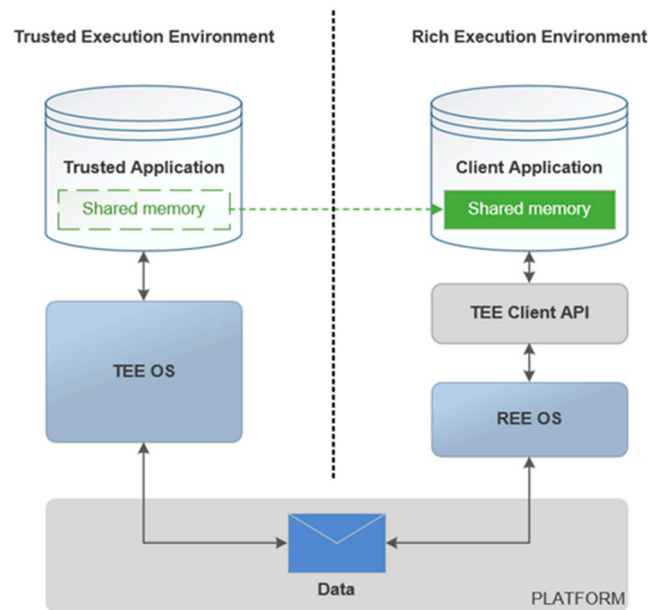


Figure 2. The interaction between Trusted Execution Environment (TEE) and Rich Execution Environment (REE).

2.2. Attribute-Based Signature (ABS) Scheme

The ABS scheme is a multifunctional control scheme. It allows for users to sign a message by taking advantage of fine-grained controls over identifying information. For this, the identity of a signer is uniquely represented by a collection of attributes and a signature of the signer is generated based on these attributes [20]. Afterward, the signature result can be used, for instance, for verifying a user's identification. In short, the rights of users depend on their attributes. Besides, more application samples can be referred in [20].

A typical ABS scheme has four main steps, which are presented [21]:

1. **Setup:** The authority or trusted third-party acquires a key pair: Public key (PK) and Master key (MK). Then, the PK will be opened and the MK will be kept privately. Both PK and MK are generated by a series of parameters (denoted as $para$). This step is shown as $(PK, MK) \leftarrow Setup(para)$.
2. **KeyGen:** In order to assign users a set of attributes (denoted as $Attr$), the third party or authority generates a Signing key (SK). The SK is given to users for further use. This step is expressed as $SK \leftarrow KeyGen(PK, MK, Attr)$.
3. **Sign:** In this step, the user obtains a Signature σ on the basis of a Claim-Predicate Υ , along with the PK , SK and the attribute set. The user can then use σ to sign a message m . The process is represented as $\sigma \leftarrow Sign(PK, SK, m, \Upsilon)$.
4. **Verify:** To verify the Signature of the message with the Predicate Υ , this step employs a Boolean function $value \leftarrow Verify(PK, m, \Upsilon, \sigma)$. According to the output, target parties can judge the identity of data generators.

2.3. Groups with Bilinear Pairings

Let G_1 , G_2 and G_T be the cyclic multiplicative groups, whose orders are all a prime p . Let g_1 and g_2 be the generators of G_1 and G_2 separately, and a map $m : G_1 \times G_2 \rightarrow G_T$. If $m(g_1, g_2)$ is a generator of G_T then $m : G_1 \times G_2 \rightarrow G_T$ is a bilinear pairing and it has the following properties [22]:

1. $m(g_1^a, g_2^b) = m(g_1, g_2)^{ab}$;
2. There exists $g_1 \in G_1, g_2 \in G_2$ that satisfy $m(g_1, g_2) \neq 1$;
3. There is always an effective method to calculate $m(g_1, g_2)$ for all $g_1 \in G_1, g_2 \in G_2$.

2.4. Monotone Span Programs

Suppose there is a matrix M with l rows and t columns, and a nonzero row vector \vec{v} , of which the number of coordinates is identical with the number of columns in M . A span program over a field F is expressed as $S = (M, \mu, \vec{v})$, in which M is the matrix with entries of F , \vec{v} is a target vector, and μ is the labeling of the rows of M .

A monotone span program means the labels of rows are simply positive literals $\{x_1, \dots, x_n\}$. The calculation results of monotone span programs are only monotone functions, and one monotone span program can calculate a monotone Boolean function [23].

Suppose δ is a monotone Boolean function. A monotone span program for δ over a domain D is a matrix $M_{l \times t}$ with entries in D . Besides, it includes a labeling function μ related to rows of M with input variables of δ . The relationship between δ and M is as follows: $\delta(x_1, x_2, \dots, x_n) = 1$ if and only if $\vec{v}M = [1, 0, 0, \dots, 0]$.

3. The SNPL Scheme

In this section, the SNPL scheme is elaborated for the improvement of reliability and robustness. First, the node fingerprint concept is described to ensure the uniqueness of the node. By doing so, the SNPL scheme can separate the true node from others. Second, the design of the scheme is discussed. Finally, the security proof of the proposed design is provided.

3.1. Node Fingerprint

To exploit the unique identification information of an IoT node, the concept of fingerprint is used to certify objects by unique features extracted from equipment information. Aside from the Universally Unique Identifier (UUID) of a device, more complex information or node attributes can be included. In the SNPL scheme, the Unique Identifier of a hardware device, which is the unique hardware configuration information of an IoT node, is the original information. If the node is replaced by a new one, the information will change correspondingly. The hash algorithm is used for a hashing of the information. The result of encryption is considered as a fingerprint value of a device node. The result of hash value can be used to ensure the trustworthiness of data sources and, therefore, the identity of the node is guaranteed.

3.2. Scheme Design

Figure 3 shows a usage scenario of the SNPL scheme in the IoT environment. Suppose there is an IoT network made up of many IoT nodes. Device nodes are located throughout the physical world and they gather information by various kinds of sensors embedded inside. The collected information will then be transmitted to the key nodes (for example, the gateways in Figure 3) in IoT. Finally, the key nodes transfer the processed data to other consumers. The main uncertainty for the security of the process is that the originality of data, such as the data gathered in key nodes, cannot be guaranteed. This is a hidden danger at the beginning of the process and may lead to failure of the whole process. To solve this problem, a security scheme is injected into the perception layer of IoT, between endpoint nodes (i.e., the IoT devices in Figure 3) and key nodes. The security scheme can effectively reduce hazards from the beginning by distinguishing nodes with the opposite status.

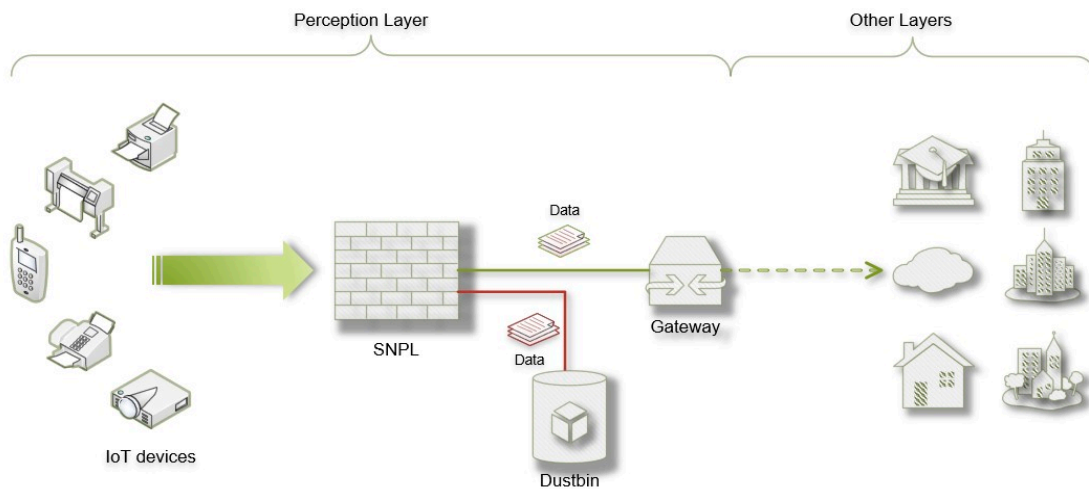


Figure 3. The usage scenario of SNPL in the IoT environment.

Figure 4 exhibits the dominant modules in the proposed security scheme, which is divided into two parts of execution by employing the REE and TEE technologies. Both parts are running on the same device as well as the object to build the SPNL scheme. In this scheme, REE runs common insensitive operations and connects unknown nodes. TEE runs the following five modules as trusted applications: *i. FMK-Gen* (responsible for fingermark generation of devices); *ii. MPK-Gen* (generating the master key and public key); *iii. IK-Gen* (create the individual key through the MK and fingermarks); *iv. Sign* (use *IK* to generate a signature for signing the incoming data from nodes and setting up an access policy); and, *v. Verify* (use *PK*, signatures, and the access policy to verify the identification of the data transmitter). After all sensitive processes have been executed in TEE, the results will be sent back to REE for later use.

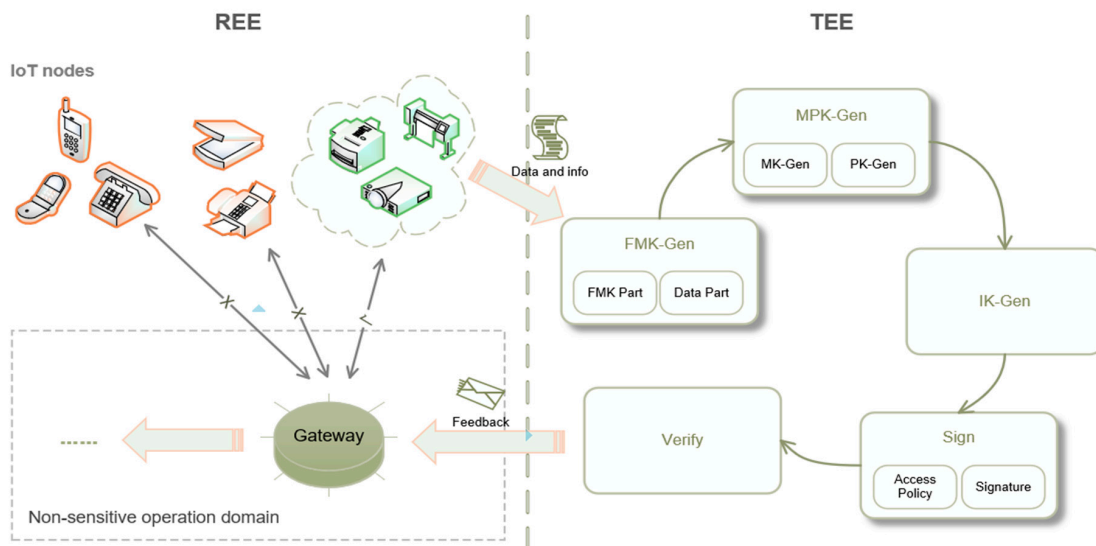


Figure 4. The dominant modules in a secure scheme.

Figure 5 elaborates on the execution process by formulating several algorithms in detail. At the end of the process, the SNPL scheme can distinguish malicious nodes from secure ones.

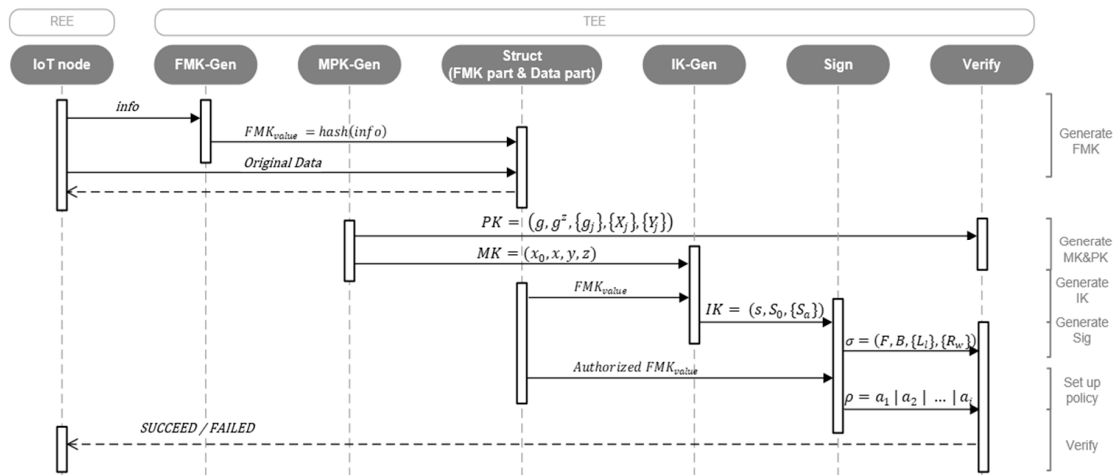


Figure 5. The sequence diagram of the architectural algorithm.

3.2.1. FMK-Gen

In this step, the device node’s unique authentication information is used to generate a safe value to be used later.

At first, the system defines a structure for each IoT device, which consists of two parts, *FMKpart* and *Data part* expressed as $Struct = \{FMK\ part, Data\ part\}$. Meanwhile, suppose that all IoT nodes with sensors are placed in REE. By the interaction process, IoT nodes transmit gathered data with their configuration information from REE to the *FMK-Gen* module in TEE.

After that, the *FMK-Gen* module extracts the identity information, such as UUID and International Mobile Equipment Identity (IMEI), from equipment nodes, and then uses the hash algorithm to generate a secure unique value FMK_{value} . This process can be expressed as: $FMK_{value} = hash(info)$. Here the Message-Digest Algorithm 5 (MD5) from OpenSSL library is used to calculate MD5 values as FMK_{value} .

At the end of this step, the system puts FMK_{value} into *FMKpart* of *Struct*. Meanwhile, the original data is sent into *Data part* by IoT terminal nodes. This can be expressed as: $FMK\ part \leftarrow FMK_{value}$ and $Data\ part \leftarrow original\ Data$.

Now, each equipment node corresponds to a structure built based on its unique identity and gathered data. The process can be realized by Algorithm 1 shown below, where i represents the current node and sum represents the maximum number of nodes:

Algorithm 1 Fingermark Generation

Input: unique information of an IoT device $info$

Output: IoT node’s FMK_{value}

Define $Struct \leftarrow \{FMK\ part, Data\ part\};$

For $i \leftarrow 0$ **to** sum **do**

$FMK_{value} \leftarrow hash(info);$

$FMK\ part \leftarrow FMK_{value};$

$Data\ part \leftarrow original\ Data;$

End for

Return $FMK_{value};$

3.2.2. MPK-Gen

In this step, the following groups and functions are defined first: two cyclic groups Z_p^* and Z_p in which prime order p is the size of the group, universe of attributes $U = Z_p^*$, a collision-resistant hash

function $H : \{0, 1\}^* \rightarrow Z_p^*$, and two cyclic groups G_1 and G_2 of size p that are equipped with a bilinear pairing $e : G_1 \times G_2 \rightarrow G_T$. Then, the required generators and parameters are emerged as follows:

$$g \leftarrow G_1; g_0, \dots, g_{w_{max}} \leftarrow G_2; x_0, x, y, z \leftarrow Z_p^*; \quad (1)$$

where g is a generator of G_1 , $g_0, \dots, g_{w_{max}}$ are generators of G_2 , x_0, x, y, z are randomly chosen from Z_p^* , respectively. Then, we set the following values:

$$X_0 = g_0^{x_0}, X_j = g_j^{x+z} (\forall j \in [0, w_{max}]); Y_j = g_j^{\frac{1}{y}}. \quad (2)$$

Next, the master key $MK = (x_0, x, y, z)$ and the public key $PK = (g, g^z, \{g_j | j \in [0, w_{max}]\}, \{X_j | j \in [0, w_{max}]\}, \{Y_j | j \in [1, w_{max}]\})$ can be generated.

The process is summarized in Algorithm 2:

Algorithm 2 Master Key and Public Key Generation

Input: cyclic groups Z_p^* and Z_p of size p , cyclic groups G_1 and G_2 of size p

Output: MK and PK

Define $U \leftarrow Z_p^*$;

$Z_p^* \leftarrow H : \{0, 1\}^*$;

$G_T \leftarrow e : G_1 \times G_2$;

Choose $g \leftarrow G_1$;

$g_0, \dots, g_{w_{max}} \leftarrow G_2$;

$x_0, x, y, z \leftarrow Z_p^*$;

For $j \leftarrow 0$ **to** w_{max} **do**

$X_0 \leftarrow g_0^{x_0}$;

$X_j \leftarrow g_j^{x+z}$;

$Y_j \leftarrow g_j^{\frac{1}{y}}$;

End for

$MK \leftarrow (x_0, x, y, z)$;

$PK \leftarrow (g, g^z, \{g_j | j \in [w]\}, \{X_j | j \in [w]\}, \{Y_j | j \in [1, w_{max}]\})$;

Return MK and PK ;

3.2.3. IK-Gen

In this step, it is assumed that $U' \subseteq U$; and $\forall a \in U'$, where U' is an attribute set that contains attributes satisfying the access policy, and a is one of the legal attributes. For a randomly selected generator $s \leftarrow G_1$, the following values can be set:

$$S_0 = s^{1/x_0z}, S_a = s^{1/(x+ya)} \quad (3)$$

Note that FMK_{value} is used to construct the value a , which is one of $a \in U'$:

$$a \leftarrow FMK_{value} \quad (4)$$

According to the generated MK and the attribute set U' , the user's individual key is:

$$IK_{U'} = (s, S_0, \{S_a\}) \quad (5)$$

The above process can be summarized as Algorithm 3:

Algorithm 3 Individual Key Generation**Input:** master key MK and Fingerprint FMK_{value} of an IoT node**Output:** individual key IK of an IoT node**Define** $U' \subseteq U$; $\forall a \in U'$;**Choose** $s \leftarrow G_1$;**For** $i \leftarrow 0$ **to** sum **do** $a \leftarrow FMK_{value}$; $S_0 \leftarrow s^{\frac{1}{x_0^2}}$; $S_a \leftarrow s^{\frac{1}{(x+ya)}}$;**End for** $IK_{U'} \leftarrow (s, S_0, \{S_a\})$;**Return** $IK_{U'}$;

3.2.4. Sign

Firstly, an access policy is set up to decide which user can get into the system on the basis of its attributes. Then, the *Sign* module defines a predicate $\rho(U') = 1$; and calculates a matrix $M_{l \times w}$ based on the predicate, and a label vector a_i , which indicates the relationship between an attribute and its corresponding row. This step means that policy ρ corresponds to the monotone span program $M \in (Z_p)^{l \times w}$ with the row labeling $a : \{l\} \rightarrow U$. According to the program M given above, a vector v corresponding to U' is computed through the following rules: v_i has two values of 0 and 1, which $v_i = 1$ means the corresponding attribute a_i is used in the access policy while $v_i = 0$ means a_i is not used or does not exist. Finally, ε is calculated as $\varepsilon = H(d \parallel \rho)$.

Note that only the set of accessible FMK_{value} of authorized nodes is applied to the policy ρ . The FMK_{value} of those inaccessible nodes are not included in ρ . The policy ρ is constructed through OR operation with authorized FMK_{value} as follows:

$$\rho = a_1 | a_2 | \dots | a_i \quad (6)$$

where $a_1 = FMK_{value1}, a_2 = FMK_{value2} \dots a_i = FMK_{valuei}$. The process can be summarized in Algorithm 4:

Choose random generators t_0 from Z_p^* and t_1, t_2, \dots, t_l from Z_p^* , which are expressed as $t_0 \leftarrow Z_p^*$ and $t_1, t_2, \dots, t_l \leftarrow Z_p^*$, respectively. Then, set values F, B, L_i and R_j where $\forall i \in \{l\}$ and $\forall j \in \{w\}$:

$$F = s^{t_0}, B = S^{t_0+z} \quad (7)$$

$$L_i = (S_{a_i} v_i)^{t_0} \cdot g^{t_i(z+\varepsilon)}; R_j = \prod_{i=1}^l (X_j Y_j^{a_i})^{M_{ij} t_i} \quad (8)$$

The signature for signing the data in *Datapart* is set as $\sigma = (F, B, L_1 \dots L_l, R_1 \dots R_w)$. The process can be summarized in Algorithm 5.

Algorithm 4 Set up An Access Policy

Input: FMK_{value} of addressable IoT nodes
Output: an access policy ρ
Define $\rho(U') \leftarrow 1$;
 $M_{l \times w} \leftarrow M \in (Z_p)^{l \times w}$;
 $a : \{l\} \rightarrow U$;
 $\varepsilon \leftarrow H(d||\rho)$;
For $i \leftarrow 0$ **to** l **do**
 $a_i \leftarrow FMK_{value_i}$
End for
 $\rho \leftarrow a_1 | a_2 | \dots | a_l$;
Return ρ ;

Algorithm 5 Signature Generation

Input: individual key IK of an IoT node
Output: signature σ of a specific IoT node
Choose $t_0 \leftarrow Z_p^*$;
 $t_1, t_2, \dots, t_l \leftarrow Z_p$;
For $i \leftarrow 0$ **to** l **&&** $j \leftarrow 0$ **to** w **do**
 $F \leftarrow s^{t_0}$;
 $B \leftarrow S_0^{t_0+z}$;
 $L_i \leftarrow (S_{a_i}, v_i)^{t_0} \cdot g^{t_i(z+\varepsilon)}$;
 $R_j \leftarrow \prod_{i=1}^l (X_i Y_j^{a_i})^{M_{ij} \cdot t_i}$;
End for
 $\sigma \leftarrow (F, B, L_1 \dots L_l, R_1 \dots R_w)$;
Return σ ;

3.2.5. Verify

Based on PK , σ , and ρ generated in the above algorithms, the system can determine whether an input should be accepted or rejected. The result SUCCEED means the node gets the authority successfully, while the result FAILED means the node is access-denied. The first step is checking the value of F . $F = 1$ indicates FAILED. For $F = 0$, the following conditions need to be check for all $j \in \{w\}$:

1. $e(B, g_0^{x_0}) = e(sF^{\frac{1}{z}}, g_0)$;
2. For $j = 1, \dots, l$, $\prod_{i=1}^l e(L_i, g_j^{(x+\frac{a_i}{y}+z) \cdot M_{ij}}) = e(F, g_1^{\frac{1}{z}}) e(g, R_1^{z+\varepsilon})$; for $j \neq 1, \dots, l$, $\prod_{i=1}^l e(L_i, g_j^{(x+\frac{a_i}{y}+z) \cdot M_{ij}}) = e(g^{z+\varepsilon}, R_j)$.

If all od the above conditions hold, the results is SUCCEED. The system then transmits the result from TEE to REE. The incoming data in *Datapart* of authorized nodes will be sent back to REE for further use. Meanwhile, the data from rejected nodes are deleted. This process can be summarized in Algorithm 6:

Algorithm 6: Verify**Input:** public key PK , signature σ and predefined access policy ρ **Output:** SUCCEED or FAILED**If** $F = 1$ **Then** FAILED;**Else****Check** $e(B, g_0^{x_0}) = e(sF^{\frac{1}{z}}, g_0)$;**If YES and** $j = 1$ **Check** $\prod_{i=1}^l e(L_i, g_j^{(x+\frac{a_i}{y}+z) \cdot M_{ij}}) = e(F, g_1^{\frac{1}{z}}) e(g, R_1^{z+\epsilon})$;**If YES, return** SUCCEED;**Else if YES and** $j \neq 1$ **Check** $\prod_{i=1}^l e(L_i, g_j^{(x+\frac{a_i}{y}+z) \cdot M_{ij}}) = e(g^{z+\epsilon}, R_j)$;**If YES, return** SUCCEED;**Else return** FAILED;

3.3. Security Proof

This section provides a security proof of our scheme, which covers illustrations of correctness, privacy, and unforgeability with formulations. The scheme satisfying the proof can be considered as a safe solution once proved correct, entirely private and unforgeable [20].

Correctness means that in the light of PK, MK, IK , access policy, and correspondingly generated true signatures, correct verification results and equations can be obtained in the process of verification. By the detailed explanation in Section 3.2 and the direct substitution method, it is obvious that the correctness is fulfilled.

Privacy means the attacker never receive attributes and IK of a legal node by the generated signature. Here we can see that even though there are different attribute sets leading to different IKs , any legal attribute sets resulting $Verify(PK, d, \rho, \sigma) = 1$ have the identical distribution while calculating signatures under the same ρ . When an attacker generates a hoped-for signature without legal attributes, there is a neglectable possibility to gain a signature satisfying the access policy, which makes $\rho(U') = 1$ and $Verify(PK, d, \rho, \sigma) = 1$. Still, the terms in $\sigma = (F, B, L_1 \dots L_l, R_1 \dots R_w)$ are unique corresponding while successfully verified. So, the IK and signature generated at different times are distinct, which draws to the conclusion that privacy is guaranteed.

The unforgeability means the success probability of an attacker in any polynomial times is ignorable when faced with the following circumstance:

1. Generate public key and master key by $(PK, MK) \leftarrow MPK-Gen$, and send the results to the attacker; and,
2. The attacker has access to the $IK-Gen$ module and $Sign$ module such that it can generate a forged signature σ^* to pass the validation of the access policy.

In other words, when an attacker has incorrect access structure and inappropriate attributes, but eventually gains a correct verification result, we can say that the attacker succeeds to get access.

The remaining of this section verifies this property. Let $M \in (Z_p)^{l \times w}$ be the monotone span program of ρ , a be the row labeling that $a : [l] \rightarrow U$, and $\epsilon = H(d||\rho)$. Then, the following steps are implemented:

1. Randomly choose $h_1, \dots, h_l \leftarrow Z_p$ and $u \leftarrow Z_p^*$;
2. Calculate $r_j = \frac{1}{y(z+\epsilon)} \left\{ \sum_{i=1}^l \frac{[(x+ya_i)h_i - uv_i](xy+yz+a_i)M_{ij}}{x+ya_i} \right\}$ for all $j \in [w]$;
3. Let the signature $\sigma = (\sigma_1, \sigma_2, \sigma_3, \sigma_4)$, where $\sigma_1 = g^{t_0 s}$, $\sigma_2 = g^{s(t_0+z)/x_0 z}$, $\sigma_3 = \{g^{h_i} | i \in [l]\}$, $\sigma_4 = \{g^{r_j} | j \in [w]\}$.

It is necessary to use the programmatic techniques of universal groups to certify the unforgeability as shown below. Before formal certification, parameters are set similar to previous algorithms in Section 3.2. An assumption is then made that the fake signature of an attacker is defined as: $\sigma^* = (g^{u^*}, g^{b^*}, \{g^{h_i^*} | i \in [l]\}, \{g^{r_j^*} | j \in [w]\})$, with data d^* and access policy ρ^* , where $(d^*, \rho^*) \neq (d^{(q)}, \rho^{(q)})$. Similarly, let $M^* \in (Z_p)^{l \times w^*}$, of which a^* is the row labeling. At last, let $\varepsilon^* = H(d^* || \rho^*)$.

Note that $u^* \neq 0$ and $b^* = \frac{s(t_0+z)}{x_0z}$. To construct the counterfeit signature, the following equation is constructed:

$$\begin{aligned} & \sum_{i=1}^{l^*} h_i^* M_{i,j}^* (xy + yz + a_i^*) \Delta_j \\ &= \sum_{i=1}^{l^*} \frac{xy+yz+a_i^*}{x+ya_i^*} u^* v'_j \Delta_j + y(z + \varepsilon^*) r_j^*, j \in [w^*] \end{aligned} \tag{9}$$

where $v'_j = [1, 0, \dots, 0]$. Here a hypothesis is that the above equation holds, followed with getting a contradictory result. That is to say, here reduction to absurdity is used as an effective method, aiming at reaching an outcome that the attacker can produce a legitimate signature using *IK*, so that the signature is not a counterfeit.

Let $Lin(P)$ be the collection of multilinear polynomials, which P is defined as:

$$\begin{aligned} P = & \{1, x_0, \Delta_0, z\} \\ & \cup \{\Delta_j, (x+z)\Delta_j, \Delta_j/y | j \in [w]\} \\ & \cup \{o_s, o_s/x_0z, o_s/(x+ay) | o \in [n], a \in U'_s\} \\ & \cup \{h_i^{(q)}, u^{(q)}, b^{(q)}, r_j^{(q)} | q \in [v], i \in [l^{(q)}], j \in [w^{(q)}]\} \end{aligned} \tag{10}$$

where $o \in Z_p^*$ is chosen randomly and P is the attribute set with coefficients in Z_p^* . Meanwhile, let $Hom(P)$ be the collection of homogeneous polynomials, which is the subset of a multilinear polynomial set, i.e., $Hom(P) \subset Lin(P)$. As our proof is based on the mathematical theory of multilinear functions and homogeneous polynomial, it comes to the conclusion that the expressions that are provided by the counterfeit of an attacker cannot embody certain specific terms.

Since it is obviously that $u^*, b^*, \{h_i^* | i \in [l]\}, \{r_j^* | j \in [w]\} \in Lin(P)$, as well as $u^* = b^* \frac{x_0 t_0 z}{t_0 + z}$, we can conclude that:

$$u^* \in Hom(\{\Delta_0 x_0\} \cup \{o_s | s \in [n]\} \cup \{u^{(q)} | q \in [v]\}) \tag{11}$$

Since $\Delta_j | y(z + \varepsilon^*) r_j^*$ and consequently $\Delta_j | r_j^*$, we get:

$$r_j^* \in Hom(\{\Delta_j, \Delta_j(x+z), \Delta_j/y\} \cup \{r_j^{(q)} | q \in [v]\}) \tag{12}$$

Here it is assumed that $v'_{j_0} \neq 0$ and u^* includes term $\Delta_0 x_0$. Therefore, $\sum_{i=1}^{l^*} \frac{xy+yz+a_i^*}{x+ya_i^*} u^* v'_j \Delta_{j_0}$ contains $\Delta_0 x_0 \Delta_{j_0}$. Note that $\Delta_0 x_0 \Delta_{j_0}$ is impossible to exist in $y(z + \varepsilon^*) r_{j_0}^*$, or appear in $\sum_{i=1}^{l^*} h_i^* M_{i,j_0}^* (xy + yz + a_i^*) \Delta_{j_0}$. As a result, it concludes that:

$$u^* \in Hom(\{o_s | s \in [n]\} \cup \{u^{(q)} | q \in [v]\}) \tag{13}$$

Assume that there is a term Δ_j in r_j^* . As u^* has no constant term, leading that $u^* v'_j \Delta_j$ is incapable of contributing Δ_j to the equation. It is the same as $\sum_{i=1}^{l^*} h_i^* M_{i,j}^* (xy + yz + a_i^*) \Delta_j$. But it is necessary to contribute Δ_j and Δ_j/y for the equation's right side, so:

$$r_j^* \in Hom(\{\Delta_j(x+z), \Delta_j/y\} \cup \{r_j^{(q)} | q \in [v]\}) \tag{14}$$

Assume that r_j^* has the term $r_j^{(q)}$, it is necessary to bring $y(z + \varepsilon^*)r_j^*$ to the equation's right side, which generates a term with the coefficient of $y(z + \varepsilon^*)/y(z + \varepsilon^{(q)})$. u^* and $\{h_i^* | i \in [l]\}$ cannot contribute $y(z + \varepsilon^*)/y(z + \varepsilon^{(q)})$ to the equation. As ε^* and $\varepsilon^{(q)}$ are always different, therefore, we can conclude:

$$r_j^* \in Hom(\{\Delta_j(x+z), \Delta_j/y\}) \quad (15)$$

As mentioned early, it is assumed that $v'_{j_0} \neq 0$ for j_0 . As $y(z + \varepsilon^*)r_{j_0}^*$ and $\sum_i^{l^*} h_i^* M_{i,j_0}^* (xy + yz + a_i^*) \Delta_{j_0}$ cannot provide the term $u^{(q)}$, u^* is impossible to contain the monomial $u^{(q)}$. Accordingly,

$$u^* \in Hom(\{o_s | s \in [n]\}) \quad (16)$$

Ultimately, we can come to a conclusion that:

$$r_j^* \in Hom(\{\Delta_j(x+z), \Delta_j/y\}) \quad (17)$$

$$u^* \in Hom(\{o_s | s \in [n]\}) \quad (18)$$

To make the Equation (9) tenable, let $h_i^* = w_i^*(O_i) + \delta^*(R/O_i)$ where $O_i = \{o_s/x + a_i y | a_i \in U'_s, s \in [n]\}$, for any terms of u^* should also be provided by the left side of (9) to realize the equality of this expression. Here we divide h_i^* into two addends to actualize the situation that o_k only exists in one part of h_i^* . There is:

$$\sum_{i=1}^{l^*} w_i^* M_{i,j}^* (xy + yz + a_i^*) = \sum_i^{l^*} \frac{xy + yz + a_i^*}{x + ya_i^*} u^* v'_j (j \in [w]) \quad (19)$$

Here a vector v_i^* is defined, which makes $v^* M^* = v'_1 \dots v'_t = 1, 0, \dots, 0$ hold. Besides, according to prior works, U'_{s_0} should comprise the attribute a_i^* . Then, v_i^* can be constructed as follows:

$$v_i^* = \left[\frac{o_{s_0}}{x + ya_i^*} \right] w_i^* / [o_{s_0}] (i \in [l]) \quad (20)$$

in which $[o_{s_0}]$ is the coefficient of o_{s_0} in u^* ($o_{s_0} \neq 0$). Based on the above mathematical and derivation, we have $\rho^*(U'_{s_0}) = 1$, which confirms that the signature is not a counterfeit. That is to say, the property of unforgeability is proved.

4. Experimental Evaluation

In this section, details about experiments on simulating the SNPL framework are described. Besides, several performance measurements are defined such that the results can be analyzed and compared from several different aspects.

4.1. Experiment Design

It is necessary to explain why experiments are not carried out on a large scale here. For the whole application scenario, the scheme simulated in experiments is a representative or an epitome of different local IoT, which is connected or communicating with each other. That is to say, other application scenarios are extensions of similar situations. So, conducting experiments in more extensive or more situations makes no sense, for those are seen as repetitive actions. The scenario explanation is shown in Figure 6.

Two experiments are conducted to simulate real IoT application scenarios. The first experiment is to test the effectiveness and verification capability of the SNPL scheme with a single node of different types. This experiment provides information on the success rate and means the processing time of one single node. The second experiment aims to evaluate the performance under different

numbers of nodes. The goal is to test whether the number of devices has an impact on performance. In the experiments, the processing time of various nodes is recorded to calculate the mean value. Besides, the accuracy of different nodes is also recorded and compared to ensure the universality of the SNPL scheme.

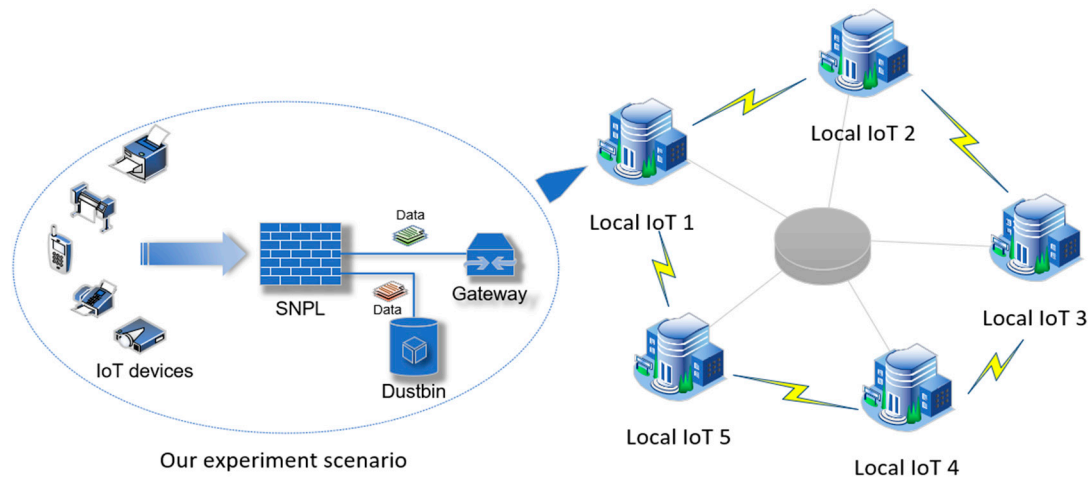


Figure 6. Scenario explanations of experiments and realistic scenes.

As mentioned early, the first experiment aims to evaluate the effectiveness of the SNPL scheme with one single node, as shown in Figure 7a. In this experiment, 20 different types of sensors are connected to the Raspberry Pi and each sensor is a separate node. Our purpose is to test whether the SNPL scheme can identify the data source. The finger marks are generated by using the configuration text files of devices, which are stored in a specific folder. A policy with one legal attribute is then defined to verify nodes' identification one by one. Among all of these nodes to be tested, the normal one represents the objective whose attribute is legal and set into the policy, while the malicious one represents the node whose attribute is illegal and not in the policy. Then, individual keys of devices are generated by using the fingermarks as an attribute in the algorithm. These individual keys are used to sign messages, after which the policy is used to provide verification of these nodes. In this process, the processing time of the SNPL scheme of 20 different nodes are recorded, their mean value is calculated, and the success rate is obtained.

The second experiment is to compare the running time and accuracy of the algorithm under different quantities of nodes, as shown in Figure 7b. This experiment simulates a real-world scenario and evaluates whether the number of parallel nodes has an influence on the SNPL scheme. The operation is substantially similar to that of the first experiment, in which the only difference is the number of processed nodes. Here virtual nodes are defined with respective fingermarks for all targets and an access policy with all legal attributes (i.e., the attributes which satisfy the policy). Then, each defined node signs up one message and generates a signature for verifying with the policy. After that, the verification results are obtained. In this process, the processing time with a different number of nodes is recorded and compared. The accuracy in terms of throughput rate and blocking rate is also compared.

The experiment is carried out on the Ubuntu operating system. To implement the proposed SNPL scheme, Raspberry Pi 3 with different types of sensors is used to simulate different IoT nodes (i.e., a temperature sensor attached to the Raspberry Pi 3 simulates the first node, while the photoelectric sensor attached can be the second simulated node, and others). Open-TEE is used to construct the TEE in Ubuntu, while the normal execution environment of Ubuntu is considered as the REE. The Ubuntu system is deployed in VMware workstation 14, which is installed on a Dell Inspiron 15-5577 notebook. Our platform is developed in the C language using qtcreator-3.6.1. The hardware are shown in Figure 8 and the device specifications are listed below:

Operating system: Ubuntu 14.04 LTS.

Hardware simulation: Raspberry Pi 3 with sensors.

CPU: Intel® Core™ i7-7700HQ CPU @ 2.80GHz

Memory: 979.8 MB

Storage: 41.1 GBxc

The SNPL scheme can be taken as a “small nodes gateway” to control data transfer to the real gateway. The “small nodes gateway” can be replicated to simulate different scales of IoT. Therefore, the experiment can be seen as a classic case of different kinds of IoT environments and the conclusions can be extended to various real-world scenarios.

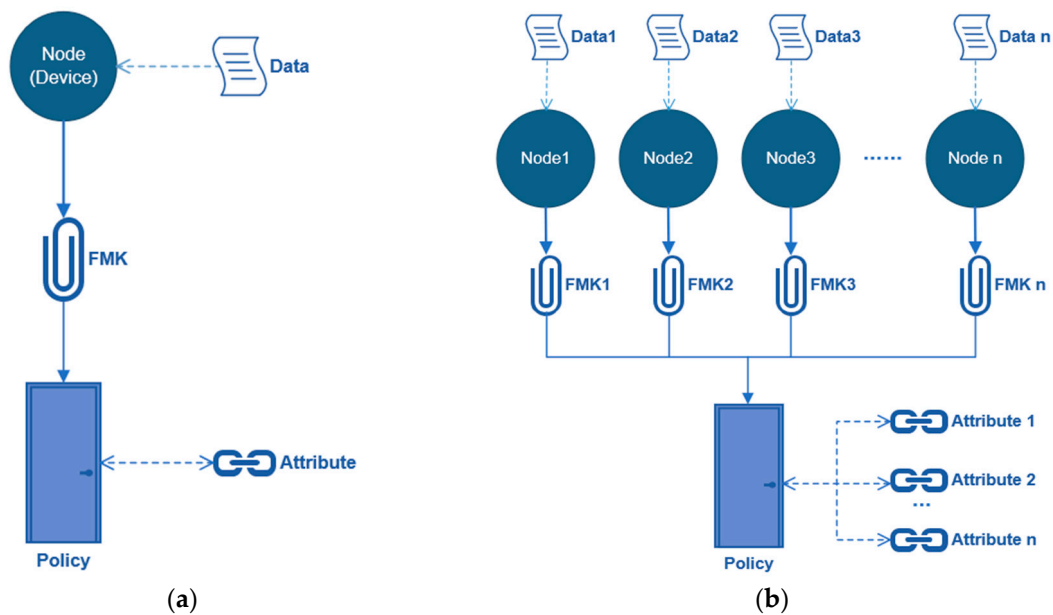


Figure 7. (a) The first test based on a single node; and, (b) The second test based on multiple nodes.

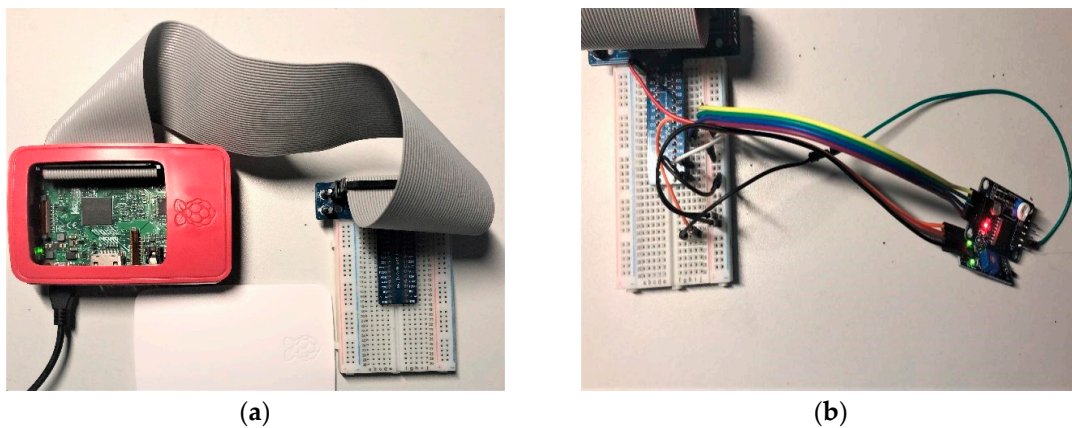


Figure 8. (a) Hardware device as an IoT node; and, (b) Breadboard with sensors installed.

4.2. Experimental Results

In this section, experimental evaluation and comparison of the SNPL scheme are presented. First, the running time of the entire process of the experiments is discussed. Then, the accuracy under different types and numbers of nodes are compared.

4.2.1. Processing Time

Figure 9a shows the processing time with 20 kinds of normal nodes. The result shows that the execution time varies with different nodes. In our experiments, the average execution time of 20 normal nodes is about 365.661 ms. Figure 9b shows the result of identifying 20 kinds of malicious nodes. Also, the SNPL scheme has different processing periods when dealing with different nodes. It must be said that the obvious discrepancy of node 1 is just due to its node type. And the difference of node 2 between normal and malicious properties is just an experimental error, which may be caused by the system operation time, the network latency, or other factors. The chart shows that the average time is approximately 365.676 ms. It can be concluded from the statistics that the processing time for the detection of normal and malicious nodes is comparable. In other words, the performance of the scheme on identifying nodes is not affected by the properties of nodes.

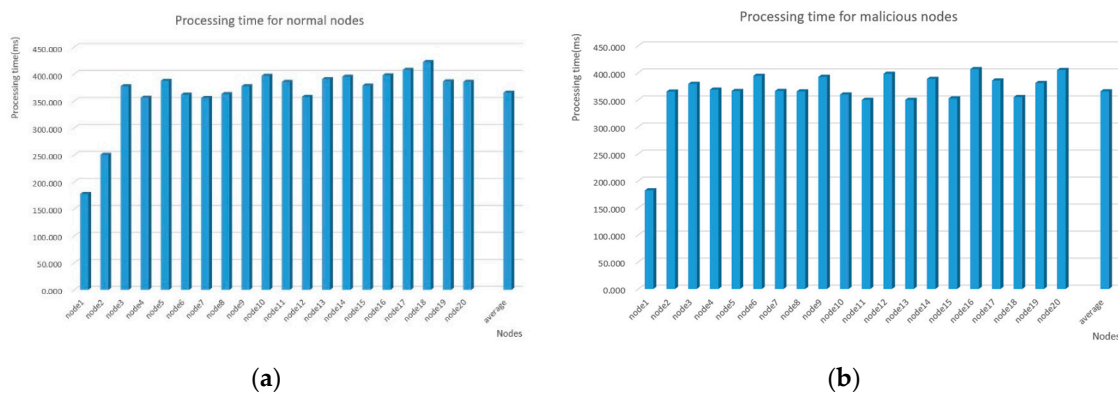


Figure 9. (a) Processing time for normal nodes; and, (b) Processing time for malicious nodes.

Table 1 shows the processing time with different numbers of nodes (1 node, 5 nodes, 10 nodes, 15 nodes, and 20 nodes) excerpted from the complete experimental results. In each case, the amounts of normal nodes and malicious nodes are shown to evaluate the verification accuracy.

Figure 10 shows a line chart, given by efficiency vs. different number nodes, for an intuitive description of the results. Here the efficiency is represented by the processing time against different numbers of nodes. Shorter processing time indicates a higher efficiency of the SNPL scheme. The chart shows that the processing time increases almost linearly with the increase on the number of nodes. That is, there is a linear increase relationship between the number of nodes and processing time.

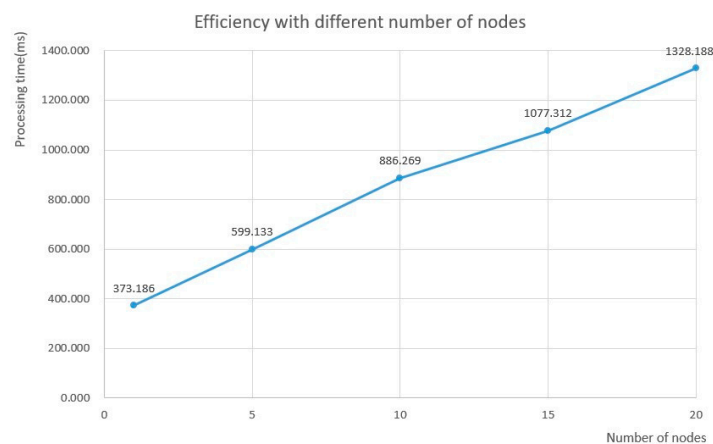


Figure 10. Efficiency changes in disposing of different numbers of nodes.

It must be stated that the scenario of a small-scale local IoT (i.e., a house, an office, or anywhere) does not contain a large number of smart nodes. To explain the performance results, here consider

an application scenario in which a man wears some wearable devices. Within this scenario, the total amount of smart nodes the man carries will not usually exceed 10. That is, the quantity of nodes in a real application is generally within the validation range of our experiment. Based on experimental outcomes of total running time and identifying precision shown in this experiment, it can be concluded that our proposed scheme improves security without compromising performance.

Table 1. Processing time with different numbers of nodes.

Total Number of Emulated Nodes	Number of Normal Nodes	Number of Malicious Nodes	Processing Time(ms)
1	0	1	362.081
	1	0	384.290
5	0	5	744.768
	1	4	549.926
	2	3	564.753
	3	2	436.345
	4	1	680.267
10	0	10	908.323
	2	8	883.820
	4	6	878.198
	6	4	906.796
	8	2	953.472
15	0	15	1098.231
	3	12	1083.484
	6	9	1108.137
	9	6	1078.030
	12	3	1089.419
20	0	20	1318.808
	5	15	1463.790
	10	10	1449.573
	15	5	1125.775
	20	0	1278.245

4.2.2. Accuracy

The first experiment shows that the SNPL scheme can recognize the identity of an inspected node correctly. During the experiment, all normal nodes are proved to be safe nodes as expected while all malicious nodes are proved unsafe. The results are obtained based on the following calculation expressions:

$$AC_{sgl_s} = \frac{T_{safe}}{N_{sum_s}} AC_{sgl_m} = \frac{T_{mali}}{N_{sum_m}} \quad (21)$$

$$AC_{sgl_m} = \frac{T_{mali}}{N_{sum_m}}; \quad (22)$$

where AC_{sgl_s} and AC_{sgl_m} represent the accuracy of safe node detection and malicious node detection, respectively, T_{safe} and T_{mali} are the times of successful validations of safe and malicious nodes, respectively, N_{sum_s} and N_{sum_m} are the total number of safe and malicious nodes to be measured, respectively. The verification accuracy of single node identification is shown in Table 2.

Table 2. Accuracy result with a single node.

Secure Node			Malicious Node		
N_{sum_s}	T_{safe}	AC_{sgl_s}	N_{sum_m}	T_{mali}	AC_{sgl_m}
1	1	100%	1	1	100%
5	5	100%	5	5	100%
10	10	100%	10	10	100%
15	15	100%	15	15	100%
20	20	100%	20	20	100%

In the second experiment, the accuracy is indicated by the percentage of all nodes that are verified correctly over the total number of nodes. Ideally, all nodes are verified correctly, regardless of the number of unidentified nodes. That is to say, on the premise that the total number of nodes unchanged, the amounts of normal nodes and malicious nodes do not affect the accuracy of recognition. The accuracy of detection on multi-node denoted as AC_{mul} , is given as:

$$AC_{mul} = \frac{P_{rec_suc}}{N_{sum}} = \frac{N_{suc_rec}}{N_{sum}} = \frac{N_{rec_s} + N_{rec_m}}{N_{sum}} \quad (23)$$

where N_{rec_s} and N_{rec_m} are the amounts of safe and malicious nodes distinguished successfully, respectively, while N_{sum} is the total amounts of nodes under test. Experimental results indicate that that system accuracy is 100% no matter how the number of nodes changes. Some experimental results are tabulated in Table 3. In all cases, the accuracy of node authentication is 100% with a relatively short checking period. The results demonstrate the exceptional performance of SNPL scheme in identification.

Table 3. Accuracy result with multiple nodes.

N_{sum}	N_{rec_s}	N_{rec_m}	AC_{mul}
5	0	5	100%
	1	4	
	2	3	
	3	2	
	4	1	
10	0	10	100%
	2	8	
	4	6	
	6	4	
15	8	2	100%
	0	15	
	3	12	
	6	9	
20	9	6	100%
	12	3	
	0	20	
	5	15	
	10	10	
20	15	5	100%
	20	0	

5. Application

This section discusses three potential application scenarios for the proposed SNPL scheme.

5.1. Smart Healthcare

IoT technologies can benefit the healthcare domain, in which tracking, identification, and data collection are the most critical applications [24]. In this field, light-weight monitoring or wearable devices play essential roles in monitoring patients' conditions, recording and sending their real-time data to doctors or hospitals automatically. These monitoring and wearable devices have access to a patient's sensitive and private data such as blood pressure and heart rate, from different nodes. Assume that there exists a malicious node and this node can modify real data and send the revised information to hospitals or doctors, which may lead to misdiagnosis. In this situation, our security scheme can identify the malicious node and prevent the modification of critical data. In other words, SNPL can protect data trustworthily.

5.2. Smart Building

In IoT, many kinds of smart devices are used in a building, constituting a local sensor network. Sensors and actuators arranged in the building can make people's life more comfortable. For example, rooms heating can adapt to the weather condition and our preferences; rooms lighting can change based on the time of a day; the electrical equipment can be automatically on/off to save energy; monitoring and alarm systems can avoid domestic incidents [25]. In a word, intelligent nodes with sensors keep watching on the status of the whole building. For better management and safety, a gateway node can be used for each family or office. All data collected by the smart equipment will be transferred and stored in the gateway node. To verify the security of the data, our SNPL security scheme can be used in the transmission between endpoint nodes and the gateway node, which can take precautions against forged nodes and invaders. Moreover, it is also useful in the protection of household information and office data.

5.3. Smart Transportation

Smart transportation, which is also known as intelligent transportation, is a typical IoT-based application [26]. This system consists of a high number of smart vehicles connected via wireless networks [27], and smart vehicles can perceive road conditions and share traffic information with others. For such, every vehicle is equipped with Electronic Control Units (ECUs) for controlling subsystems and sharing gathered data within the vehicle. Besides, vehicles can connect to external networks for communications [28]. However, as adversaries may take over ECUs through launching attacks against endpoint nodes and subsystems, through there are some data protection schemes [29], data can be modified and transferred to other ECUs or vehicles and cause damages to the whole transportation system [30,31]. To protect against the security thread, our SNPL security scheme can be deployed between subsystems and the ECU for a reliability check on source nodes' identities and messages. As a result, malicious data from unsafe nodes do not get into the transportation network and the stability of the whole system can be guaranteed.

6. Conclusions

In this paper, a SNPL scheme is proposed to ensure the trustworthiness of IoT nodes that are based on nodes attributes. In such scheme, the MD5 algorithm is used to generate a private value for an IoT node that is used as an identification attribute of the IoT node. Next, the node's attribute and predefined access policy are applied for node authentication in TEE, which is a trusted development circumstance for realizing sensitive operations. To demonstrate the effectiveness of the proposed scheme, a series of experiments are conducted to evaluate the performance in terms of processing time, accuracy and success rate. Experimental results show that the proposed SNPL security scheme

can identify normal nodes and malicious nodes that are based on their unique identify information with high efficiency and accuracy. Also, this indicates that the proposed SNPL scheme can protect the security of the source data at the beginning of the IoT interaction flow. Future directions of this research include the investigation of optimized algorithms design and their implementation in real applications.

Author Contributions: Conceptualization, Y.F. and G.Z.; Supervision, Y.F.; Resources, Y.F.; Data curation, G.Z.; Formal analysis, G.Z.; Investigation, G.Z., X.S. and F.X.; Methodology, G.Z.; Project administration, G.Z.; Software, G.Z.; Writing—original draft, G.Z.; Writing—review & editing, Y.F., K.-C.L., B.Z. and G.T. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by CERNET Innovation Project, under grant NGII20180406; by Beijing Higher Education Young Elite Teacher Project, under grant YETP0683; by Beijing Higher Education Teacher Project, under grant 00001149.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Dai, H.-N.; Zheng, Z.; Zhang, Y. Blockchain for Internet of Things: A Survey. *IEEE Internet Things J.* **2019**, *6*, 8076–8094. [[CrossRef](#)]
2. Liang, W.; Xie, S.; Long, J.; Li, K.-C.; Zhang, D.; Li, K. A Double PUF-based RFID Identity Authentication Protocol in Service-Centric Internet of Things Environments. *Inf. Sci.* **2019**, *503*, 129–147. [[CrossRef](#)]
3. Uttarkar, R.; Kulkarni, R. Internet of things: Architecture and security. *Int. J. Comput. Appl.* **2014**, *3*, 12–19.
4. Karygiannis, T.; Eydt, B.; Barber, G.; Bunn, L.; Phillips, T. Guidelines for securing radio frequency identification (RFID) systems. *Nist Spec. Publ.* **2007**, *80*, 1–154.
5. Zhao, K.; Ge, L. A survey on the internet of things security. In Proceedings of the 2013 Ninth International Conference on Computational Intelligence and Security, Emei Mountain, China, 14–15 December 2013; IEEE: Piscataway, NJ, USA, 2013.
6. Bharathi, M.V.; Tanguturi, R.C.; Jayakumar, C.; Selvamani, K. Node capture attack in Wireless Sensor Network: A survey. In Proceedings of the 2012 IEEE International Conference on Computational Intelligence and Computing Research, Coimbatore, India, 18–20 December 2012; IEEE: Piscataway, NJ, USA, 2012.
7. Mo, Y.; Sinopoli, B. Secure control against replay attacks. In Proceedings of the 2009 47th Annual Allerton Conference on Communication, Control, and Computing (Allerton), Monticello, IL, USA, 30 September–2 October 2009; IEEE: Piscataway, NJ, USA, 2009.
8. Yang, B.; Wu, K.; Karri, R. Scan based side channel attack on dedicated hardware implementations of data encryption standard. In Proceedings of the 2004 International Conference on Test, Charlotte, NC, USA, 26–28 October 2004; IEEE: Piscataway, NJ, USA, 2004.
9. Babar, S.; Stango, A.; Prasad, N.; Sen, J.; Prasad, R. Proposed embedded security framework for Internet of Things (IoT). In Proceedings of the 2011 2nd International Conference on Wireless Communication, Vehicular Technology, Information Theory and Aerospace & Electronic Systems Technology (Wireless VITAE), Chennai, India, 28 February–3 March 2011; IEEE: Piscataway, NJ, USA, 2011.
10. Pacheco, J.; Hariri, S. IoT security framework for smart cyber infrastructures. In Proceedings of the 2016 IEEE 1st International Workshops on Foundations and Applications of Self* Systems (FAS* W), Augsburg, Germany, 12–16 September 2016; IEEE: Piscataway, NJ, USA, 2016.
11. Huang, X.; Craig, P.; Lin, H.; Yan, Z. SecIoT: A security framework for the Internet of Things. *Secur. Commun. Netw.* **2016**, *9*, 3083–3094. [[CrossRef](#)]
12. Kalra, S.; Sood, S.K. Secure authentication scheme for IoT and cloud servers. *Pervasive Mob. Comput.* **2015**, *24*, 210–223. [[CrossRef](#)]
13. Zhou, L.; Chao, H.C. Multimedia traffic security architecture for the internet of things. *IEEE Netw.* **2011**, *25*, 35–40. [[CrossRef](#)]
14. Tao, M.; Zuo, J.; Liu, Z.; Castiglione, A.; Palmieri, F. Multi-layer cloud architectural model and ontology-based security service framework for IoT-based smart homes. *Future Gener. Comput. Syst.* **2018**, *78*, 1040–1051. [[CrossRef](#)]

15. Kang, W.M.; Moon, S.Y.; Park, J.H. An enhanced security framework for home appliances in smart home. *Hum. Cent. Comput. Inf. Sci.* **2017**, *7*, 6. [[CrossRef](#)]
16. Meidan, Y.; Bohadana, M.; Shabtai, A.; Guarnizo, J.D.; Ochoa, M.; Tippenhauer, N.O.; Elovici, Y. ProfilIoT: A machine learning approach for IoT device identification based on network traffic analysis. In Proceedings of the Symposium on Applied Computing 2017, Marrakech, Morocco, 3–7 April 2017; ACM: New York, NY, USA, 2017.
17. Atzori, L.; Iera, A.; Morabito, G.; Nitti, M. The social internet of things (siot)—when social networks meet the internet of things: Concept, architecture and network characterization. *Comput. Netw.* **2012**, *56*, 3594–3608. [[CrossRef](#)]
18. Asokan, N.; Ekberg, J.E.; Kostiaainen, K.; Rajan, A.; Rozas, C.; Sadeghi, A.R.; Wachsmann, C. Mobile trusted computing. *Proc. IEEE* **2014**, *102*, 1189–1206. [[CrossRef](#)]
19. Sabt, M.; Mohammed, A.; Abdelmajid, B. Trusted execution environment: what it is, and what it is not. In Proceedings of the 2015 IEEE Trustcom/BigDataSE/ISPA, Helsinki, Finland, 20–22 August 2015; IEEE: Piscataway, NJ, USA, 2015; Volume 1.
20. Maji, H.K.; Prabhakaran, M.; Rosulek, M. Attribute-based signatures. In Proceedings of the Cryptographers' Track at the RSA Conference, San Francisco, CA, USA, 27 February–2 March 2011; Springer: Berlin/Heidelberg, Germany, 2011.
21. Maji, H.K.; Prabhakaran, M.; Rosulek, M. Attribute-Based Signatures: Achieving Attribute-Privacy and Collusion-Resistance. *IACR Cryptol. EPrint Arch.* **2008**, *2008*, 328.
22. Ye, J.; Wang, J.; Zhao, J.; Shen, J.; Li, K.-C. Fine-grained searchable encryption in multi-user setting. *Soft Comput.* **2017**, *21*, 6201–6212. [[CrossRef](#)]
23. Liang, W.; Tang, M.; Long, J.; Peng, X.; Xu, J.; Li, K.-C. A Secure Fabric Blockchain-based Data Transmission Technique for Industrial Internet-of-Things. *IEEE Trans. Ind. Inform.* **2019**, *15*, 3582–3592. [[CrossRef](#)]
24. Atzori, L.; Iera, A.; Morabito, G. The internet of things: A survey. *Comput. Netw.* **2010**, *54*, 2787–2805. [[CrossRef](#)]
25. Buckl, C.; Sommer, S.; Scholz, A.; Knoll, A.; Kemper, A.; Heuer, J.; Schmitt, A. Services to the field: An approach for resource-constrained sensor/actor networks. In Proceedings of the 2009 International Conference on Advanced Information Networking and Applications Workshops, Bradford, UK, 26–29 May 2009; IEEE: Piscataway, NJ, USA, 2009.
26. Lin, J.; Yu, W.; Yang, X.; Yang, Q.; Fu, X.; Zhao, W. A novel dynamic en-route decision real-time route guidance scheme in intelligent transportation systems. In Proceedings of the 2015 IEEE 35th International Conference on Distributed Computing Systems, Columbus, OH, USA, 29 June–2 July 2015; IEEE: Piscataway, NJ, USA, 2015.
27. Kim, R.; Lim, H.; Krishnamachari, B. Prefetching-based data dissemination in vehicular cloud systems. *IEEE Trans. Veh. Technol.* **2015**, *65*, 292–306. [[CrossRef](#)]
28. Khanjary, M.; Hashemi, S.M. Route guidance systems: Review and classification. In Proceedings of the 2012 6th Euro American Conference on Telematics and Information Systems (EATIS), Valencia, Spain, 23–25 May 2012; IEEE: Piscataway, NJ, USA, 2012.
29. Liang, W.; Fan, Y.; Li, K.-C.; Zhang, D.; Gaudiot, J.-L. Secure Data Storage and Recovery in Industrial Blockchain Network Environments. *IEEE Trans. Ind. Inform.* **2020**. [[CrossRef](#)]
30. Azadegan, S.; Yu, W.; Liu, H.; Sistani, M.; Acharya, S. Novel anti-forensics approaches for smartphones. In Proceedings of the 2012 45th Hawaii International Conference on System Sciences, Maui, HI, USA, 4–7 January 2012; IEEE: Piscataway, NJ, USA, 2012.
31. Yang, Q.; Liu, Y.; Yu, W.; An, D.; Yang, X.; Lin, J. On data integrity attacks against optimal power flow in power grid systems. In Proceedings of the 2017 14th IEEE Annual Consumer Communications & Networking Conference (CCNC), Las Vegas, NV, USA, 8–11 January 2017; IEEE: Piscataway, NJ, USA, 2017.

