Secure Bootstrapping and Rebootstrapping for Resource-Constrained Thing in Internet of Things

Seung Wook Jung and Souhwan Jung

School of Electronic Engineering Soongsil University, 369 Sangdo-ro, Dongjak-gu, Seoul 156-743, Republic of Korea

Correspondence should be addressed to Souhwan Jung; souhwanj@ssu.ac.kr

Received 9 January 2015; Revised 27 March 2015; Accepted 27 March 2015

Copyright © 2015 S. W. Jung and S. Jung. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

In Internet of Things, secure key establishment and building trust relationship between the thing and the home gateway (or the controller) in home network or Body Area Network are extremely important. Without the guarantee of establishment of key and trust relationship, the traffic over the Internet of Things network cannot be presumed secure. Also, when the home gateway, which knows the shared secret key, is out of order and the new gateway should be installed, the secure key establishment and building trust relationship are also very important because this scenario is likely to happen in real world. In addition, the thing is resource-constrained device, so it is difficult when secure key establishment and building trust relationship involve Public Key Infrastructure.

To address these problems, this paper presents user-friendly bootstrapping and rebootstrapping schemes in the context of Internet of Thing using the human memorable password. The proposed schemes do not require a special hardware module on the thing or special equipment for bootstrapping and rebootstrapping. All these properties make the proposed scheme cost-effective and easy to use.

1. Introduction

Internet of Things (IoT) is a promising technology, but many security problems are still unsolved. The “thing” is usually resource-constrained mote-class device without user interface such as keyboard or screen; the things may have a LED and a switch. Also, the network is assumed to be the lossy network with limited throughput, for example, 10 kb/s. Furthermore, in such limited environment, the mote-class thing has limited power and network to conduct the Public Key Infrastructure (PKI) [1] to establish trust between entities.

Some researches provide a bootstrapping scheme [2] and initial-key reconfiguration scheme [3] to securely share the secret key between the thing and the controller in such limited environment. However, they do not consider the situation where the home gateway or the smartphone should be replaced. The replacement is likely to happen in real world, when the smartphone and the home gateway get old or broken. In this situation, the thing and new home gateway or new smartphone cannot build the trust relationship and they do not make it easy to share securely the secret key without PKI. The worst case is that every device (the smartphone, the home gateway, and storage such as USB memory) is out of order at the same time. For considering the worst case, any long random secret values such as a random secret key or a private key to make trust relationship between the thing and new home gateway or new smartphone cannot be stored in any device (the smartphone, the home gateway, or storage), because restoring the random value is impossible and the random values in the thing cannot be sent securely without PKI.

The problem of PKI for the mote-class thing is too complex which includes X.509 certificates validation in the certificate chain and checking certificate revocation list. The reason why we have to do these complex tasks is that the private key in the smartphone and the home gateway is lost and new certificate must be used in the worst case.

One good alternative is using the human memorable password as a secret key for making trust relationship. The password does not need to be stored in the smartphone, the home gateway, or storage.
This paper proposes a new bootstrapping and rebootstrapping scheme for the mote-class thing in considering that the home gateway or the smartphone is replaced without using asymmetric cryptographic algorithms which takes longer time and consumes the battery to execute than the symmetric cryptographic algorithms. At the end of bootstrapping and rebootstrapping, the thing and the home gateway or the smartphone shares securely a shared secret key as a security association.

The rest of the paper is organized as follows. Section 2 discusses related works. In Section 3, we define assumptions, problems, and attack model. In Section 4, we review password-based key exchange protocols and modify those for bootstrapping and rebootstrapping of IoT devices. Section 5 proposes our scheme. Section 6 analyzes the security and the trust relationship. Finally, Section 7 concludes the paper.

2. Related Works

Recent researches focus on the secure key sharing between the things and the home gateway [2] or initial-key reconfiguration [3]. These researches build the trust relationship between the things and the home gateway.

However, these researches do not consider how to conduct securely rebootstrapping procedure in the case that the home gateway should be replaced by upgrading the home gateway hardware or having a problem of the home gateway. For example, [2] shares DevSecret, which is a shared secret, between the home gateway and the things by bootstrapping and the user loses or removes securely QR code. In this case, when the home gateway is out of order, DevSecret is only known to the thing which has no user interface to show DevSecret. Thus, the user has no means to transport DevSecret to the home gateway securely through insecure channel. One possible way to share DevSecret between the things and new home gateway is transporting the DevSecret through asymmetric cryptographic algorithms after building trust relationship between the thing and the home gateway using PKI. However, for the resource-constrained mote-class things, the PKI is too heavy. Therefore, the trust relationship between the home gateway and the thing cannot be easily built.

Reference [3] reconfigures the initial-key securely and at the end of reconfiguring the things and the home gateway shares a preshared key (PSK). However, if the home gateway is out of order, there is no means to transport PSK to new home gateway as [2] does.

This paper considers the human memorable password as a good alternative, so this paper is also related to the password-based secure key exchange (PBEKE). The starting research was PBEKE protocol of Bellovin and Merritt [4]. The general idea of PBEKE is to transmit ephemeral public keys (Diffie-Hellman public value) encrypted using the password as a shared key without security proof. After [4], a lot of PBEKE research papers are published.

There are a lot of secure Diffie-Hellman-based PBEKEs [5–7] with the security proof. After that, augmented PBEKEs [5, 8–11] are emerged. The augmented PBEKEs store an image of the password, for example, the hash value of the password, instead of using password itself. In the worst case, any value should not be stored in the device such as the smartphone. Moreover, usually the augmented PBEKE requires more computations than the nonaugmented PBEKE. For the mote-class thing, the number of computations at the thing is important. Therefore, this paper excludes the augmented PBEKE as a candidate of PBEKEs for the bootstrapping and rebootstrapping of the things.

Although most PBEKEs rely on the Diffie-Hellman key exchange, it is not surprising that the RSA-based PBEKEs [4, 12] are researched. In general, RSA-based PBEKEs do not support forward secrecy; if the private key is compromised, old session keys can be recovered for all sessions. Therefore, this paper excludes the RSA-based PBEKEs as a candidate of PBEKEs for the bootstrapping and rebootstrapping of the things. Nevertheless, in the case where only RSA-based asymmetric algorithms are installed in the thing, RSA-based PBEKE can be used for the bootstrapping and rebootstrapping of things.

There are also PBEKEs using a server public key [13–16]. These require storing the long random private at the smartphone or the home gateway. Because this paper considers the worst case where all devices are out of order except the thing, PBEKE using a server public key cannot be a candidate of bootstrapping and rebootstrapping of things.

3. Environment, Problem Definition, and Attacker Model

3.1. Environment

Environment 1. The first considered environment in this paper is a home network as shown in Figure 1. The things, the home gateway, and the smartphone are located in the home. The things such as the sensor or actuator have constrained computing power, small battery, and limited amounts of memory, and limited computing power. Also, mote-class things have no user interface such as keyboard or screens; they may have only a single button or LED as a user interface. The mote-class thing also does not have a port for wired network, so they communicate with the other entities through wireless network. This wireless throughput is also limited and lossy network [17]. For example, 6LoWPAN [17] considers high packet error rate and a typical throughput of...
10 kbps. We assume that the things can communicate with entities outside the home through the home gateway. In Figure 1, the dotted line is wireless network and the full line means the wired network.

We assume that the smartphone and the home gateway have trust relationship through secure TLS using X.509 certificate. Here, secure TLS means that TLS use secure cryptographic primitives which do not have secure flaws. The home gateway and the server outside of the home such as the cloud server have a security association.

Environment 2. This paper also considers the Body Area Network (BAN) such as in Figure 2. The difference from environment 1 is only that the smartphone has the role of the home gateway in environment 1. That is, the smartphone communicates with the thing, collects sensing information from the things, controls the things, and works as proxy outside of the home. The smartphone and the server outside of the home communicate through secure TLS if it is necessary. Other environment properties are the same as environment 1.

3.2. Problem Definition. This paper considers how to transport the secret and to build trust relationship between the thing and the home gateway, even if every device is out of order including the home gateway in home network, the smartphone in BAN, and storage such as USB memory. For the worst case where every device except the things is out of order at the same time, we cannot store a long random key. Without the PKI, to make trust relationship between the thing and other devices and to share a secret, this paper considers the human memorable password as good alternative. This paper does not consider the situation where the thing is out of order and should be replaced, because this problem is reduced to the bootstrapping problem.

The main considerations of this paper are how to securely bootstrap and rebootstrap between the thing and the home gateway or the smartphone using the human memorable password in the worst case.

The secure bootstrapping and rebootstrapping scheme based on the human memorable password should satisfy the following security properties.

(i) It has to be secure against the dictionary attack or guessing attack.
(ii) Key secrecy: an attacker has a negligible probability of compromising the shared secret.
(iii) Key authenticity: the thing and the gateway share unaltered key.
(iv) Robust to user error: a system should be intuitive for nonexperts.
(v) Cost effective: secure bootstrapping and rebootstrapping of things require additional hardware on each node or special setup hardware [18]. Proposed scheme should not require additional hardware on each thing or special setup hardware.

3.3. Attacker Model. The goal of the attacker is to compromise the key shared by the thing and the home gateway or the smartphone. The communication between the thing and the home gateway is over a wireless medium, and this paper adopts the Dolev-Yao attack model, where an attacker can eavesdrop, intercept, modify, or inject messages into the wireless communication [19]. Moreover, this paper assumes that an attacker uses the more powerful device such as the PC or the server. Also, the attacker pretends the smartphone or the home gateway in order to communicate with the thing.

3.4. Notation. In this paper, the following notation will be used.

(i) $A$: the identifier of the thing,
(ii) $B$: the identifier of the owner of the thing,
(iii) $\pi$: a password,
(iv) $p$: a large safe prime (usually at least 1024 bits),
(v) $q$: a prime with $q \mid p - 1$,
(vi) $G$: a subgroup of $\mathbb{Z}_p^*$ with order $q$,
(vii) $g$: a generator of $G$,
(viii) $r_A$, $r_B$: random integers of the same size as the order of $G$, chosen by $A$ and $B$, respectively,
(ix) $n_A$, $n_B$: nonces of $A$ and $B$, respectively,
(x) $t_A$, $t_B$: public Diffie-Hellman values of $A$ and $B$, respectively,
(xi) $Z_{AB}$: the shared secret calculated by the principals,
(xii) $K_{AB}$: the derived session key,
(xiii) IK: initial-key,
(xiv) PSK: the randomly preshared key,
(xv) $H_1(\cdot)$: one-way hash function with the size of the result value longer than 1024 + 160 bits,
(xvi) $H_2(\cdot)$ and $H_3(\cdot)$: one-way hash function with the size of the result value as 160 bits,
(xvii) $E_k(\cdot)$: the encryption function with the secret key $k$.

We will often omit “mod $p$” from expressions when it is obvious that we are working in $Z_p^*$.

4. Password-Based Encrypted Key Exchange

4.1. Insecure Bootstrapping Method. The mote-class thing has no input interface and output interface, so to share installed initial key with the home gateway or the smartphone, we use QR code as [2] does. QR code can be packed in the package of the thing, so the employee of the company who makes the thing can see the QR code. Therefore, the initial-key should be changed as [3] does. However, while the thing is shipped to the user, the transporter cannot read QR code, because of shielding in the package.

References [2, 3] consider only the bootstrapping and the rebootstrapping with no consideration of the worst case where all devices are out of order except the thing.

For the successful secure rebootstrapping, we cannot store a long random key in the devices which are out of order in the worst case. Also, we do not consider using PKI as discussed in the problem definition section. Therefore, one good solution for secure rebootstrapping in the worst case is using the human memorable password as a secret key.

One insecure method is using password as a secret key for encrypting any message. Usually the message can contain a known plaintext such as on/off message to turn on or turn off the thing. Therefore, it is insecure against the dictionary attack.

The other way is that a random rebootstrapping key is encrypted by the password. The thing stores the rebootstrapping key $k$ and encrypts rebootstrapping key $E_{n}(k)$. During the rebootstrapping procedure, the thing sends the encrypted rebootstrapping key to the smartphone or the home gateway. Then, the user types the password and decrypts the rebootstrapping key. After that, the smartphone and the thing use the rebootstrapping key for encrypting any message between them. If the message contains any known plain text or verifiable values, the attacker can conduct the dictionary attack. The attacker guesses $n'$ and decrypts $k$ with $n'$. After that, the attacker decrypts message, which contains a known plain text or verifiable value, with $k$. Then the attacker checks the known plain text or verifiable values. If true, then the guessed password and rebootstrapping key are right and the dictionary attack is succeeded. Figure 3 is an example. For key confirmation and authenticity, the thing sends encrypted nonce $n_A$, and $n_A + n_B$. The smartphone sends encrypted nonce $n_A$. In this case, the attacker guesses a password $n'$ and decrypts $k'$. Then the attacker decrypts $n'_A, n'_B, A'$ with the $k'$ and checks $n'_A + n'_B = A'$. If true, then the guessed password is right and the dictionary attack succeeds.

The solution for the secure rebootstrapping with the password as a secret key is reduced to the password-based encrypted key exchange (PBEKE) problem. There are many well-known PBEKEs of which security is proved.

4.2. Modification Method of PBEKE for Secure Bootstrapping and Rebootstrapping. In this section, we show how to modify the PBEKE for secure bootstrapping and rebootstrapping of the resource-constrained things. As an example, this paper selects Password Protected Key (PPK) exchange [5] as shown in Figure 4. For more details of PPK, see [5].

Assume that $A$ is a thing and $B$ is a smartphone. For the mote-class thing, the computations at the thing must be minimized. For that, $P_1, P_2, r_A, t_A$, and $m$ can be computed at the smartphone and sent to the thing with public parameters, $q, r, g$, and $g$, through secure channel. In this paper, we call these values precomputed values. For the rebootstrapping, the thing sends original message $m$ and public parameters to the smartphone. The smartphone computes the computations of original version of PBEKE, new $r_A''$, and new $m'' = g^{r_A''}P_1$ online, and sends $m'$, the message of original version of PBEKE and encrypted $(r_A'', m'' = g^{r_A''}P_1)$ using $K_{AB}$ to the thing. $r_A''$ and $m''$ will be used for the next rebootstrapping. The thing computes $t_B, Z_{AB}$, and $K_{AB}$ online. Therefore, the
thing needs only to compute one exponentiation computation which is minimal computation.

In sum, the precomputed values are computed at the smartphone and are sent to the thing securely. The message from the thing which is required by original PBKEKs and the public parameters are sent to the smartphone for rebootstrapping. Also, computations at the smartphone and the thing which are specified by the original PBEKE are conducted at each side. As a result, the thing and the smartphone can share a secret key securely with the minimal computation at the thing. This is a general modification method of PBEKE for the bootstrapping and rebootstrapping for the resource-constrained thing.

Table 1 shows the comparison of the original PPK and modified PPK. Basically, the rebootstrapping protocol of modified PPK is the same as original PPK except sending encrypted new \( r''_A \) and new \( m'' \) at the end of the original PPK for the next rebootstrapping. For rebootstrapping, at the bootstrapping in the modified PPK B computes \( P_1, P_2, r_A, t_A, m \) and a PSK (preshared key) and encrypts these with initial-key in the thing. After that, B sends the encrypted values to A and A keeps the encrypted values which will be used in the rebootstrapping protocol.

### 4.3. Comparison of PBKEKEs

This paper excludes augmented PBKEKE, RSA-based PBKEKE, and PBKEKE using a server public key as described in Section 2. Therefore, this paper compares Diffie-Hellman-based PBKEKE for our purpose in this section.

Table 2 shows the comparison of original PBKEKEs of which security is proven.

In IoT environment, the network is considered LLN (low power and lossy network). So minimizing the number of messages is important. PPK has good property that is the smallest number of messages. In the case where the device is much more powerful than the thing, the computation load at the device is not so important, so we select PPK to make full version of secure bootstrapping and rebootstrapping for the resource-constrained thing which is presented at the next section.

### 5. Secure Bootstrapping and Rebootstrapping for Resource-Constrained Thing

This section shows the full version of bootstrapping and rebootstrapping scheme for the mote-class thing based on PPK to which the modification method of PBEKE is applied as an example. Any secure PBEKE can be used as the basis of bootstrapping and rebootstrapping scheme for the resource-constrained thing according to the requirements of the environment, for example, the cryptographic primitives installed in the thing.

#### 5.1. Secure Bootstrapping Based on QR Code

As we discussed, to share installed initial-key between the thing and the home gateway, we use QR code as [2] does. The initial-key should be changed to preshared secret key (PSK) as [3] does.

Figure 5 depicts the bootstrapping procedure including the enrollment procedure and initial-key reconfiguration procedure for environment 1.

1. Initially, a thing stores an initial-key (IK) and a random identifier (A), and QR code which includes IK and A.
2. The user opens the package to get the thing and the QR code. The user reads the QR code with his/her smartphone.
3. The smartphone sends IK and A to a home gateway through secure TLS.
4. After the home gateway and the thing start pairing, then the thing selects a random number \( n \) and encrypts it with IK and sends \( E_{IK}(A, n_A) \) to the home gateway.
5. The home gateway decrypts \( E_{IK}(A, n_A) \) and checks whether the A is correct. If the A is correct, then home gateway requests precomputed values \( (p, q, r, g, P_1, P_2, r_A, \text{ and } t_A P_1) \) where \( p = 1 = qr \mod p \). The user types his identifier (B) and the password and then the smartphone computes precomputed values and sends the precomputed values to the home gateway through secure TLS.
Table 2: Comparison of original PBEKEs.

|               | Number of messages | Number of exponentiations of this size at the thing |
|---------------|--------------------|-----------------------------------------------------|
|               | 160 bits | 864 bits | 1024 bits |
| PAK [5]       | 3        | 2        | 1         |
| PPK [5]       | 2        | 2        | 2         |
| PAK-R [6]     | 3        | 3        |           |
| SPEKE [21]    | 3        |          | 2         |

Figure 5: Secure bootstrapping with the home gateway.

(1) A thing initially stores IK and A, and a QR code includes IK and A.

(2) The user opens the package of the thing and gets the QR code. Then, the smartphone reads the QR code to get IK and A.

(3) The smartphone and the thing start pairing.

(4) The thing selects a random number $n_A$ and sends encrypted $E_{IK}(A, n_A)$.

(5) The smartphone decrypts $E_{IK}(A, n_A)$ with IK and checks whether $A$ is correct. Then, the user types the password and his identity $B$ into the smartphone. The smartphone computes precomputed values ($p, q, r, g, P_1, P_2, r_A,$ and $t_A P_1$).

(6) The smartphone sends $E_{PSK}(n_A, n_B)$ to the thing.

(7) The thing decrypts $E_{IK}(PSK, n_B, n_A, p, q, r, P_1, P_2, r_A, t_A P_1)$ and checks whether $n_A$ and $n_B$ are correct. If correct, the thing is authenticated by the smartphone and stores PSK.

(8) The smartphone decrypts $E_{PSK}(n_A, n_B)$ and checks whether $n_A$ and $n_B$ are correct. If correct, the thing is authenticated by the smartphone and stores PSK.

(9) After that, the thing and the smartphone can communicate securely with PSK.

Also, this bootstrapping for environment 2 uses $E_{IK}(\cdot)$ as a secure channel to deliver the precomputed values.
5.2. Rebootstrapping for the Worst Case. Figure 7 shows rebootstrapping procedure for environment 1 for the worst case.

(1) The thing stores precomputed values \((p, q, r, g, P_1, P_2, r_A, t_A P_1)\) as the result of bootstrapping.

(2) The thing and the home gateway start pairing, and the home gateway sends the rebootstrapping request to the thing.

(3) The thing sends \(A, g, p, q, r, m = t_A P_1\) to the home gateway.

(4) The home gateway forwards these to the smartphone through secure TLS, and the user types the password and his identity \(B\).

(5) The smartphone computes \(t_A = m/P_1\) and selects random numbers \(r_B, r'_A \in Z_q\). The smartphone computes \(t_B = g^{r_B}, t'_A = g^{r'_A}, t_A P_1, Z_{AB} = t'_A, m = t'_A P_1\), and \(m' = t_B P_2\).

(6) The smartphone sends \(B, r'_A, P_1, m', t'_A P_1, Z_{AB}\) to the home gateway through a secure channel.

(7) The home gateway computes \(K_{AB} = H_3(A, B, m, m', Z_{AB}, P_1)\) and sends \(m', B, E_{K_{AB}}(r'_A, t'_A P_1)\) to the thing.

(8) The thing computes \(t_B = m'/P_2, Z_{AB} = t_B, m = t'_B P_2\), and \(K_{AB} = H_3(A, B, m, m', Z_{AB}, P_1)\) and stores \(K_{AB}\).

(9) The thing stores \(r'_A, t'_A P_1\) for the next rebootstrapping.
The thing and the home gateway share $K_{AB}$ as a secret key. Therefore, subsequent message can be encrypted by this key if the security of message is necessary.

The thing can connect to the smartphone and send $A, g, p, q, r, m, A, p, q, r$ directly to the smartphone, if it is more efficient than the above proposed rebootstrapping protocol.

Figure 8 shows bootstrapping procedure for environment 2 in the worst case.

1. The thing has precomputed values $(p, q, r, g, P_1, r_A^t, t_A^t P_1)$.
2. The thing and the smartphone start pairing, and the smartphone sends the rebootstrapping request to the thing.
3. The thing sends $A, m, g, p, q, r$ to the smartphone.
4. The user types the password and his identity $B$ into the smartphone. The smartphone selects random numbers $r_B, r_A^t \in Z_q$, and computes $t_A = m/P_1, t_B = g^r, t_A^t = g^{r_A^t}, t_A^t P_1, t_A^t P_1 = t_B P_2, Z_{AB} = t_A^t P_2$, and $K_{AB} = H_5(A,B,m,m',Z_{AB},P_1)$.
5. The smartphone sends $B, m', E_{K_{AB}}(r_A^t, t_A^t P_1)$ to the thing.
6. The thing computes $Z_{AB} = (m'/P_2)^{r_B}$ and decrypts $E_{K_{AB}}(r_A^t, t_A^t P_1)$. The thing stores $Z_{AB}, r_A^t, t_A^t P_1$. $K_{AB}$ is used for the shared secret key, and $r_A^t$ and $t_A^t P_1$ will be used for the next rebootstrapping.
7. The $K_{AB}$ will be used for encrypting any message between the thing and the smartphone as the shared secret.

In our bootstrapping and rebootstrapping schemes we require only one exponentiation at the mote-class thing. It is minimized computation load at the mote-class thing.

6. Security Analysis

6.1. The Security of Bootstrapping and Rebootstrapping Method. The PPK protocol was proven by MacKenzie [20] to be secure Shoup’s simulation model assuming the hash functions act as random oracle. In summary, the proof consists of constructing a simulator for a real system so that the transcript of an adversary attacking the simulator is computationally indistinguishable from the transcript of an adversary attacking the real system. Finally the paper [16] shows that if the adversary breaks the PPK protocol with nonnegligible probability, then we can break the DDH (Decision Diffie-Hellman) problem with nonnegligible probability. DDH problem is hard problem, so breaking PPK is infeasible. For more detail of the security proof, see [20].

What we have changed from the PPK are the following.

1. The precomputed values $(p, q, r, g, P_1, P_2, r_A^t, t_A^t P_1)$ are generated in the $B$ instead of $A$. It does not harm the security of PPK if $B$ is trusted.

2. The precomputed values, PSK, $n_A$ and $n_B$, are encrypted with IK and sent to the thing. PSK and IK are a long random number (typically 128 bits) and if the encryption algorithm such as AES is secure, getting $r_A$ is infeasible. Therefore, it does not harm the security of PPK.

3. In the rebootstrapping procedure, $g, p, q, r, m, A, B, m, m', Z_{AB}, P_1$ are sent through insecure channel. Original PPK sends $m$ through insecure channel, and $g, p, q, r$ are public values. Therefore, it does not harm the security of PPK.

4. Communication channel between the smartphone and the home gateway must be secure channel such as secure TLS. Therefore, the values transmitted between smartphone and the home gateway through secure channel are protected from the attacker. Therefore, it does not harm the security of PPK.

5. At the end of rebootstrapping procedure, $r_A^t$ and $t_A^t P_1$ are encrypted with $K_{AB}$ and sent to the thing from the home gateway. $K_{AB}$ is a long random number and if the encryption algorithm is secure, it does not harm the security of PPK.

Therefore, our proposed bootstrapping and rebootstrapping protocols are secure under the same line of the security of PPK. Also, PPK is secure against dictionary attack and brute-force attack on $t_A^t P_1$ and $t_B^t P_2$ under DDH assumption. Our
protocols also use the same form of $t_A P_1$ and $t_B P_2$, and we use a long random secret key for encrypting other values, so if the encryption algorithm is secure such as AES, our scheme is also secure against the dictionary attack and the brute-force attack.

6.2. Trust Relationship. In environment 1, the thing trusts the smartphone whose owner opens the package of the thing. The home gateway is trusted by the thing, because the smartphone has a trust relationship with the home gateway using PKI and possesses IK. After bootstrapping procedure, the thing and the home gateway share a secret key, which means these two entities have a security association. During rebootstrapping procedure, the thing will trust who can decrypt $t_A P_1$ with the password, that is, the smartphone which gets the password from the owner of the thing. If the home gateway has a trust relationship with the smartphone, the home gateway will be able to get $Z_{AB}$. Finally, the thing will trust the home gateway that has $Z_{AB}$.

In environment 2, the thing will trust the smartphone whose owner opens the package of the thing. Finally, the thing and the smartphone share PSK as a security association. For rebootstrapping, the thing will trust who can decrypt $t_A P_1$ with the password. Only legitimate smartphone will be able to decrypt it.

6.3. Fake Smartphone and Fake Home Gateway. For bootstrapping procedure, the only legitimate smartphone will read QR code. The smartphone and the home gateway have trust relationship by PKI using secure TLS. Therefore, the proposed scheme protects from the fake home gateway during bootstrapping.

For rebootstrapping procedure, the fake home gateway in environment 1 and the fake smartphone in environment 2 can request $t_A P_1$, so the problem reduces to knowing the password. For that, the attacker conducts the dictionary attack and the brute-force attack on the password. After dictionary attack or the brute-force attack, the attacker gets the Diffie-Hellman public value which looks like a random value, but the attacker cannot know the Diffie-Hellman private value of the thing. Therefore, the attacker has to solve Decisional Diffie-Hellman problem which is hard problem.

6.4. Other Security Properties

(i) Key secrecy: secure PBEXE provides the key secrecy, but the initial-key can be known to the manufacturer, so the initial-key reconfiguration is important.

(ii) Key authenticity: PBEXE provides key authenticity.

(iii) Robust to user error: the user reads QR code using the smartphone and just types the password, so the proposed scheme is not error-prone.

(iv) Cost effective: the proposed scheme does not require a special hardware module in the thing and special equipment for bootstrapping and rebootstrapping.

7. Conclusion

This paper considered the worst case where every device is out of order except the thing at the same time and no PKI. For this situation, this paper proposed using the human memorable password to make trust relationship between the thing and the home gateway or the smartphone. For that, this paper provided modification method based on the well-known password-based encrypted key exchange protocol to make secure bootstrapping and rebootstrapping scheme. This paper selected PPK as a good candidate for the bootstrapping and rebootstrapping scheme. This paper proposed the secure bootstrapping scheme and rebootstrapping scheme based on PPK. One of the main contributions of this paper is providing the means to rebootstrapping when the home gateway or the smartphone which shares the secret key is out of order and should be replaced by new one. At the end of the rebootstrapping, the thing and the home gateway or the smartphone shares the secret key securely without using PKI.

This paper also analyzes the trust relationship and the security of the proposed scheme. The security properties of the proposed scheme include key secrecy and key authenticity. The proposed scheme does not require a special hardware module on the thing or special equipment for bootstrapping or rebootstrapping. These characteristics make the proposed scheme cost-effective, usable, and secure for commodity things.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This research was supported by the MSIP (Ministry of Science, ICT and Future Planning), Korea, under the ITRC (Information Technology Research Center) Support Program (IITP-2015-H8501-15-1008) supervised by the IITP (Institute for Information & Communications Technology Promotion). This work was supported by the ICT R&D Program of MSIP/IITP, Republic of Korea (10041671, Enterprise Wireless LAN Solution (AP and AP controller)).

References

[1] D. Solo, R. Housley, and W. Ford, “Internet X.509 public key infrastructure certificate and CRL profile,” 1999.
[2] C. Jennings, “Transitive Trust Enrollment for Constrained Devices,” 2012.
[3] N. Kang, S.-H. Oh, and S. Yoon, “Secure initial-key reconfiguration for resource constrained devices,” IETF Draft, 2014.
[4] S. M. Bellovin and M. Merritt, “Encrypted key exchange: password-based protocols secure against dictionary attack,” in Proceedings of the IEEE Symposium on Research in Security and Privacy, pp. 72–84, 1992.
[5] V. Boyko, P. MacKenzie, and S. Patel, “Provably secure password-authenticated key exchange using Diffie-Hellman,” in Advances in Cryptology—EUROCRYPT 2000, vol. 1807 of
[6] P. MacKenzie, "More efficient password-authenticated key exchange," in Topics in Cryptology—CT-RSA 2001, vol. 2020 of Lecture Notes in Computer Science, pp. 361–377, Springer, Berlin, Germany, 2001.

[7] D. P. Jablon, "Strong password-only authenticated key exchange," ACM SIGCOMM Computer Communication Review, vol. 26, no. 5, pp. 5–26, 1996.

[8] S. M. Bellovin and M. Merritt, "Augmented encrypted key exchange: a password-based protocol secure against dictionary attacks and password file compromise," in Proceedings of the 1st ACM Conference on Computer and Communications Security, pp. 244–250, ACM, November 1993.

[9] D. P. Jablon, "Extended password key exchange protocols immune to dictionary attack," in Proceedings of the 6th IEEE Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, pp. 248–255, IEEE, Cambridge, Mass, USA, June 1997.

[10] T. D. Wu, "The secure remote password protocol," in Proceedings of the Internet Society Network and Distributed System Security Symposium (NDSS ’98), pp. 97–111, 1998.

[11] T. Kwon, Ultimate Solution to Authentication via Memorable Password, Contribution to the IEEE P1363 Study Group, 2000.

[12] P. MacKenzie, S. Patel, and R. Swaminathan, "Password-authenticated key exchange based on RSA," in Advances in Cryptology—ASIACRYPT 2000, vol. 1976 of Lecture Notes in Computer Science, pp. 599–613, Springer, Berlin, Germany, 2000.

[13] L. Gong, M. A. Lomas, R. M. Needham, and J. H. Saltzer, "Protecting poorly chosen secrets from guessing attacks," IEEE Journal on Selected Areas in Communications, vol. 11, no. 5, pp. 648–656, 1993.

[14] T. Kwon and J. Song, "Efficient and secure password-based authentication protocols against guessing attacks," Computer Communications, vol. 21, no. 9, pp. 853–861, 1998.

[15] K. W. O. N. Taekyoung and S. O. N. G. Jooseok, "Efficient key exchange and authentication protocols protecting weak secrets," IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, vol. 81, no. 1, pp. 156–163, 1998.

[16] S. Halevi and H. Krawczyk, "Public-key cryptography and password protocols," ACM Transactions on Information and System Security, vol. 2, no. 3, pp. 230–268, 1999.

[17] N. Kushalnagar, G. Montenegro, and C. Schumacher, "RFC 4919: IPv6 over low-power wireless personal area networks (6LoWPANs): overview," in Assumptions, Problem Statement, and Goals, 2007.

[18] C. Kuo, M. Luk, R. Negi, and A. Perrig, "Message-in-a-bottle: user-friendly and secure key deployment for sensor nodes," in Proceedings of the 5th ACM International Conference on Embedded Networked Sensor Systems (SenSys ’07), pp. 233–246, ACM, November 2007.

[19] D. Dolev and A. C. Yao, "On the security of public key protocols," IEEE Transactions on Information Theory, vol. 29, no. 2, pp. 198–208, 1983.

[20] P. MacKenzie, "The PAK suite: protocols for password-authenticated key exchange," in Proceedings of the IEEE P1363
Submit your manuscripts at
http://www.hindawi.com