TESLA 鍵への ID の埋め込みについて

面和成 †

† 北陸先端科学技術大学院大学 情報科学研究科
〒 923-1292 石川県能美市旭台 1 − 1
omote@jaist.ac.jp

あらまし TESLA は効率的なブロードキャスト認証方式である．この方式はハッシュチェーンを用いており，ノードに埋め込まれているハッシュチェーンの末尾が TESLA 鍵（認証の信頼点）になる．しかしながら，TESLA で認証付き鍵を実現するためには公開鍵インフラに頼らざるを得ない．すなわち，TESLA は署名ベースの認証方式による鍵管理が必要になり，システム全体として効率的とは言い難い．
本稿では，ハッシュ関数のような軽量な演算のみを用い，ID をベースとするように改良した TESLA ブロードキャスト認証方式を提案する．本方式では，各ノードの ROM に埋め込まれている ID がメッセージ認証の信頼点となることを保証する．つまり，攻撃者はこの ID にリンクするような正当なメッセージを偽造できない．さらに，ID が TESLA 鍵の一部に含められることを示す．ゆえに，本方式は ID を追加しても TESLA 鍵のサイズが TESLA とほぼ変わらない方式を実現している．

An identity embedding to the TESLA-key

Kazumasa Omote†

†School of Information Science, Japan Advanced Institute of Science and Technology
1-1, Asahi-dai, Nomi-shi, Ishikawa, 923-1292, Japan
omote@jaist.ac.jp

Abstract TESLA is an efficient broadcast authentication scheme. This is a scheme using the hash chain, and the end of the chain is “TESLA-key” (root of trust) for message authentication, which is embedded in each node and is not a private key. However, TESLA relies on a public key infrastructure that has to be used in order to provide authentic TESLA-key. Hence, TESLA requires the key management by signature-based authentication schemes, and thus it is not necessarily efficient as an entire system.
In this paper, we propose an alternative broadcast authentication scheme based on a simple form of identity-based cryptography using only the lightweight computation such as hash function. Our scheme is guaranteed that the identity embedded in each node is the root of trust for message authentication. In other words, an adversary cannot falsify a legitimate message related to the identity. Furthermore, we show that the identity can be included in TESLA key. The TESLA-key in our scheme is the same size as one in TESLA, even if our scheme newly adds the identity into the TESLA-key.

1 Introduction

Recently, the spread of wireless mobile devices (e.g. mobile phone, RFID and sensor networks) is remarkable. A wireless mobile network is composed of a great number of client nodes and some base stations. It is assumed that each node does not have an expensive CPU power, an abundant storage capacity, a big network bandwidth and is tamper resistant. Each node can be easily captured and compromised by the adversary. Therefore, it is important not to use an authentication based on public key cryptography and not to preserve a private key on the node side when each node conducts an authentication using the broadcast network.

TESLA is an efficient broadcast authentication scheme [1, 2, 3, 4]. This is a scheme using the hash chain, and the end of the chain is “TESLA-key” (root of trust) for message authentication, which is embedded in each node and is not a private key. However, TESLA relies on a public key infrastructure that has to be used in order to provide authentic TESLA-key. Hence, TESLA requires the key management by signature-based authentication schemes, and thus it is not necessarily efficient as an entire system.

Another chain-based scheme is proposed in [5], which uses the chain with the squaring computa-

tion in a finite field ($f(x) = x^2 \mod n$) instead of the hash computation. This scheme has the advantage of enabling to set an enough long chain. However, this enlarges the overhead of the computational complexity, the amount of communication and the storage consumption, because it is based on public key cryptography. Also, the scheme based on a one-time signature such as [6] is proposed. The one-time signature scheme is based on the hash function and uses a one-time signing key. It is more efficient than the signature scheme based on public key cryptography, although the size of key becomes large. Furthermore, the message recognition protocols [7, 8] are proposed, which consider the man-in-the-middle attack in ad hoc network. These protocols have the advantages that time-synchronization is not required and the authentication delay does not occur. However, in these protocols, two-round communications are necessary and each node has to store its private key in its own storage.

The idea of identity-based cryptosystem is proposed by Shamir [9]. This is avoiding the high cost of the public-key management and signature authentication in cryptosystems relying on a public key infrastructure (PKI). Each user can define public key by an arbitrary string. In other words, users may use some well-known information such as email address and IP address as their public key. Thus, there is no need to propagate this common information through the network and to manage public keys. IDHC (ID-based Hash Chain) [10] has an approach related to our scheme, and is based on RSA cryptosystem. Although this scheme is identity-based scheme, it uses the modular operation which requires the large communication load and high computation cost.

In this paper, we propose an alternative broadcast authentication scheme based on a simple form of identity-based cryptography using only the efficient computation such as hash function. Our scheme is guaranteed that the identity embedded in each node is the root of trust for message authentication. In other words, an adversary cannot falsify a legitimate message related to the identity. As a result, our scheme does not require the key management, and is efficient as the entire scheme. Furthermore, we show that the identity can be included in TESLA key. For example, if the length of identity is 80 bits in 160-bit TESLA-key, an adversary can incidentally falsify a message at the probability of $1/2^{80}$. The TESLA-key in our scheme is almost the same size as one in TESLA, even if our scheme newly adds the identity into the TESLA-key.

An example of an application of our scheme is an authenticated software update. In a software update, the center may distribute the update program to each node. It is necessary to prevent an illegal update program from being applied to a node. If we apply our scheme to a software update system, each node can authenticate the update program by regarding the update program as a message.

The rest of this paper is organized as follows. In the next Section 2 we describe preliminaries. We review TESLA in Section 3, propose our lightweight linking algorithm in Section 4, construct our scheme in Section 5, and discuss the security and efficiency analysis of our scheme in Section 6. We finally conclude this paper in Section 7.

2 Preliminaries

2.1 Requirements

We assume that an identity-based TESLA is used in wireless network. The following requirements need to be considered when designing identity-based TESLA.

- **No relying on PKI**. PKI requires the key management by signature-based authentication schemes. On the other hand, a wireless mobile network is composed of a great number of mobile client nodes. Thus, it is better not to rely on PKI in a wireless mobile network.
- **No use the modular operation**. The modular operation requires the large communication load and high computation cost. A mobile node should use the lightweight computation without modular operation, such as hash functions and symmetric-key encryption.

2.2 Security assumptions

Assumption 1 (Secure hash function) *A function H is a cryptographic secure hash function if it satisfies the following properties [11]. We define that λ is a security parameter of H.*

1. *Function H maps bit strings, either of an arbitrary length or a predetermined length, to strings of a fixed length λ ($H : \{0,1\}^* \rightarrow \{0,1\}^\lambda$).*
2. *One-wayness: Given x, it is easy to compute $H(x)$. Conversely, given $H(x)$, it is computationally infeasible to compute x.*

3. *Collision resistance: For any given x, it is computationally infeasible to find y ($\neq x$) such that $H(x) = H(y)$*

Assumption 2 (One-way hash chain) *Select a cryptographic secure hash function H defined by* Assumption 1 *with security parameter λ, $H : \{0,1\}^* \rightarrow \{0,1\}^\lambda$. Pick a seed c_0 randomly and apply H recursively k times to the initial seed c_0 to generate a one-way hash chain, $c_0, c_1, \ldots, c_k$ ($c_i = H(c_{i-1})$, $1 \leq i \leq k$). Note that the one-way hash chain is also denoted as $c_i = H^{i-1}(c_0)$ ($1 \leq i \leq k$).*

One-way hash chain is a widely-used cryptographic primitive. One of the first uses of one-way chain was for one-time passwords by Lamport [12]. Then, the concept of hard core predicate is refined by Goldreich and Levin [13].

Assumption 3 (Hard core predicate) *A hard core predicate of an one-way function $f : \{0,1\}^* \rightarrow \{0,1\}^*$ is a Boolean predicate $B : \{0,1\}^* \rightarrow \{0,1\}$ such that for every probabilistic polynomial-time algorithm $\mathcal{A}'$, every polynomial $p(\cdot)$, all sufficiently large n there is:*

$$Pr[\mathcal{A}'(f(U_n)) = B(U_n)] \leq \frac{1}{2} + \frac{1}{p(n)},$$

where U_n denotes a random variable uniformly distributed over $\{0,1\}^n$.

Assumption 4 (Equal frequency of B) *A bit of the output of B has an equal probability of being a 1 or 0.*

2.3 Entities

- **Base station**. This is a management server of mobile client node, and broadcasts an authenticated message to the node. An authenticated message is same in all nodes.
- **Mobile client node**. A node receives an authenticated message such as an update program from the base station. Each node has its TESLA-key to verify the message.

2.4 Adversary model

The goal of the adversary is to cause each node to accept a false self-serving message. In this attack, for example, the adversary may apply an illegal update program to a node to compromise. We assume that such an attack is implemented using the broadcast communication. Hence these attacks can influence all nodes within the range reached by the broadcast. We also assume "node-capture attack", which makes the node's operation become under the control of the adversary. Of course, the adversary can use the content in the broadcast network and public information. Note that the adversary cannot illegally rewrite data on the storage such as the ROM of each node, although he can read data from such storage. Since each node does not have confidential information such as a private key, the adversary cannot obtain useful information from the node even if two or more nodes are compromised.

3 The TESLA scheme

An authentication scheme based on the hash chain was proposed for the first time by Lamport [12], and then Perrig et al. applied it to the broadcast authentication scheme called TESLA [1, 2, 3, 4]. A lot of schemes using the hash chain have been proposed. We concretely explain TESLA in this section.

TESLA is an efficient broadcast authentication scheme, and is the practical scheme which has been adopted in the SRTP (Secure Real-time Transport Protocol). A client node that receives the broadcast message can authenticate a message. We present the following concrete procedure of TESLA.

- **System setup.** The base station generates the hash chain $\{c_0, c_1, \ldots, c_k\}$ ($c_j = H(c_{j-1})$, $1 \leq j \leq k$) by selecting a random seed c_0 and using the one-way hash function H. c_k is securely distributed to all the client nodes and committed.
- **Authentication.** A message authentication key is used for verification of the keyed MAC. Let $MAC_K(M)$ be the keyed MAC of a message M with the private key K. The message authentication key is defined as $K_{k-j} = KDF(c_j)$ ($j = 0, \ldots, k-1$), using the key derivation function KDF, and allocated in the $(k-j)$-th interval (see Fig.1). The authentication procedure is as follows. It is assumed that the client node received the broadcast message $\{M, MAC_{KDF(c_{j-1})}(M), c_j\}$ in the interval I_{k-j}. The client node checks $c_{j+1} \stackrel{?}{=} H(c_j)$ by using the c_{j+1} that has already been committed. If this checking result is valid, the client node can confirm that c_j is a legitimate element of the hash chain made by the base station. The client node moves the trust point from c_{j+1} to c_j, and preserves the broadcast message $\{M, MAC_{KDF(c_{j-1})}(M), c_j\}$ in its own storage. When the client node receives c_{j-1} in the next interval $I_{k-(j-1)}$, it checks $c_j \stackrel{?}{=}$

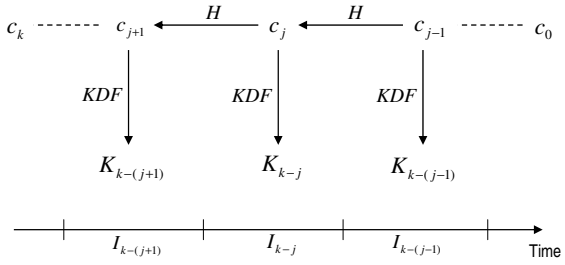

Figure 1: Chain structure in TESLA.

$H(c_{j-1})$ by using the c_j that has already been committed. Only when this checking result is valid, can the client node derive the message authentication key $K_{k-(j-1)}$. The client node can authenticate M by checking $MAC_{KDF(c_{j-1})}(M) \stackrel{?}{=} MAC_{K_{k-(j-1)}}(M)$. The client node can confirm the validity of the broadcast message M in the interval I_{k-j} during the next interval $I_{k-(j-1)}$.

TESLA has the advantages that a secret key is not required to be preserved in each node, and that it can achieve authentication like a public key authentication using only the hash function, although it has a disadvantage that an authentication delay occurs. However, TESLA eventually relies on PKI that has to be used in order to provide authentic TESLA-key. Hence TESLA requires the key management by signature-based authentication schemes, and thus it is not necessarily efficient as the entire system.

4 New lightweight linking algorithm

When Alice wants to show the link of two arbitrary values to Bob, she can easily show the link by using the signature. If she computes $\mathrm{Sig}_s(a||b)$ using her private key s in order to link a and b, then Bob verifies her signature with her public key to confirm the link. However, we want to use neither PKI nor modular operation. This algorithm can show the link of two arbitrary values efficiently without PKI and modular operation.

A lightweight linking algorithm links the predetermined random number and the arbitrary identity ID. In other words, given a random number h_0, we can show the link of ID and h_0. We aim to construct a linking algorithm using a cryptographic secure hash function H and a hard core predicate B.

Each node has the ROM where TESLA-key is stored. While TESLA has the 2ℓ-bit TESLA-key, our scheme has ℓ-bit ID and approximately ℓ-bit supplement key s as the TESLA-key, where s guarantees that h_0 is linked with ID. Let $\mathrm{ID} = ID_\ell||\cdots||ID_2||ID_1$ and $s = \cdots||s_2||s_1 = s[\ell]||\cdots||s[2]||s[1]$, where $|ID_i| = |s[j]| =$1-bit. The values of ID, s, s_i and s'_i are the binary representations. $|s_i|$ is 0-bit or more than 0-bit in Algorithm 1 and 2, because s_i includes a null value too. $x||y$ denotes the concatenation of the strings x and y.

4.1 Computation of supplement key

Given h_0, ID and ℓ, a supplement key s is computed by using Algorithm 1.

Algorithm 1 Computation of supplement key s

Input: h_0, ID, ℓ
Output: s

```
for i = 1 to ℓ do
  h'_i ← H(h_{i-1}) and ID'_i ← B(h'_i)
  if ID_i = ID'_i then
    s'_i ← ()
  else
    s'_i ← 0
    while ID_i ≠ ID'_i do
      h'_i ← H(h_{i-1}||s'_i) and ID'_i ← B(h'_i)
      s'_i ← s'_i + 1
    end while
  end if
  s_i ← s'_i and h_i ← h'_i
end for
```

The size of s becomes approximately ℓ-bit. We explain this reason as follows. If $ID_i = B(H(h_{i-1}))$ then s'_i is empty. The probability of $ID_i = B(H(h_{i-1}))$ will be $1/2$ by Assumption 4. In this case, s_i is also empty. Otherwise, 1-bit or more s_i is required. We set $s'_i = 0, 1, 00, 01, 10, 11, 000, \ldots$ by turns. We use s_i such that $ID_i = B(H(h_i||s_i))$ for the first time. For example, if $ID_i = B(H(h_i||0))$ then $s_i = 0$. If $s_i \neq 0$ and $ID_i = B(H(h_i||1))$ then $s_i = 1$. The probability of $ID'_i = 0$ or $ID'_i = 1$ will be $1/2$, respectively. If s_i is neither 0 nor 1 then the size of s_i is increased to 2 bits. As a result, we can compute $|s| = \ell \cdot \sum_{i=0}^{\infty} \frac{i}{2^{i+1}} \approx \ell$.

4.2 Verification of linking

We show a simple example of verification of linking in the case of $\ell = 80$ in Fig. 2. Let s be a binary representation such as $s = \cdots 101$. First, we check $ID_1 \stackrel{?}{=} B(H(h_1)||1)$ since $s_1 = 1$. If this checking is valid, then we check $ID_2 \stackrel{?}{=} B(H(h_2))$ because s_2 is a null value. We similarly check

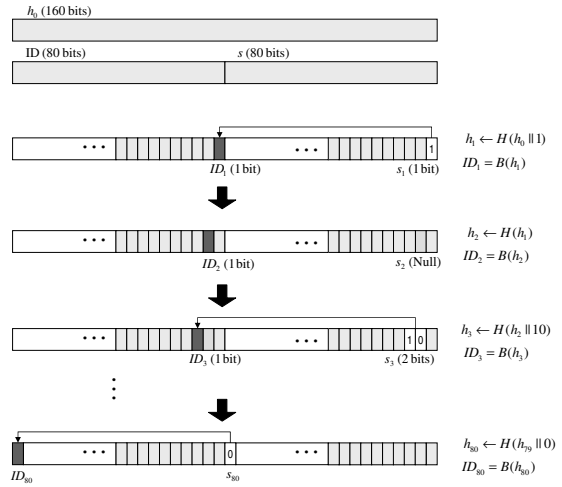

Figure 2: Simple example of verification of linking ($\ell = 80$).

$ID_3 \stackrel{?}{=} B(H(h_3)||10)$. If all the checking are valid, the verification of linking h_0 and ID succeeds.

Given h_0, ID, ℓ and s, formally, we can verify whether or not h_0 is linked with ID using Algorithm 2.

Algorithm 2 Verification of linking

Input: h_0, ID, ℓ, s
Output: TRUE or FALSE

```
for i = 1 to ℓ do
    h'_i ← H(h_{i-1}||s_i) and ID'_i ← B(h'_i)
    if ID_i ≠ ID'_i then
        return FALSE
    end if
end for
return TRUE
```

5 Our scheme

TESLA is a broadcast authentication scheme based on a hash function, and it is very efficient. We improve TESLA, and propose an alternative scheme which is a simple form of identity-based cryptography with lightweight linking algorithm. We concretely explain our scheme in this section.

5.1 Premise

We state some premises in our scheme.

- Anyone can use the hash function H and the hard core predicate B.
- The time in each client node has been synchronized with the base station in the same way as TESLA.
- An identity ID and a supplement key s are embedded in the ROM of each client node. An adversary cannot alter such data in the ROM.
- While ID and s are different in each node, h_0 is the same in all nodes.

5.2 Protocol description

Our protocol is composed of the following three phases.

- **System setup.** The base station chooses random seed $c_0 \in_R \{0,1\}^\lambda$, sets the length of hash chain k, and generates the hash chain $\{c_0, c_1, \ldots, c_k\}$ ($c_j = H(c_{j-1})$, $j = 1, \ldots, k$). The values of k and c_0 are securely preserved in the base station, and c_k is set to h_0. Then, the base station calculates the supplement key s corresponding to each ID of client nodes using Algorithm 1. ID and s are embedded in the ROM of each node.
- **Registration.** The base station broadcasts h_0 ($= c_k$). Each client node verifies h_0 using Algorithm 2. If the result of Algorithm 2 is valid, each node accepts c_k and then preserves c_k in its own storage. The trust point of the hash chain becomes from ID to c_k at this time.
- **Authentication.** This phase is the same as TESLA.

6 Analysis

6.1 Security

We comply with the adversary model described in Section 2.4. We discuss unforgeability of a message in our scheme. In order to prove the unforgeability of a message, we have to prove both the unforgeability of linking h_0 with ID and the unforgeability of TESLA. Theorem 6.1 proves that it is difficult to calculate h_0 ($= c_k$) which is linked with ID.

Theorem 6.1 (Unforgeability of linking) *The proposed scheme satisfies unforgeability of linking if the hash function H is a cryptographic secure hash function.*

Proof. Assume to the contrary, there exists a polynomial time algorithm $\mathcal{A}_1$ that on input ID, s and

h_1 outputs, with a probability which is not negligible, the value of h_0 $(= c_k)$ which is the end of a hash chain in TESLA. Then, we can use $\mathcal{A}_1$ to break the one-wayness of H as follows: For a randomly given instance a_1, define $h_1 = a_1 = H(b_1)$ (The value of b_1 is not disclosed). $\mathcal{A}_1$ can output h_0. Then, we can compute b_1 such that $b_1 = h_0||s_1$. Since s_1 is a few bits at most, we can easily find it. Therefore, we can break the one-wayness of H. On the other hand, there exists a polynomial time algorithm $\mathcal{A}_2$ that on input ID, s, h_1 and h_0 outputs, with a probability which is not negligible, the value of h'_0 $(\neq c_k)$ which is the end of another hash chain in TESLA. Then, we can use $\mathcal{A}_2$ to break the collision resistance of H as follows: For a randomly given instance a_2, define $h_1 = H(a_2) = H(b_2)$ (The value of b_2 is not disclosed). $\mathcal{A}_2$ can output h'_0. Then, we can compute b_2 such that $b_2 = h'_0||s'_1$. Since s'_1 is a few bits at most, we can easily find it. Therefore, we can break the collision resistance of H. ∎

Theorem 6.2 *Given ℓ-bit ID, s and an arbitrary h'_0 $(\neq h_0)$, the probability that Algorithm 2 incidentally outputs TRUE is $1/2^\ell$.*

Proof. By Assumption 4, the probability such that $ID_i = ID'_i$ $(\exists i \in \{1, \ldots, \ell\}, ID'_1 = B(H(h'_0||s_1)))$ is 1/2. Hence the probability such that $ID_i = ID'_i$ $(\forall i \in \{1, \ldots, \ell\})$ is $1/2^\ell$. ∎

6.2 Efficiency

The computational complexity, the amount of communication and the storage consumption in our scheme is as efficient as TESLA. Our scheme does not use the modular operation in an authentication. While the size of TESLA-key in [5] is 1024-bit, one in our scheme is 160-bit which is almost the same as TESLA, where $|n|$ =1024-bit and $|H|$ =160-bit.

7 Conclusion

We proposed a new lightweight linking algorithm and improved a broadcast authentication scheme based on a simple form of identity-based cryptography using hash function. As a result, our scheme does not require the key management, and is efficient as the entire scheme. Also, we proved the unforgeability of our scheme. Our scheme has the same advantages as TESLA, in which a client node need not have a private key and the authentication requires only one-round of communication. Furthermore, the computational complexity, the amount of communication and the storage consumption in our scheme is as efficient as TESLA.

References

[1] A. Perrig, R. Canetti, J.D. Tygar and D. Song, "Efficient authentication and signing of multicast streams over lossy channels," In S&P'00, pp.56–, IEEE, 2000.

[2] A. Perrig, R. Canetti, D. Song and J.D. Tygar, "Efficient and secure source authentication for multicast," In NDSS'01, pp.35–46, 2001.

[3] A. Perrig, R. Szewczyk, V. Wen, D. Culler and J.D. Tygar, "SPINS: Security protocols for sensor networks," In MobiCom'01, pp.512–534, ACM, 2001.

[4] A. Perrig, R. Canetti, J.D. Tygar and D. Song, "The tesla broadcast authentication protocol," RSA CryptoBytes, Vol.5, No.2, 2002.

[5] B. Groza, "Broadcast authentication with practically unbounded one-way chains," Journal of Software, vol.3, no.3, pp.11–20, 2008.

[6] S. Chang, S. Shieh, W.W. Lin and C.M. Hsieh, "An efficient broadcast authentication scheme in wireless sensor networks," In ASIACCS'06, pp.311–320, ACM, 2006.

[7] A. Mashatan and D.R. Stinson, "A new message recognition protocol for ad hoc pervasive networks," In CANS'08, LNCS 5339, pp.378–394, Springer-Verlag, 2008.

[8] I. Goldberg, A. Mashatan and D.R. Stinson, "A new message recognition protocol with self-recoverability for Ad Hoc Pervasive Networks," In ACNS'09, LNCS 5536, pp.219–237, Springer-Verlag, 2009.

[9] A. Shamir, "Identity-based cryptosystems and signature schemes," In CRYPTO'84, LNCS 0196, pp.47–53, Springer-Verlag, 1984.

[10] P. Michiardi and R. Molva "IDHC: ID-based hash-chains for broadcast authentication in wireless networks," EURECOM Technical Report RR-04-111, 2004.

[11] NIST, "Secure hash standard," National Institute for Standards and Technology, Gaithersburg, MD, USA, 1995.

[12] L. Lamport, "Password authentication with insecure communication," Communications of the ACM, vol.24, no.11, pp.770–772, November 1981.

[13] O. Goldreich and L.A. Levin, "A hard-core predicates for any one-way functions," In STOC'89, pp.25-32, 1989.