Journal of Information Hiding and Multimedia Signal Processing ©2017 ISSN 2073-4212
Ubiquitous International Volume **8**, Number **4**, July **2017**

ShortHC: Self-Regenerated Short Hash Chain for Resource-Constrained System

Liang-min Wang

School of Computer Science and Technology
Anhui University
No. 111, Jiu Long Rd., Hefei, Anhui, 230601, China
jasonwanglm@gmail.com

Qing-qing Xie

School of Computer Science and Technology
Anhui University
No. 111, Jiu Long Rd., Hefei, Anhui, 230601, China
qingqingxieahu@gmail.com

Tao Jiang

Department of Computer Engineering
Xidian University
No. 2, South Taibai Rd., Xi'an, Shaanxi, 710071, China
jiangt2009@gmail.com

Chin-Chen Chang*

Department of Information Engineering and Computer Science
Feng Chia University
No. 100, Wenhwa Rd., Seatwen, Taichung, Taiwan, 40724, R.O.C.
Department of Computer Science and Information Engineering
Asia University
No. 500, Lioufeng Rd., Wufeng, Taichung, Taiwan, 41354, R.O.C.
*Corresponding author: alan3c@gmail.com

Received July, 2014; revised March, 2016

Abstract. *Hash Chains are widely used in the resource-constrained systems to realize the lightweight security. They can partly replace the public key algorithms to implement the function of message source and integrity authentication with a lower cost. However, Hash Chain meets a dilemma that a long chain increases the system's overhead of computation and storage, whereas a short chain produces a short security lifetime and reduces the scalability of the system. This paper presents a self-regenerated short Hash Chain protocol. It can effectively extend the lifetime of the Hash-Chain-based security protocol on-demand. A group of short Hash Chains is employed to replace the long Hash Chain, in which many different Hash Chains are connected by the secure and authenticated communication. In the security analysis section, the security of the proposed protocol is proved to rely on the one-way characteristic of each short Hash Chain. Taking the wireless sensor network as a typical scenario of the resource-constrained system, the proposed protocol is used to improve some existing typical secure protocols, such as broadcast authentication μTESLA and ShortPK. The simulation results show that our protocol can prolong the lifetime of the network and effectively reduce the network resource overheads.*
Keywords: Integrity, Network Protocols, Security, Wireless sensor networks

1. **Introduction.** Hash Chains are computation-efficient and can provide one-way feature in cryptographic protocols. They are widely used in various scenes, such as password system [4, 9], secure routing [5, 6], micro-payment scheme [8, 15], RFID security and privacy [13, 16], and WSNs broadcast authentication [11, 14, 17]. Hash Chains can be used to optimize or even partly replace the public key algorithms [3, 12] to achieve the function of the message source authentication and integrity authentication.

However, the Hash Chain has the following drawbacks which limit its applications:

(a) The Hash Chain can only provide the backward security but not the forward security.

(b) The length of the Hash Chain is limited, which makes it difficult to meet the requirements of various applications.

(c) Extending the length of the Hash Chain is difficult, because a secure channel established through other encryption mechanisms is needed, and a larger overhead is required.

2. **Related Work.** When the advantages of the backward security of the Hash Chain are taken, the overhead of the message source authentication can be reduced. However, the Hash Chain cannot provide the forward security, which affects the independence of keys in the Hash Chain. Therefore, the presence of drawback (a) limits the application of the Hash Chain in ensuring the data confidentiality. Thus, the Hash Chain is often used for the authenticable communication, where the forward security does not need to be considered. For this reason, the Hash Chain is widely used to improve both the public-key-mechanism-based security protocols [8, 15, 17] and protocols where the public-key application mechanism is constrained [1, 11, 14, 16]. In these protocols, the Hash Chain can be used to optimize or even replace the public key mechanisms to achieve the message source and integrity authentication.

However, drawbacks (b) and (c) of Hash Chains are difficult to overcome in the authentication process. The drawback (b) is caused by the length of the Hash Chain, which is a constant value and cannot be changed after the Hash Chain is used. Unfortunately, the length of Hash Chain is bound with the security lifetime. The Hash Chain cannot be extended directly as it is often used in a reverse direction. Thus, the Hash Chain generator is required to generate a Hash Chain with a length of N_{Max} large enough for most application requirements. However, N_{Max} is difficult to be predetermined. If N_{Max} is not big enough, the system will become unavailable when the Hash Chain is used up. On the other hand, N_{Max} is not allowed to be too large as the computation and storage overheads of a long Hash Chain are difficult to be accepted in many applications. Liu et al. [11] pointed out that, if each Hash value had a 100 ms life cycle, the broadcast node needed to use a Hash Chain with a length of $N = 24 \times 60 \times 60 \times 10 = 864,000$ every day. Hence, the broadcast node needs to store $8 \times N = 69{,}120{,}000$ bytes Hash value. If the system only uses the seed of the Hash Chain to calculate the Hash Chain, $N(N-1)/2$ Hash operations are required. Therefore, the length of Hash Chains should be short to reduce the computation and storage overheads in the application process.

When a short Hash Chain is exhausted, the Hash Chain extension problem occurs. In other words, a new Hash Chain should be deployed. In the process of the Hash Chain extension, a secure channel is required to securely transmit the tail of the new Hash Chains to the message receivers. In order to establish the secure transmission channel to prevent the impersonation attack and guarantee the message source authentication of the Hash Chain, an asymmetry or symmetry encryption mechanism was proposed in [14]. However, in the resource-constrained networks, the public key mechanism is unacceptable because of its heavy computation and communication overheads. Generally, the resource-constrained networks are lack of the public key infrastructure. Thus, the symmetric cryptography mechanism is considered to establish the secure channel because of the

fewer overheads of computation and communication. However, the key management of the symmetric cryptography mechanism is costly in the new keys deployment phase. For example, in wireless sensor networks, Perrig et al. [14] use the base station to assist the deployment of new Hash Chains, in which the base station is assumed as a non-exhausted node.

Actually, we can evaluate the time and communication costs. Let n be the number of nodes in the network, and the base station deploys an 8-byte Hash value for each node with the help of a handshake protocol between the base station and the normal node. Then, $O(n)$ operations of encryption, decryption and communication run at the base station. Liu et al. [10] pointed out that, if each packet was 30 bytes and the network bandwidth was 10 kbps. Even making full use of the channel, and without considering the case of the multi-hop communication and the packet loss, the base station had to take at least $T = \frac{4{,}000\times30\times8}{10{,}240} = 93.75$ seconds to deploy a new Hash value by sending and receiving messages in a network with 2,000 nodes. The waiting time is also too long for most applications. Therefore, both the symmetric and the asymmetric encryption mechanism are not good enough in deploying a new Hash Chain for the short Hash Chain extension.

In this paper, we present a self-regenerated short Hash Chain protocol, in which a group of short Hash Chains, instead of a long Hash Chain, are adopted to reduce the computation and communication overheads. The protocol securely deploys a new short Hash Chain by taking advantage of the one-way feature of the current Hash Chain. The most related schemes are in [2, 7, 20, 21]. Our work is similar to them because they are also aiming at providing efficient methods for Hash Chain re-initialization in resource-constrained system. Hu et al. [7] present the sandwich chain scheme, in which only one current Hash is required, just like the hierarchical one-way chain schemes. It induces a delay before the secondary chain value. Compared with previous reported schemes, the first of the contributions of our scheme is that we realize the authentication of new Hash Chain without any additional delay. The second contribution is the flexible re-initialization process. We can re-initialize a new Hash Chain in any point of the current Hash Chain. This random re-initialization way can resist packages loss or package capture attack on specific package to some extent, and makes our scheme different from the schemes in [2, 20, 21].

We also test our protocol in some typical broadcast authentication protocols for wireless sensor networks, which is a typical resource-constrained network. The results show that our protocol significantly reduces the computation and communication overheads of the broadcast scheme and prolongs the lifetime of the networks.

3. **ShortHC: Self-regenerated Short Hash Chain.**

The self-regenerated short Hash Chain protocol has three phases: short Hash Chain pre-distribution phase, short Hash Chain usage phase, and short Hash Chain extension phase. Here the notations and pre-loaded materials used in our protocol are as follows. Let S and R denote the communication initiator and the recipient, respectively. S is equipped with a random value generator, a message authentication code (MAC), and a Hash function $h(\cdot)$. R is equipped with the MAC and the Hash function $h(\cdot)$. The time of the network is divided into some periods with the same length, and each Hash value is bound to a time period in a reversed way.

3.1. **Short Hash Chain pre-distribution phase.**

The initiator S generates an initial random value $K^0_{n_0}$ as a seed of the first short Hash Chain K^0. Then S uses the pre-loaded

Hash function $h(\cdot)$ to compute the n_0 Hash value of the short Hash Chain K^0 as follows:

$$K_1^0 \xleftarrow{h(\cdot)} \cdots \xleftarrow{h(\cdot)} K_i^0 \xleftarrow{h(\cdot)} \cdots \xleftarrow{h(\cdot)} K_{n_0}^0. \tag{1}$$

Before deployment, the value K_1^0 is preloaded in R. Then K_1^0 and the one-way Hash Chain K^0 help R and S realize the message source and integrity authentication in the subsequent communication.

3.2. **Short Hash Chain usage phase.** During the short Hash Chain usage phase, the Hash value K_i^j $(i = 1, 2, \ldots, n_j - 1, j \in \mathbb{N})$ is used from K_1^j to $K_{n_j}^j$ corresponding to time period T_{i-1}^j $(i = 2, 3, \ldots, n_j, j \in \mathbb{N})$. In time period T_i^0 $(i = 1, 2, \ldots, n_0 - 1)$, S releases the message Msg_i^0 and its corresponding message authentication code $MAC(K_{i+1}^0, Msg_i^0)$ to R, where Msg_i^0 is the content of the current message and K_{i+1}^0 is the key used for the receivers to verify the validity of Msg_i^0. R is responsible for receiving and storing messages and can authenticate the stored message when receiving K_{i+1}^0 in time period T_{i+1}^0. The Hash Chain usage phase is introduced in detail in μTESLA [14].

If the length of the Hash Chain is insufficient, the protocol enters the short Hash Chain extension phase.

3.3. **Short Hash Chain extension phase.** Assume that the running short Hash Chain is K^j. The seed of K^j is $K_{n_j}^j$ and the length is n_j. In time period T_i^j, K_{i+1}^j $(1 \le i \le n_j - 2)$ is the MAC key of the current message and K_{i-1}^j is stored in R. According to (1), S randomly chooses a new seed $K_{n_{j+1}}^{j+1}$ and generates a new short Hash Chain K^{j+1}:

$$K_1^{j+1} \xleftarrow{h(\cdot)} \cdots \xleftarrow{h(\cdot)} K_i^{j+1} \xleftarrow{h(\cdot)} \cdots \xleftarrow{h(\cdot)} K_{n_{j+1}}^{j+1}. \tag{2}$$

Then, the current Hash Chain K^j is used to distribute the tail of the new Hash Chain K^{j+1}. Before K^j is exhausted, S needs to send three binding information messages to R in different time periods for securely transferring first value K_1^{j+1} of the reversed Hash Chain K^{j+1}.

Assume that the first message is sent in time period T_i^j, and S calculates $K_0^{j+1} = h(K_1^{j+1})$ and $K' = K_m^j \oplus K_n^{j+1}$ $(i + 1 \le m \le n_j)$ before time period T_i^j. Then S sends binding information

$$Time_i^j | i | K' | MAC(K_{i+1}^j, Msg_i^j) | K_i^j \tag{3}$$

with the current sending message Msg_i^j to R, in which $Time_i^j$ is the time stamp of the current message and i is the identity of K_i^j. After receiving message (3), R discards K_{i-1}^j and stores K_i^j if $K_{i-1}^j = h(K_i^j)$. Then R stores the received message until obtaining K_m^j and K_1^{j+1} for the next authentication process.

In time period $T_m^j (i + 1 \le m \le n_j)$, S sends the second binding information

$$Time_m^j | m | MAC(K_{m+1}^j, Msg_m^j) | K_m^j \tag{4}$$

with the current sending message Msg_m^j to R. If $K_{m-1}^j = h(K_m^j)$, R uses K_m^j and $MAC(K_m^j, Msg_{m-1}^j)$ to authenticate the integrity of Msg_{m-1}^j. Msg_{m-1}^j and $MAC(K_m^j, Msg_{m-1}^j)$ is received in time period T_{m-1}^j. Then, R discards K_{m-1}^j and stores K_m^j. After that, R calculates and stores $K_0^{j+1} = K' \oplus K_m^j$. At this point, R stores message (4) for future authentication.

In time period T_1^{j+1}, S sends the third binding information (5) with Msg_1^{j+1} to R and releases the secret key K_1^{j+1}

$$Time_1^{j+1}|1|MAC(K_2^{j+1}, Msg_1^{j+1})|K_1^{j+1}. \tag{5}$$

After receiving message bound with (5), R uses the stored K_0^{j+1} and the received K_1^{j+1} to verify $K_0^{j+1} = h(K_1^{j+1})$. Now R securely obtains the tail of the new Hash Chain K^{j+1}. Before the second message, K_m^j is not released. Then no one knew K_m^j and sent $K' = K_m^j \oplus K_0^{j+1}$ in time period T_i^j except S. Thus R believes that S is the unique node who sent him the value K'. Therefore, the attackers are prevented from forging K' for the reason that K_m^j is unknowable.

We stressed one of our contributions that m could be randomly chosen and the sender could re-initialize a new Hash Chain at any point of current Hash Chain. Thus, our scheme is flexible and provides a probability re-initialization process which, to some extent, will resist the specific package capture attack in the determined Hash Chain re-initialization schemes.

By sending the messages (3), (4) and (5) bound with Msg_i^j, Msg_m^j and Msg_1^{j+1} in time period T_i^j, T_m^j and T_1^{j+1}, S and R have securely generated the deployment of the new short Hash Chain K^{j+1} without a trusted third part. Therefore, the Hash Chain self-regeneration protocol provides an on-demanded short Hash Chain extension without exhaustion if the current short Hash Chain can provide more than 2 Hash values for the distribution of the new Hash Chain.

4. Security Analysis of ShortHC.

In this section, first the definitions of the security tools used in ShortHC are introduced, and then our security assumption is presented. We also present the security proof of our ShortHC when it is used in the classical μTESLA, in which the message source and integrity authentication characteristics are proved.

4.1. Definitions of basic concepts.

Definition 4.1. SHF: (***Secure Hash Function***). The SHF is a publicly known function $f_n : \{0,1\}^* \mapsto \{0,1\}^n$. It takes x as an input and outputs a bit string $f_n(x)$ of length n. In $f_n(x)$, x is selected uniformly from a bit string with an arbitrary length $\{0,1\}^*$. At the same time, the SHF satisfies such a condition that A is a probabilistic polynomial time algorithm that takes $f_n(x) \in \{0,1\}^n$ as input and outputs $A(f_n(x)) \in \{0,1\}^n$. For each A, polynomial P and all sufficiently large n, there is an arithmetic expression as follows:

$$\text{Prob}\{f_n(A(f_n(x))) = f_n(x)\} < \frac{1}{P(n)}. \tag{6}$$

That is to say, if $h(\cdot)$ is a secure Hash function, it satisfies the following informal conditions:

1. The description of the SHF $h(\cdot)$ must be publicly known.
2. The argument X have an arbitrary length and the result $h(X)$ has a fixed length of k bits and that is for $h : \{0,1\}^* \mapsto \{0,1\}^k$.
3. For given h and X, it is easy to compute $h(X)$.
4. The Hash function must be one-way. For a given Y in the image of $h(\cdot)$, it is hard to find a message X to make $h(X) = Y$. Also for given X and $h(X)$, it is hard to find a message $X' \neq X$ to make $h(X') = h(X)$.
5. The Hash function must be collision resistant. This means that it is hard to find any two distinct messages X' and X to achieve $h(X') = h(X)$.

Definition 4.2. ***Random number generator.*** $g = \{g_n | n \in \mathbb{N}\}$ is a pseudo-random string generator (PSG). *iff* $g(U) = \{g_n(U) | n \in \mathbb{N}\}$ is a deterministic polynomial time string generator. For each statistical test $T(\cdot)$ that on input x outputs a bit 0/1, for each polynomial P, and for sufficiently large n,

$$|Prob\{T(x_1)\} - Prob\{T(x_2)\}| < \frac{1}{P(n)}, \tag{7}$$

where $x_1 \in E^{1\sum l(n)}, x_2 \in E^{2\sum l(n)}$, both E^1 and E^2 are assembled with a length $l(n)$.

Definition 4.3. ***One-way Hash Chains.*** Select a seed s with the random number generator and use the Hash function $h(\cdot)$ to generate the Hash Chain $\{s_i\}_{i=1}^n$, in which

$$s_{i+1} = h(s_i) = \overbrace{h(h(...h}^{i}(s))). \tag{8}$$

One way Hash Chains $\{s_i\}_{i=1}^n$ can be figuratively expressed as following:

$$s(s_1) \xrightarrow{h(\cdot)} s_2 \xrightarrow{h(\cdot)} \cdots \xrightarrow{h(\cdot)} s_i \xrightarrow{h(\cdot)} s_{i+1}.$$

Definition 4.4. ***Secure Message Authentication Code.*** The secure Message Authentication Code (MAC) is a function satisfying the following conditions:
1. The description of $MAC(\cdot, \cdot)$ must be publicly known and the only secret information lies in the key.
2. Giving a key K of n-bit length and an X of an arbitrary length as the arguments of $MAC(\cdot, \cdot)$, the result $MAC(K, X)$ has a fixed length of n bits.
3. Giving $MAC(\cdot, \cdot)$, X and K, the computation of $MAC(K, X)$ must be easy.
4. Giving $MAC(\cdot, \cdot)$ and X. It is hard to determine $MAC(K, X)$ with a probability of success which is significantly higher than $1/2^n$. Even when a large set of pairs $\{X_i, MAC(K, X_i)\}$ (X_i has been selected by the opponent) are known, it is hard to determine the key K or to compute $MAC(K, X')$ for any $X' \neq X_i$.

Definition 4.5. ***Broadcast authentication scheme based on One-way Hash Chains.*** The scheme is a distributed system and each participator uses a polynomial-time machine to initiate its instance of the scheme. The scheme includes the following components:
1. A security parameter k which determines the size of keys. All of algorithms are polynomial in k.
2. A message space MS and a key space KS. All messages of a given size should be polynomial in k.
3. A system time state TS which records the state of the system.
4. A key generation algorithm $KGA(\cdot)$ which uses the random number generator to randomly generate two k-bit private keys $K_{n_j}^j$ and $K_{n_{j+1}}^{j+1} \in_R \{0,1\}^k$ as the seeds. The key generation algorithm $KGA(\cdot)$ uses Hash function $h(\cdot)$ defined in Definition 4.1 to generate the one-way Hash Chains $\{K_i^j\}_{i=1}^{n_j}$ and $\{K_i^{j+1}\}_{i=1}^{n_{j+1}}$ as defined in Definition 4.3.
5. A message signing algorithm $MSA(\cdot)$ giving a message, a system state, and a key. It generates a MAC and updates the time state of the system.
6. A message verification algorithm $MVA(\cdot)$, which gives a message, a MAC, a key, and a time state of the system. It checks the validity of the corresponding message.

4.2. **Security assumptions.** The asymmetry characteristic of the ShortHC is based on the "first storage and then authentication" mode. Therefore, the security of the ShortHC protocol relies on the following assumptions:

1. The selected Hash function $h(\cdot)$ and the message authentication function MAC are secure and can provide weak collision resistance.

2. The receiver's clock is time synchronized up to a maximum error, usually denoted as Δ [14]. The receiver can reject the message out the time period for Δ plus the acceptable transmission delay.

3. In the resource-constrained environment, the security authentication problems of the multi-hop environment can refer to literature [19]. The fault data injection and the path-based denial of the service attack are related to the application environments. Thus we don't take them into account in our security and performance analysis of the ShortHC.

4.3. **ShortHC systems.** The secure extension of the Hash Chain is realized in the ShortHC protocol, in which the message source and the integrity authentication are obtained from μTESLA. Our ShortHC system consists of the following processes.

1. Initialization

a. Creating the first Hash Chain. The first Hash Chain of the system is initialized by creating a secure k-bit seed $K_{n_0}^0$ with the random number generator from space $\{0,1\}^k$. We denote this Hash Chain as $\{K_i^0\}_{i=1}^{n_0}$ according to Definition 4.5. The seed $K_{n_0}^0$ of $\{K_i^0\}_{i=1}^{n_0}$ is securely stored at the trusted place of the system.

b. Initializing the time state of the system $TS_1^0 = 1$.

c. Securely distributing the Hash Chain tail K_1^0 and the system time state TS_1^0. It should be emphasized that K_1^0 and TS_1^0 are distributed through pre-distribution or in some securely authenticable way.

2. Message signing

In time state TS_i^0, the system initiator runs the $MSA(Msg_i^0, TS_i^0)$, and computes the message authentication code $MAC(K_{i+1}^0, Msg_i^0)$ of message Msg_i^0. Then, it constructs the broadcast message $\{Msg_i^0|MAC(K_{i+1}^0, Msg_i^0)|K_i^0\}$, and distributes the message to the receivers.

3. Message verification

Upon receiving the message $\{Msg_i^0|MAC(K_{i+1}^0, Msg_i^0)|K_i^0\}$, the receivers run the $MVA($ $Msg_{i-1}^0, MAC(K_i^0, Msg_{i-1}^1), K_i^0, TS_i^0)$ and check $K_{i-1}^0 = h(K_i^0)$, where K_{i-1}^0 is received in the last time period and stored in the local storage. With the message Msg_{i-1}^0 and the key K_i^0, the receivers then compute $MAC(K_i^0, Msg_{i-1}^0)$ and verify the validity of message Msg_{i-1}^0.

If all the verifications are successful, the verifier has to update the time state of the system and obtains $TS_{i+1}^0 = i+1$. Then, the receivers store the message $\{Msg_i^0|MAC(K_{i+1}^0,$ $Msg_i^0)|K_i^0\}$ for the authentication of the next message.

4. Self-regeneration of Hash Chain

When the current Hash Chain $\{K_i^0\}_{i=1}^{n_j}$ is not long enough for the system, the system initiator self-regenerates the Hash Chain as follows:

a. Initialization of the new Hash Chain. Generating a secure k-bit seed $K_{n_{j+1}}^{j+1}$ with the random number generator from space $\{0,1\}^k$, creating a Hash Chain $\{K_i^{j+1}\}_{i=1}^{n_{j+1}}$ as defined in Definition 4.4, and keeping the seed $K_{n_{j+1}}^{j+1}$ securely in the local storage.

b. Distribution of the tail of Hash Chain K^{j+1}. When the time state is TS_i^j, the system initiator computes $K_0^{j+1} = h(K_1^{j+1})$, $K' = K_m^j \oplus K_0^{j+1} (i+1 \le m \le n_j)$ and the message authentication code $MAC(K_{i+1}^j, Msg_i^j)$, where $Msg_i^j = \{Time_i^j|i|K'\}$. In Msg_i^j, $Time_i^j$ is the time stamp of the message and i is the identity of the Hash value in the current Hash Chain. Then initiator constructs the broadcast message $\{Time_i^j|i|K'|MAC(K_{i+1}^j, Msg_i^j)|K_i^j\}$, in which $1 \le i \le n_j - 1$. After that the initiator distributes the message to the receivers.

5. New Hash Chain verification

When the receivers receive the message $\{Time_i^j|i|K'|MAC(K_{i+1}^j, Msg_i^j)|K_i^j\}$, they can use the key K_i^j to verify the validity of the message Msg_{i-1}^j as mentioned in the message verification phase. The message Msg_{i-1}^j is received in time state TS_{i-1}^j. When the message $\{Time_m^j|m|MAC(K_{m+1}^j, Msg_m^j)|K_m^j\}$ is received in time state TS_m^j, the receivers verify $K_{m-1}^j = h(K_m^j)$ and compute $MAC(K_m^j, Msg_{m-1}^j)$ with K_m^j as the key, in which K_{m-1}^j and Msg_{m-1}^j are received in time period TS_{m-1}^j. In time state TS_1^{j+1} of the system, when the receivers receive the message $\{Time_1^{j+1}|1|MAC(K_2^{j+1}, Msg_1^{j+1})|K_1^{j+1}\}$, the receivers compute and verify $K_0^{j+1} = h(K_1^{j+1})$ and then obtain the new Hash Chain tail K_1^{j+1}.

4.4. **Security proof of ShortHC systems.** An adversary can successfully conduct an attack, which means that it can break the message source or the integrity authentication of the protocol. Assume that the attacker can conduct the attack with a non-negligible probability $\varepsilon \geq 1/P(k)$ and the probabilities for the attacker to destroy the message source authentication and the message integrity authentication are $p(0 \leq p \leq 1)$ and $q(0 \leq q \leq 1)$, respectively. Thus, in a successful attack, the probability of destroying the message source authentication is $p \cdot \epsilon$ and the probability of destroying the message integrity is $q \cdot \epsilon$.

Theorem 4.1. *During each Hash Chain usage phase, the ShortHC protocol is cryptographic and secure.*

Proof: Theorem 4.1 means that if the tail of each Hash Chain is distributed to the receiver in a secure way and the selected cryptography tools can provide properties as shown in the aforementioned definitions, the ShortHC protocol can provide the source and integrity of messages.

On the one hand, to break the security of message source authentication in time state TS_i^j, the attacker is able to forge the Hash Chain. That means that attacker can find the key $(K_t^j)'$ and $K_i^j = h^{t-i}((K_t^j)')$ $(i+1 \leq t \leq n_j)$, when the receivers only know K_i^j $(1 \leq i \leq n_j)$. According to Definition 4.1, we know that it is hard to find $(K_t^j)' \neq K_t^j$ and $K_i^j = h^{t-i}((K_t^j)')$ $(i+1 \leq t \leq n_j)$, if the adopted Hash function is an SHF. Therefore, the probability for the attacker to successfully realize the attack on the message integrity authentication of the protocol is $\varepsilon_1 < p/P(k)$.

On the other hand, to break the security of the data integrity authentication, before acquiring the key $K_t^j (i+1 \leq t \leq n_j)$, the successful attacker is able to forge data $K^* \neq K_m^j \oplus K_0^{j+1}$ and

$$\begin{aligned} &MAC((K_t^j)', Time_{t-1}^j|t-1|K^*) \\ =&MAC(K_t^j, Time_{t-1}^j|t-1|K_m^j \oplus K_0^{j+1}), \end{aligned} \tag{9}$$

where $K_i^j = h^{t-i}((K_t^j)')$. According to the above analysis, it is difficult to find a key which satisfies $(K_t^j)' \neq K_t^j (i+1 \leq t \leq n_j)$, which means that the attacker is able to forge the data descripted by (9) without knowing the secret key K_t^j. According to Definition 4.4, we know that it is a hard problem in cryptography to find $MAC((K_t^j)', Time_{t-1}^j|t-1|K^*)$ satisfying (9) without knowing the secret key of the adopted SMAC. Thus the probability of that an attacker to successfully realize an attack on the data integrity authentication of the protocol is $\varepsilon_2 < q/P(k)$.

Therefore, the probability of conducting an attack is $\varepsilon^* = 1 - \overline{\varepsilon_1} \cdot \overline{\varepsilon_2} = (p+q-pq)/P(k) < 1/P(k)$ and the result guarantees the statement of Theorem 4.1. During each Hash Chain, the ShortHC is secure against the attacks mentioned above.

Theorem 4.2. *The ShortHC protocol is secure in the extension phase of a new Hash Chain.*

Proof: If the Theorem 4.2 is untenable, the attacker can successfully choose a new Hash Chain to access the network and the legal receivers cannot perceive the malicious action.

Before the beginning of our proof, we present our assumption that the receivers can receive the message in an acceptable time delay. Under this condition, we focus on the security of the ShortHC, not the other security problems mentioned in the security assumption.

When the time state of the system is TS_i^j, the system initiator sends the message $\{Time_i^j|i|K_0^{j+1} \oplus K_m^j|MAC(K_{i+1}^j, Msg_i^j)|K_i^j\}$, where $K' = K_0^{j+1} \oplus K_m^j$ and $1 \leq i \leq n_j - 1$. Then, a successful attacker is able to find $K^* \neq K'$ to ensure that

$$MAC((K_{i+1}^j)', (Msg_i^j)') = MAC(K_{i+1}^j, Msg_i^j), \tag{10}$$

where $K_i^j = h((K_{i+1}^j)')$ and $(Msg_i^j)' = \{Time_i^j|i|K^*\}$. Theorem 4.1 implies that it is untenable to find $(K_i^j)' \neq K_i^j$ and $K_{i+1}^j = h((K_i^j)')$. It means that the attacker is able to find $(Msg_i^j)' \neq Msg_i^j$ and (10).

However, according to Theorem 4.1, the attacker cannot forge a false data packet during each Hash Chain. Thus, the ShortHC protocol can ensure the source and the integrity of $K_0^{j+1} \oplus K_m^j$. In time state ST_1^{j+1} , the received K_m^j has been verified through $K_i^j = h^{m-i}(K_m^j)$, and $K_0^{j+1} = K' \oplus K_m^j$ is obtained. Therefore, according to the security of the Hash function, the tail of the new Hash Chain K_1^{j+1} can be verified through $K_0^{j+1} = h(K_1^{j+1})$ and the theorem follows.

The above security analysis of the ShortHC proves that under the security assumption, the pre-deployment of the Hash Chain, the utilization and deployment of the new Hash Chain are secure. Thus, one Hash Chain can be used to securely initialize another Hash Chain, which does not affect the security of the scheme in the extension process.

5. **Application.** The ShortHC protocol can be applied to the existing wireless sensor network broadcast authentication schemes [11, 14, 17]. The on-demanded extension of the Hash Chains can enhance the lifetime of the network. At the same time, The ShortHC protocol can effectively reduce the computation, storage, and communication overheads of both broadcasters and receivers if our short Hash Chains and the Hash Chain self-regeneration method are adopted in these broadcast authentication schemes.

5.1. **μTESLA based on ShortHC.** We use the symbols described in TABLE 1. The total computation and communication overhead of the broadcaster in the broadcasting system based on our ShortHC is

$$E_{\text{ShortHC}} = a \sum_{j=1}^{n} \frac{x_j(x_j+1)}{2} + (3a+2b)(n-1). \tag{11}$$

In (11), $3a$ includes the overheads of 1 Hash computing and 2 MAC calculations, and $2b$ represents the overheads of 2 message transitions. FIGURE 1 shows the system overheads of a single Hash Chain and our ShortHC-based multi short Hash Chains with different chain lengths used in the μTESLA-based broadcasting system. Obviously, the length of the Hash Chain is shorter, and the overhead is less.

In the ShortHC protocol, if all the short Hash Chains have the same length x, equation (12) is obtained

$$\begin{aligned} E_{\text{ShortHC}} = {} & a\tfrac{x(x+1)}{2} \left\lfloor \tfrac{N}{x} \right\rfloor + \left((3a+2b) \left\lfloor \tfrac{N}{x} \right\rfloor - 1\right) \\ & + a\tfrac{(x+r)(x-r+1)}{2}, \end{aligned} \tag{12}$$

TABLE 1. Symbol and notation

Symbol	Notation
N	Length of a single Hash Chain
K^j	The j^{th} Hash Chain of ShortHC
x_j	Length of short Hash Chain K^j
n	Number of Hash Chains in ShortHC
a	Computation overhead of Hash function
b	Communication overhead of transmitting a corresponding Hash value

in which $r = N \bmod x$. If N can be divided by x, equation (12) is simplified to

$$E_{\text{ShortHC}} = \frac{aNx}{2} + \frac{(3a+2b)N}{x} + \frac{aN - 4b - 10a}{2}. \tag{13}$$

$E_{ShortHC}$ has a minimum value when x satisfies (14):

$$\frac{aNx}{2} = \frac{(3a+2b)N}{x}. \tag{14}$$

From (14), we obtain

$$x = \sqrt{6 + 4b/a}. \tag{15}$$

That is to say, the minimum value of $E_{ShortHC}$ is determined by the ratio of communication overhead to computation overhead of the Hash function. Then the recommended optimal length of the short Hash Chain is also decided by this ratio, and we have Theorem 5.1 as a good feature of our ShortHC system:

Theorem 5.1. *Assume that the short Hash Chains in a ShortHC system are of the same length x, let a and b denote overhead of computing a Hash function and transmitting a Hash value respectively, then (15) gives the optimal value for the system to meet its minimum cost of the total overheads of computation and communication.*

To be much simpler and more intuitive, we assume that the RC5 is used as the Hash function in the ShortHC system. In [18], we know that the computational overhead of RC5 is equivalent to 1-byte transmission overhead. Thus, in the ShortHC, the message of each transmission contains 2 Hash values, 1 Hash value identity and 1 MAC value. The Hash value and the MAC have 8 bytes and the Hash value identity is assumed to have 2 bytes. Therefore, the entire length of the message is 26 bytes and b/a is 26. Then x is 10.49 by (15). Thus, when the length of the Hash Chain is 10, the energy overhead of the broadcaster is minimal.

Taking the calculation energy consuming of RC5 as a unit, the total energy overhead of the ShortHC with different short Hash Chain lengths are shown in FIGURE 1. In FIGURE 1, we can see that the energy overhead of the broadcaster gradually increases with the Hash Chain length increased when the length is larger than 10. It was also shown that the energy consuming grows very fast when one long Hash Chain is used, whereas our ShortHC method keeps the energy consumption of the broadcaster increases in a linear way.

Here we discuss the ShortHC in the μTESLA broadcast authentication protocol, but not the point-to-point Hash Chain redeployment scenario. If the ShortHC is used in a point-to-point way to extend the Hash Chain, the computation and communication overheads

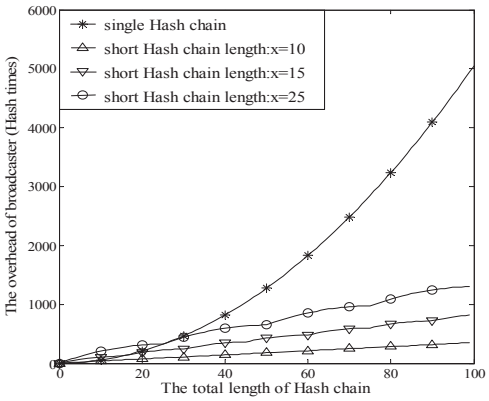

FIGURE 1. Overhead of Hash Chain with different chain length

of the broadcast node linearly increases with the number of network nodes. That is to say, the broadcast way to realize the extension of the Hash Chain is an important content of the ShortHC protocol to keep the energy consumption of the broadcaster at a constant value.

5.2. **Multi-level μTESLA based on ShortHC.** In Multi-level μTESLA [11], Liu et al. pointed out that if the life time of a key was 100 ms, a three level structure with 1000 keys of the top level was sufficient to maintain the entire network continuing to work for $10^8 s$ (more than three years). That is to say, the broadcaster has to store 3000 keys and conduct $3 \times 0.5 \times 1000 \times 1001$ times Hash computing.

In Multi-level μTESLA, if the top-level Hash Chain length is $N_1 = 10^3$, the middle layer Hash Chain value is $N_2 = 10^6$, and the lowest level Hash value is $N_3 = 10^9$. Thus, the whole overheads of the broadcast node include both the computation and communication cost

$$E_{\text{M}-\text{L}\mu\text{TESLA}} = E_{communication} + E_{computation}. \tag{16}$$

In (16), $E_{communication}$ represents the communication energy overhead defined in (17), and $E_{communication}$ represents the calculation energy overhead which will be described by (17)

$$E_{communication} = 2b_1 \sum_{i=1}^{3} \left\lfloor \frac{N_1}{n} \right\rfloor + 2b_2(N_1 + N_2), \tag{17}$$

where b_1 represents the authentication communication for the Hash Chains in different levels, and b_2 represents the authentication communications for the Hash Chains in the same level. Here, b_1 has 26 bytes as shown in message (3) and b_2 also has 26 bytes, such as the value $CDM_i = i|K_{i+2,0}|MAC_{K_i'}(i|K_{i+2,0})|K_{i-1}$ in [11]. Therefore,

$$\begin{aligned} E_{computation} = a(&4(N_1 + N_2) \\ &+ \sum_{i=1}^{3}((\tfrac{n(n+1)}{2} + 5) \cdot \lfloor \tfrac{N_i}{n} \rfloor \\ &+ \tfrac{(n+r_i)(n-r_i+1)}{2})). \end{aligned} \tag{18}$$

where n is the length of short Hash Chains and $r_i = N_i \bmod n$. When the computation energy cost of the Hash is equal to 1-byte communication energy overhead, we can obtain that $b_1 = b_2 = 26a$. The overhead of the protocol is shown in FIGURE 2.

If we don't consider the problem of the key package loss in the top level, the best length of the short Hash Chains is around 10. Thus the broadcast node achieves the minimum energy cost, as shown at the lowest point of the curves in FIGURE 2. At these lowest

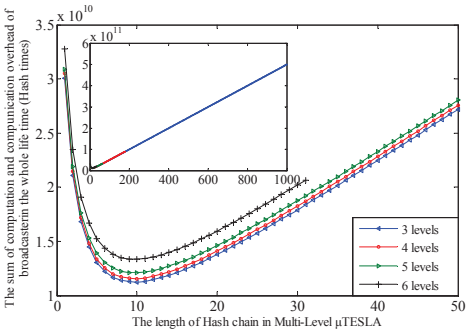

FIGURE 2. Energy overhead comparison of different levels μTESLA based on ShortHC

points, the energy consumption is 1/5 of that in 3-level μTESLA when the length of the short Hash Chain is around 100.

If the package loss in Multi-level μTESLA should be considered, then top layer should be set as a unique short Hash Chain. That is to say, a long Hash Chain in the top layer should be set to avoid the key packet loss problem when the ShortHC is used to extend the Multi-level μTESLA. Thus, we should know how many levels should be divided in the multi-layer protocol. For example, in order to ensure that there are 10^9 keys of the lowest level in an l layers Hash Chain structure, we obtain $x^l = 10^9$ where x is the length of short Hash Chain. As shown in FIGURE 2, when the level number l is 4, 5, and 6, the corresponding lengths of short Hash Chain should be 178, 63, and 32, respectively. FIGURE 2 also shows that the 3-level μTESLA protocol has the lowest overheads among different layers in Multi-level μTESLA, and the total energy overhead increases with the level of layer increased.

5.3. **ShortPK based on ShortHC.** Wang et al. proposed ShortPK [17], a famous broadcast authentication protocol based on the short public keys. The protocol can realize the real-time broadcast authentication with lightweight overheads. The protocol also needs to use the Hash Chain to solve the redeployment problem of the short public keys. Therefore, our ShortHC can improve its performances.

Firstly, the Hash Chain length is the same as the number of short public keys for the key redeployment. For example, when the life time of a short public key is 10 minutes in ShortPK, the number of short public key is $N = 6 \times 24 \times 365 = 52,560$. We keep the assumption presented in [10] that the life time of a key is $100ms$. Then the length of the corresponding Hash Chain is also N which is too large to be acceptable. Thus, our ShortHC can make the length shorter and the cost lower.

Secondly, ShortPK can use the short public key to extend the Hash Chain, and use the current Hash Chain to deploy new short public keys. However, it isn't a good solution to the security independence in the key updating, where the security of the key deployments and the Hash Chain extension are bound as a vicious cycle.

Therefore, our ShortHC can reduce the overheads of ShortPK by using the self-regeneration characteristic of the ShortHC to replace the single long Hash Chain. It can also enhance the independency between Hash Chains and public keys by extending the new short Hash Chains relying only on the current short Hash Chain rather than short public keys.

6. **Conclusion.** The length of Hash Chain limits the applications of the Hash Chain. The short Hash Chain is binding with a short security lifetime, but the long Hash Chain increases the overheads of the system. In this paper the ShortHC protocol is presented,

in which the one-way characteristic of the Hash function is used to realize on-demand and secure extension of the Hash Chain.

We proved the security of the ShortHC system, and used ShortHC to improve some broadcast authentication protocols of the wireless sensor network, which is a typical resource-constrained network. The results show that our ShortHC effectively reduces the overheads of these famous broadcast authentication protocols. We also obtain a theoretically optimal length of the Hash Chains when using our ShortHC by using the Hash Chains with the same length. The optimal length is determined by the ratio of the message communication overhead and the computation overhead, and using the Hash Chains with the optimal length in ShortHC system can achieve the minimum energy consumption in the broadcast authentication protocol.

Acknowledgments. This work was supported in part by Natural Science Foundation of China under Grand 61272074 and Natural Science Foundation of Jiangsu Province under Grand BK2011464.

REFERENCES

[1] N. Asokan, G. Tsudik, and M. Waidner, Server-supported signatures, *Proc. of 4th European Symposium on Research in Computer Security*, pp. 131-143, 1996.

[2] V. Goyal, How to re-initialize a hash chain, URL: http://eprint.iacr.org/2004/097.pdf.

[3] Q. Guan, R. Yu, S. Jiang, and C. M. Leung, Joint topology control and authentication design in mobile ad hoc networks with cooperative communications, *IEEE Transactions on Vehicular Technology*, vol. 61, no. 6, pp. 2674-2685, 2012.

[4] N. M. Haller, The S/KEY one-time password system, *Proc. of the Symposium on Network and Distributed Systems Security*, pp. 151-157, 1994.

[5] Y. C. Hu, A. Perrig, and D. Johnson, Efficient security mechanisms for routing protocols, *Proc. of Annual Symposium on Network and Distributed System Security*, pp. 57-73, 2003.

[6] Y. C. Hu, D. Johnson, and A. Perrig, SEAD: Secure efficient distance vector routing in mobile wireless ad hoc networks, *Ad Hoc Networks*, vol. 1, no. 1, pp. 175-192, 2003.

[7] Y. Hu, M. Jakobsson, and A. Perrig, Efficient constructions for one-way hash chains, *Proc. of the Third International Conference on Applied Cryptography and Network Security*, pp. 423-441, 2005.

[8] M. S. Hwang, and P. Sung, A study of micro-payment based on one-way hash chain, *International Journal of Network Security*, vol.2, no. 2, pp. 81-90, 2006.

[9] L. Lamport, Password authentication with insecure communication, *Communications of the ACM*, vol. 24, no. 11, pp. 770-772, 1981.

[10] D. Liu, and P. Ning, Efficient distribution of key chain commitments for broadcast authentication in distributed sensor networks, *Proc. of Annual Symposium on Network and Distributed System Security*, pp. 263-276, 2003.

[11] D. Liu, and P. Ning, Multi-level μTESLA: Broadcast authentication for distributed sensor networks, *ACM Transactions in Embedded Computing Systems*, vol. 3, no. 4, pp. 800-836, 2004.

[12] M. E. Mahmoud, and X. Shen, FESCIM: Fair, efficient, and secure cooperation incentive mechanism for multi-hop cellular networks, *IEEE Transactions on Mobile Computing*, vol. 11, no. 5, pp. 753-766, 2012.

[13] M. Ohkubo, K. Suzuki, and S. Kinoshita, A cryptographic approach to 'Privacy-friendly' tags, *Proc. of RFID Privacy Workshop*, pp. 77-83, 2003.

[14] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and J. D. Tygar, SPINS: Security protocols for sensor networks, *Proc. of the 7th Annual International Conference on Mobile Computing and Networking*, pp. 189-199, 2001.

[15] R. L. Rivest, and A. Shamir, PayWord and MicroMint: Two simple micropayment schemes, Proceedings of International Workshop of Security Protocols," *Proc. of International Workshop of Security Protocols*, vol. 1189, pp. 69-87, 1997.

[16] I. Syamsuddin, T. Dillon, E. Chang, and S. Han, A survey of RFID authentication protocols based on hash-chain method, *Proc. of Third International Conference on Convergence and Hybrid Information Technology*, vol.2, pp. 559-564, 2008.

[17] R. Wang, W. Du, and P. Ning, ShortPK: A short term public key scheme for broadcast authentication in sensor networks, *ACM Transactions on Sensor Networks*, vol. 6, no. 9, 2010.
[18] F. Ye, H. Luo, S. Lu, and L. Zhang, Statistical en-route filtering of injected false data in sensor networks, *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 4, pp. 839-850, 2005.
[19] C. Yu, Y. Tsou, C. Lu, and S. Kuo, Constrained function-based message authentication for sensor networks, *IEEE Transactions on Information Forensics and Security*, vol.6, no.2, pp. 407-425, 2011.
[20] H. Zhang, and Y. Zhu, Self-updating hash chains and their implementations, *Proc. of 7th International Conference on Web Information Systems Engineering*, vol. 4255, pp.387-397, 2006.
[21] Y. Zhao, and D. Li, An improved elegant method to re-initialize hash chains," URL http://eprint.iacr.org/2005/011.pdf.