SECURITY AND COMMUNICATION NETWORKS
Security Comm. Networks 2015; **8**:1026–1039
Published online 22 August 2014 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/sec.1057

RESEARCH ARTICLE

Efficient self-healing group key management with dynamic revocation and collusion resistance for SCADA in smart grid

Rong Jiang[1,2]*, Rongxing Lu[3], Jun Luo[1], Chengzhe Lai[2,4] and Xuemin (Sherman) Shen[2]

[1] School of Computer, National University of Defense Technology, Changsha 410073, China
[2] Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, Ontario N2L 3G1, Canada
[3] School of Electrical and Electronics Engineering, Nanyang Technological University, Singapore, Singapore
[4] State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an 710071, China

ABSTRACT

In this paper, in order to simultaneously resolve the transmission security and availability in Supervisory Control And Data Acquisition (SCADA) group communications, we propose a robust and efficient group key management scheme, called LiSH+, which is characterized by developing a secure self-healing mechanism with *t*-revocation and collusion resistance capability. A dual direction hash chain is utilized to guarantee the backward secrecy and forward secrecy of group key. A novel self-healing mechanism is constructed to ensure availability of the group member in case of devices failure and prevent the collusive users from exploiting the group key in the proposed scheme. In addition, the compromised users can be revoked from the group dynamically by broadcasting message. Detailed security analysis shows that the proposed LiSH+ scheme meets the requirements of group communication and is secure in terms of *t* user collusion-free. Performance evaluation also demonstrates its efficiency in terms of low storage requirement and communication overheads. Copyright © 2014 John Wiley & Sons, Ltd.

KEYWORDS
smart grid; SCADA; security; availability; group communication

***Correspondence**
Rong Jiang, School of Computer, National University of Defense Technology, Changsha 410073, China.
E-mail: jiangrong@nudt.edu.cn

1. INTRODUCTION

Power grid plays a vital role in modern society. However, the development of traditional power grid cannot keep pace with the industrial and social advancements [1]. An unstable and unreliable power grid will bring great inconvenience to our daily life and caused huge economic loss. For example, the Northeast blackout of 2003 in north America affected an estimated 10 million people in Ontario and 45 million people in eight US states and caused billions of loss [2]. The India blackout in July 2012 affected more than 60 million people (about 9% of the world population) and plunged 20 of India's 28 states into darkness [3]. Indeed, the traditional power grid, which is surprisingly still grounded on the design more than 100 years ago, is no longer suitable for today's society [4]. With the development of information system and communication technology, the traditional power grid has been changing into a new paradigm—smart grid—to improve its reliability. In smart grid, due to the widely used bi-directional communications, data collected from different part of power grid can be shared promptly among various departments. Many new applications, such as real-time electricity price, can be implemented to facilitate the control of power delivery and distribution. Thus, smart grid has attracted great attention not only from government but also from the industry and academia for its high fidelity power-flow control, self-healing, energy reliability, and energy security [5,6].

According to the conceptual model of National Institute for Standards and Technology (NIST), four main components, that is, generation, transmission, distribution, and customer, feature two-way power and information flows in smart grid. Supervisory Control And Data Acquisition (SCADA) systems are used to monitor and control sensitive processes and physical functions in the electricity distribution, transmission, and generation environments, as shown in Figure 1. In order to monitor and control

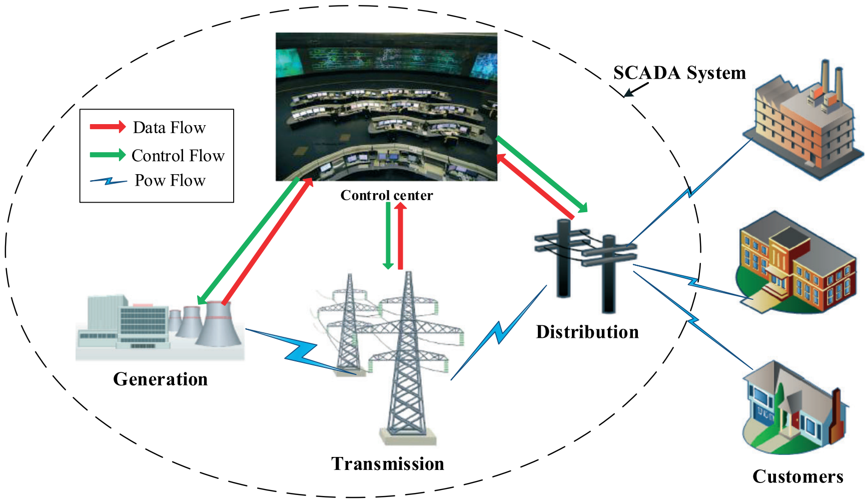


Figure 1. SCADA system in smart grid architecture.

sensitive operation and physical functions in power grid, millions of sensors and remote equipment are distributed over the whole system. These devices communicate with each other and form a variety of communication networks to achieve diversified applications, such as angle and frequency monitoring, voltage and voltage stability monitoring, post-mortem analysis, improved state estimation, and steady-state benchmarking. The SCADA systems enable the operators at the control center to monitor and gain measurement from remote devices and to manipulate settings on remote equipment. For example, the control center communicates with remote terminal units (RTUs) located in substations to gain monitoring data and issue control commands. Then, the RTUs connect to digital relays in substation to collect status indicators for circuit breaker and collect analog measurements from current or potential transformers for the control center [7]. Due to the huge number of resource-constrained RTU, the group communication, which is both bandwidth-efficient and energy-efficient, is an essential part for time-critical SCADA systems. For example, in SCADA systems, the control center needs to periodically collect field devices status by broadcasting a status scan request to remote sites and collect measured values of field devices by a measured value scan request. Furthermore, the control center also broadcasts messages such as emergent shutdown messages or set-the-clock-time messages to multiple RTUs [8].

However, due to the specific operational environment in smart grid, a SCADA system has constrains as follows [9–11]:

- **Limited resource capability**: The RTUs in an embedded system have low computational microprocessor and limited memory.
- **Low-rate data transmission**: The SCADA systems have been set up for a long time, the communication channels between entities will remain low bandwidth in the short run. Generally speaking, the bandwidth of a SCADA communication is from 300 to 19 200 baud rate [12].
- **Real-time processing**: The data collected from the remote field and command from the control center should be transmitted real time. Any latency or loss may have adverse effects on the electrical power grids. Specifically, the time delay for states and alarms must not be more than 0.900 s and SCADA transactions should have a time latency of less than 0.540 s [12]. The availability of a SCADA system is considered to be one of the most critical issues, because an unavailable SCADA system can cause large-scale physical damage or even threaten human life. Countermeasures designed to provide improved integrity or confidentiality need to be implemented in such a way that the availability of the system is not decreased [13].
- **Physical insecurity**: Most of the devices in SCADA locate far away from the residential areas so that the monitoring devices have to be remote from the central control center. It is hard to physically protect them.

In addition, SCADA systems used to be isolated and not designed with public access in traditional power grid, and they are typically lack even rudimentary security mechanism. As the increasing demand for improving transmission reliability and efficiency, SCADA systems are increasingly using standard protocols, such as DNP3 and IEC 60870-5 [14]. However, the use of standard protocols, combined with increased interconnectivity with other networks, makes SCADA system susceptible to different types of cyber attacks. An actual attack on SCADA systems is reported, and the attacker conducts a series of electronic attack after his or her job application have been rejected [8]. Therefore, it is critical to protect communications among SCADA devices against cyber attacks using security mechanisms. Key management is one of the fundamental security mechanisms to guarantee the communication security in SCADA systems. The other security mechanisms such as secure routing, secure localization, authenticity, and integrity are built upon the secure key management. Because group communications are indispensable in SCADA as mentioned before, group key is one of the most important key management paradigms that should be well-designed [15].

In the literature, the key management schemes can be classified into three categories [16], that is, Diffie–Hellman algorithm extended contributory key management, computational number theoretic approach and logical key hierarchy (LKH) approach. The LKH-based scheme, which is designed for conventional wired networks at first [17,18], can reduce the rekeying overhead to $O\log(N)$ by building a tree of key encryption keys. In the last few years, many key management schemes have been developed for resource-constrained networks, such as wireless sensor networks [19–21] and VANET [22].

Several secure communication schemes have been proposed for SCADA systems [9,10,12,13,23]. However, these schemes suffer from availability problem when some remote units break down. In order to simultaneously resolve the transmission availability and security in resource-constrained SCADA system, we propose a robust and efficient *Li*mited *S*elf-*H*ealing key distribution [24], called LiSH, with collusion-resistant and revocation capability for SCADA group communication. In LiSH, the group member can still keep availability even if the main device such as key distribution center (KDC) has broken down. LiSH scheme meets all the security requirements for group communication such as forward and backward security. Although the overall performance of LiSH has many advantages compared to previous schemes, it can be less efficient during the multicast communication process and cannot revoke compromised users dynamically. Therefore, in this paper, we propose an enhanced scheme LiSH+ to further improve the performance. The main contributions of this paper are twofold:

- Firstly, we construct a bivariate polynomial to further reduce the storage cost in resource-constrained RTUs. In our LiSH+ scheme, the store requirement is related to the degree of the bivariate polynomial. Performance evaluation results show that our LiSH+ scheme needs less storage overhead than the previous schemes need in the same condition.
- Secondly, because the user nodes may be compromised at any time in reality, our LiSH+ scheme supports to eliminate compromised users as soon as the compromised users are detected by means of some intrusion detection systems to avoid greater harm to the power system. The dynamic revocation mechanism improve the security level of our proposed LiSH+ scheme.

The remainder of the paper is organized as follows. In Section 2, we briefly review relate works. In Section 3, we introduce the system model, security model, and design goal. In Section 4, we present our proposed LiSH+ scheme in detail, followed by security and efficiency analysis in Section 5 and Section 6, respectively. Finally, we draw our conclusion in Section 7.

2. RELATED WORKS

Security for SCADA system in smart grid is a very active topic. SKE [10] is an elementary key management scheme for the SCADA system with low-cost security. However, efficient multicast and broadcast, which are essential in power systems, are not supported in this scheme. SKMA proposed in [13] also does not support broadcast communication. ASKMA [23] uses a LKH to support broadcast communication and multicast communication, but it may be less efficient during the multicast communication process. ASKMA+ [9] reduces the number of stored keys and provides efficient multicast and broadcast communication. However, it does not satisfy the availability requirement. Choi *et al.* [12] propose a hybrid key management architecture for robust SCADA systems, which supports replace protocol for availability, but the affected devices stops working during the replacement.

Self-healing group key distribution schemes, which enable large and dynamic groups of users to establish group keys over unreliable network for secure multicast communication, can improve the robustness of the system. The formal definitions of self-healing are first proposed by Staddon *et al.* [25] using entropy theory. Blundo *et al.*[26] point out that some of the constructions in [25] are blemished. They present some new constructions based on rectification of previous definitions. Hong [27] reduces storage and communication overhead according to the lower bound defined in [26]. Although these self-healing schemes are unconditional secure, many of them would incur heavy overload. Jiang *et al.*[28] introduce dual directional hash chain to construct a lightweight self-healing scheme with time-limited node revocation, which requires much less communication and storage overhead, but unfortunately, it cannot resist collusion attack. Dutta *et al.* propose several efficient computationally secure solutions [29,30] that can greatly reduce resource cost and achieve forward and backward security, yet these schemes would lead to session keys being exposed [31–33]. Dutta *et al.* further replace m mask polynomials with a bivariate t-degree polynomial [30]. In this way, the storage and communication overhead can both be considerably lowered to $O((t+1)\log q)$. Although it is efficient in term of cost due to the elegantly designed mask polynomial, a collusion attack called *sandwich attack* may happen here. To prevent sandwich attack, Du *et al.* introduce a secret random number for each session in [34]. Only if the random number is not cracked, attackers still get nothing about the session key. Another technique used in [34] is the self-healing mechanism. They introduce a private number for each session to produce a self-healing chain. Only legal nodes with appropriate time window can decrypt lost keys. However, their self-healing mechanism exits some security risks. The current session key may be leaked directly to attackers in case of both private numbers and session keys in two neighboring past sessions are cracked. (This can be easily achieved by capturing nodes in two adjacent sessions, and we will elaborate on it in Section 5.) Besides, newly joined users collaborating with the revoked users, whose life cycles overlap, are still able to recover session keys they are not entitled to.

The main concern of self-healing property is that it can only be used by user nodes to validate and decrypt messages collected in previous sessions, but it cannot be used in the future sessions yet due to lack of appropriate session keys [35]. In [24], we introduce a delay of l sessions between distribution of a session key and usage of this key for the protection of multicast group communication, which improves the availability of group key. In this paper, we further reduce the storage overhead in each user node by using a bivariate polynomial and design a dynamic revocation mechanism.

3. MODELS AND DESIGN GOALS

In this section, we present the system model, security model, and identify our design goal.

3.1. System model

There are three main components in SCADA system: *human–machine interface*, *master terminal unit*, and *remote terminal unit*. These entities communicate with each other and constitute a hierarchical structure as shown in Figure 2. The explanation of each entity is described as follows:

- Human–machine interface (HMI): The HMI is the device that is used to interact with a SCADA system. The HMI for SCADA system has been developed supporting a graphic interface, such as Web browsers.
- Master terminal unit (MTU): MTU is the superior in a communication hierarchy and provides supervisory control of RTUs. In smart grid, there is a central MTU which communicates with sub-MTUs and RTUs. The MTU and sub-MTUs have reasonable computational resources and the control messages and data that they transmit are more important, so a public key cryptosystem can be applied among them [12]. In each MTU and sub-MTU, there is a particular computer called KDC which takes charge of key generation and rekeying. The MTU is assumed to be physically secure, while sub-MTUs may be located remotely from the control center. The sub-MTUs cannot be compromised, but they may suffer DoS attack or physical damage especially the vital part such as KDC. When a sub-MTU breaks down, the device should be repaired or replaced in a limited time.
- Remote terminal unit (RTU): RTUs are devices composed of a microprocessor that controls sensors and actuators that interact with the physical environment [13]. The RTUs have limited resource. The location of RTUs can often be remote to the main corporate offices. As a result, it is difficult to physically protect the units.

In smart grid, the entities of SCADA system are set up in advance. Moreover, available communication channels between entities are known in advance as well. The

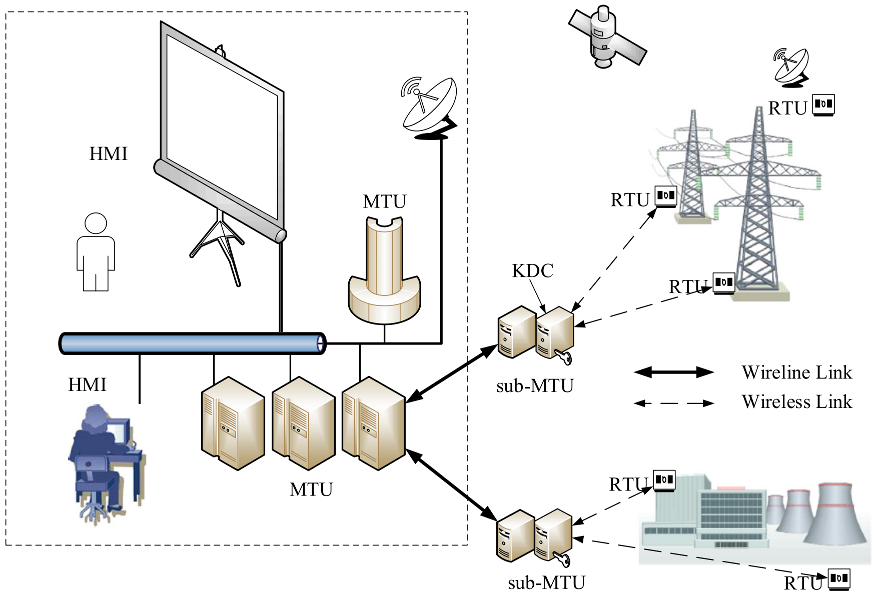


Figure 2. A simple SCADA system architecture.

Table I. Notation used in this paper.

Notation	Definition
KDC	Key distribution center
U	The finite set of all user nodes in the network
n	Total number of user nodes in the network
u_i	The ith user in U_n
m	The number of sessions in the group lifetime
s_j	The jth session
$\mathbb{F}_q$	A field of order q
q	A large prime number, $q > n$
$h(x, y)$	A t-degree bivariate polynomial
c_j	The jth secret random number broadcasted in s_j
d_j	The jth private number corresponding to s_j
b_j	Broadcast polynomial in s_j
T_j	Self-healing set in s_j
t_k	An element in set T_j corresponding to s_k
L_j	A set of user IDs which are irrelevant to our group
R_j	A set of users revoked in and before s_j
B_j	A set of messages broadcasted in s_j
J_j	A set of new users joining the group in s_j
SK_j	The jth session key
fk^j	The jth forward key in the forward hash chain
bk^j	The jth backward key in the backward hash chain

communication model between a MTU (sub-MTU) and RTUs is master–slave model. A MTU (sub-MTU) communicates a number of RTUs through wireless channel, since the RTUs always locate remote from the control center and are difficult to reach.

For clarity, we list the notation used throughout the rest of this paper in Table I.

3.2. Security model

In our security model, we focus on how to guarantee the security of communication between RTUs and the control center. We consider an outside adversary, which could compromise and control some RTUs. Specifically, the following security requirements on group communication should be desired (Let $\mathcal{J}_j$ be a set of new users joining the group in session j, and $\mathcal{R}_j$ a set of users revoked in and before session j):

- *Group confidentiality:* users that are not the part of the group should not have access to any key that can decrypt any data broadcast to the group.
- *Backward secrecy:* given any set $\mathcal{J}_j$, it is computationally infeasible for the users $u_i \in \mathcal{J}_j$ colluding together to recover any of past session keys $SK_1, \cdots, SK_{j-1}$, that is, new user joining the group should not be able to know any previous keys so that he or she cannot decrypt previous transmitted message.
- *Forward secrecy:* given any set $\mathcal{R}_j$, it is computationally infeasible for the users $u_i \in \mathcal{R}_j$ colluding together to recover any of subsequent session keys $SK_j, \cdots, SK_m$, that is, a user should not be allowed to know future keys after he or she leaves the group.
- *t-collusion-resistant:* given any set R_i, and any set $\mathcal{J}_{j+1}$, such that $\left|\mathcal{R}_i \bigcup \mathcal{J}_{j+1}\right| \leq t$, it is computationally infeasible for a colluding coalition $\mathcal{R}_i \bigcup \mathcal{J}_{j+1}$ to recover any of session keys $SK_i, \cdots, SK_j$.

3.3. Design goal

Our design goal is to propose a robust and efficient group key distribution scheme to satisfy the above security objectives. In particular, we will achieve

- *Availability:* The proposed LiSH+ scheme should achieve availability in SCADA communications, so that even if some sub-MTUs have broken down, the whole system can keep on working in a limited time (the broken sub-MTUs should be repaired or replaced in a limited time).
- *Efficiency:* The proposed LiSH+ scheme should also minimize the computation, memory, communication, and energy costs in group communication. It shall take into account RTUs limitations, because the RTUs run industry standard protocols on 16 bit microprocessors with 8 kB of RAM (working memory), and 64 kB of EPROM (persistent memory) [13].

4. THE PROPOSED LISH+ SCHEME

In this section, we propose our LiSH+ scheme, which consists of five parts: initialization, rekeying, self-healing mechanism, adding new member nodes, and re-initialization mechanism. The basic procedure of the proposed scheme is shown in Figure 3.

As the RTUs are located remotely from the control center, they are physically insecure and susceptible to attacks. Therefore, the keys stored in the RTUs need to be updated periodically. Our scheme focuses on the group key management between MTU (sub-MTU) and RTUs. A group consists of a KDC which is a part of MTU (sub-MTU) and n user nodes (RTUs). Revoked users are never allowed to rejoin the network with the same ID afterwards. The lifetime of the network is divided into m periods with particular duration, where each period is called a session and s_j denotes the jth session. The key distribution period is an important parameter in the proposed scheme. If the session key is used over a longer time span, it can increase the possibility of being cracked. Besides, it is not easy even to change the key too often because the frequent key changing causes the increase of the network traffic and communication failures. This issue is more significant in SCADA systems because of its unique characteristics of data size and communication frequency. Thus, it is important to find an appropriate period to update the session keys. The QoS function to find the period is proposed in [36] to address this problem. All operations are taking place in a finite field $\mathbb{F}_q$, where q is a large prime number such as 64-bit or 128-

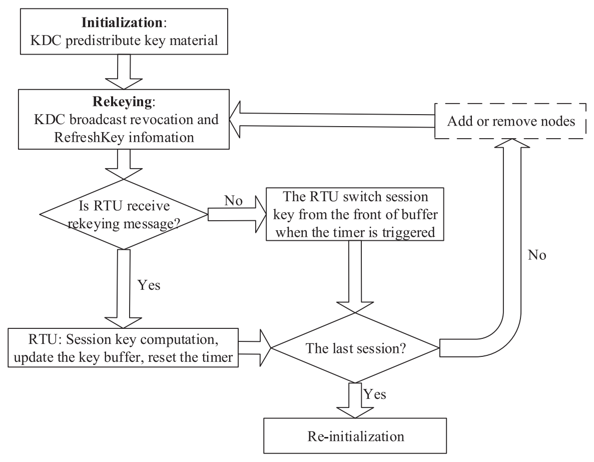

Initialization: KDC predistribute key material
Rekeying: KDC broadcast revocation and RefreshKey infomation
Add or remove nodes
Is RTU receive rekeying message?
No
The RTU switch session key from the front of buffer when the timer is triggered
No
Yes
RTU: Session key computation, update the key buffer, reset the timer
The last session?
Yes
Re-initialization

Figure 3. The basic procedure of the proposed scheme.

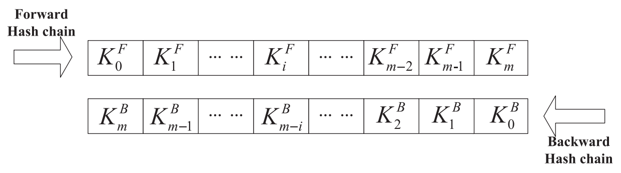

Forward Hash chain
K_0^F K_1^F ··· ··· K_i^F ··· ··· K_{m-2}^F K_{m-1}^F K_m^F
K_m^B K_{m-1}^B ··· ··· K_{m-i}^B ··· ··· K_2^B K_1^B K_0^B
Backward Hash chain

Figure 4. The structure of DDHC.

bit integer. Before going into the details, we first recall the dual directional hash chains, which serve as the basis of the proposed protocol.

4.1. Dual directional hash chains

One-way hash chain is a sequence of hash values which are computed by a hash function $H(\cdot)$ iteratively based on a secret seed x_0. The relation of the hash values in one-way hash chain is shown as follows:

$$\begin{cases} x_1 = H(x_0) \\ x_2 = H(x_1) = H^2(x_0) \\ \quad\vdots \\ x_n = H(x_{n-1}) = \cdots = H^n(x_0) \end{cases}$$

The most attractive property of one-way hash chain is that, given any x_i, it is easy to compute any x_j $(j > i)$ with $H_j = H^{j-i}(x_i)$, but it is computationally infeasible to calculate x_k $(k < i)$.

A dual directional hash chain (DDHC) is based on two one-way hash chains with equal length, one of which is called forward hash chain and the other is called backward hash chain. For example, given two random key seeds FK and BK, a dual directional hash chain can be derived, as shown in Figure 4, where the forward hash chain K^F is

$$\left\{K_0^F = FK, K_1^F = H\left(K_0^F\right), \cdots, K_m^F = H^m\left(K_0^F\right)\right\}$$

the backward hash chain K^B is

$$\left\{K_m^B = H^m\left(K_0^B\right), \cdots, K_1^B = H\left(K_0^B\right), K_0^B = BK\right\}$$

The dual directional hash chain is used to construct session keys in our proposed scheme.

4.2. Description of LiSH+

4.2.1. Initialization.

The KDC picks randomly a bivariate polynomial

$$h(x,y) = \sum_{0 \le i,j \le t} a_{ij} x^i y^j \bmod q$$

from $\mathbb{F}_q[x,y]$. It also creates m random numbers $d_1, d_2, \cdots, d_m \in \mathbb{F}_q$ for self-healing. The group key used in our scheme is a session key, and it is computed by two parts: dual directional hash chain value and a secret number c_j, which are generated as follows.

The dual directional hash chain is computed by KDC before it is transmitted to the group member nodes. First of all, the KDC randomly picks two initial key seeds: the forward seed FK and the backward seed BK. It repeatedly applies the same one-way hash function $H(\cdot)$ to each seed during the pre-processing time to produce two hash chains of length m. For $1 \le j \le m$, the dual directional hash chain can be generated as

$$fk^j = H^j(FK), bk^j = H^{m-j+1}(BK) \quad (1)$$

The secret number c_j corresponding to s_j chosen by the KDC is released in section s_j, and can be retrieved by each legal user through the rekeying message broadcasted in session s_j as follows:

$$c_j = \frac{b_j(u_i) - h_j\left(u_i, fk^j + bk^j\right)}{v_j(u_i)} \mod q \quad (2)$$

where $b_j(\cdot)$ is the broadcast polynomial which is constructed by the KDC in s_j, u_i is the ID of user i, and $v_j(\cdot)$ is the revocation polynomial which can only retrieve by the legal nodes. The detailed construction of the broadcast polynomial and the revocation polynomial is given in Section 4.2.2.

Thus, the jth session key in our scheme can be calculated as

$$\begin{aligned} SK_j &= fk^j + c_j \times bk^j \\ &= H^j(FK) + c_j \times H^{m-j+1}(BK) \end{aligned} \quad (3)$$

For $1 \le j \le m$, each user u_i with lifetime from s_1 to s_2 $(1 \le s_1 < s_2 \le m)$ is assigned the polynomial

$$h(u_i, y) = a_{0,0} + a_{1,0}u_i + a_{0,1}y + \cdots + a_{t,t}u_i^t y^t \mod q$$

set $\mathcal{D}_i = \{d_{s_1}, \cdots, d_{s_2}\}$, two key seeds: the forward key seed $fk^{s_1} = H(FK)^{s_1}$ and backward key seed $bk^{s_2} = H(BK)^{m-s_2+1}$ as its private secrets. The user u_i receives these secret information via secure channel between KDC and the user itself. At time t_{init}, the KDC sends the following message to u_i:

$$KDC \to u_i : \{l \| T_{refresh} \| \mathcal{D}_i \| h(u_i, y) \| fk^{s_1} \| bk^{s_2}\}$$

where l is the length of key buffer and $T_{refresh}$ is the rekeying period. When the user u_i receives the Init message, it processes this message according to Algorithm 1.

Algorithm 1: Initialization

```
1 Function Init()
2 {
3 if (receiving InitGroupKey message) then
4     Decrypt the message to get
      $\{l, T_{refresh}, \mathcal{D}_i, h(u_i, y), fk^{s_1}, bk^{s_2}\}$;
5     Allocate a key buffer with length
      $l : (SK[1], \cdots, SK[l])$;
6     Set RefreshKey timer to $T_{refresh}$;
7     Store $\mathcal{D}_i, h(u_i, y), fk^{s_1}, bk^{s_2}$;
8 end if
9 }
```

4.2.2. Rekeying.

At the beginning of each session, the KDC periodically discloses the secret number c for this session. In addition, it constructs a self-healing set $\mathcal{T}$ for legal users to compute session key in their lifetime and a revocation set $\mathcal{R}$ to expel some illegal users. The construction of self-healing set $\mathcal{T}$ and revocation set $\mathcal{R}$ is described as follows.

As the SCADA system is typically employed for real-time applications of power control and sensitive data collection, stored keys should be prudently used in future sessions in consideration of security. Moreover, as RTUs are resource constrained, a self-healing set with growing size of elements may cause heavy communication overhead. Thus, from the point of view of security and practical utility, we limit the size of self-healing set to l sessions, for example, if the current session is j, then its self-healing set is $\mathcal{T}_j = \{t_{j+1}, t_{j+2}, \cdots, t_{j+l}\}$.

To avoid the secrecy exposed, a novel self-healing mechanism is proposed. Let the current session be s_j and the corresponding broadcasting secret be c_j. The later secret c_k in $s_k(j < k \le (j + l))$ is concealed in this way:

$$\begin{cases} F_k = H_1(d_{k-1} + c_{k-1}) \\ F'_k = H_1(F_k + d_k) \\ F''_k = H_1(F'_k + c_j) \\ t_k = c_k - F''_k \end{cases} \quad (4)$$

where $H_1(\cdot)$ is a one-way hash function and t_k is the encrypted secret that will be published openly. Each of the later session secrets $c_k(j < k \le j + l)$ can be produced similarly.

Let $\mathcal{R}_j = \{r_1, r_2, \cdots, r_{\omega_j}\} \in \mathbb{F}_q$ be the set of users all revoked in and before s_j, in which $|\mathcal{R}_j| = \omega_j \le t$. In s_j, the KDC calculates the forward key fk^j and backward key bk^j and then produces mask polynomial $h\left(x, kf^j + kb^j\right)$. As to the revocation polynomial, its common expression is a ω_j-degree polynomial

$$v_j = (x - r_1)(x - r_2) \cdots \left(x - r_{\omega_j}\right) \mod q \quad (5)$$

where $r_1, r_2, \cdots, r_{\omega_j} \in \mathcal{R}_j$. However, it suffers from the attack against revocation polynomial [33]. We modify the design by attaching a set of random users to the original revocation polynomial who are unrelated to our system. A new t-degree revocation polynomial is produced as

$$v_j = (x - r_1) \cdots \left(x - r_{\omega_j}\right)\left(x - r_{\omega_j+1}\right) \cdots (x - r_t) \mod q \quad (6)$$

in which $\mathcal{L}_j = \{r_{\omega_j+1}, \cdots, r_t\}$ is a set of irrelevant users and $|\mathcal{L}_j| = t - \omega_j$. After generating a random secret number c_j from $\mathbb{F}_q$, the KDC can compute the broadcast polynomial:

$$b_j = v_j c_j + h\left(x, kf^j + kb^j\right) \mod q \quad (7)$$

For the jth rekeying, the KDC broadcasts RefreshKey message with $\mathcal{T}_j$ ($j = 1, \cdots, m - l$) to all user nodes at the beginning of section j:

$$B_j \rightarrow all : \left\{ \mathcal{R}_j \bigcup \mathcal{L}_j \,\|\, \mathcal{T}_j \,\|\, b_j \right\}$$

When non-revoked user u_i receives the jth session broadcast message B_j, it processes this message according to the Algorithm 2 (lines 3–20). It checks whether the revoked set is changed. If not, it switches the session key SK_j from the front of the key queue Q, calculates SK_{j+l}, and resets the timer of refresh key. Otherwise, it will calculate the new session key SK_j and update the key buffer. It evaluates revocation polynomial $v_j(u_i)$, computes the forward key fk^j and backward key bk^j, obtains $h\left(u_i, kf^j + kb^j\right)$ and then recovers

$$c_j = \frac{b_j(u_i) - h_j\left(u_i, kf^j + kb^j\right)}{v_j(u_i)} \mod q \tag{8}$$

Finally, user u_i computes the jth session key as

$$SK_j = fk^j + c_j \times bk^j \tag{9}$$

After that, u_i can iteratively obtain all c_k before s_{j+l} in its legal lifetime by self-healing set:

$$\begin{aligned} c_{j+1} &= t_{j+1} + F''_{j+1} \\ &= t_{j+1} + H_1\left(H_1\left(H_1\left(d_j + c_j\right) + d_{j+1}\right) + c_j\right) \end{aligned} \tag{10}$$

$$\begin{aligned} c_{j+2} &= t_{j+2} + F''_{j+2} \\ &= t_{j+2} + H_1\left(H_1\left(H_1\left(d_{j+1} + c_{j+1}\right) + d_{j+2}\right) + c_j\right) \end{aligned} \tag{11}$$

$$\vdots$$

$$c_k = t_k + H_1\left(H_1\left(H_1\left(d_{k-1} + c_{k-1}\right) + d_k\right) + c_j\right) \tag{12}$$

Then, the corresponding SK can be calculated by Equation (3) and will be pushed into the key queue.

Algorithm 2: Rekeying

```
1  Function Rekeying()
2  {
3  if (receiving broadcast message $B_j$) then
4      if (the revoked set is not changed) then
5          switch the session key from the key buffer $SK[1]$;
6          if (the number of remain session of the user is larger than l) then
7              calculate session key $SK_{j+l}$ and push it into the buffer;
8
9      else
10         evaluate revocation polynomial $v_j(u_i)$, obtain $h\left(u_i, kf^j + kb^j\right)$;
11         recover $c_j$ and compute the $j$th session key $SK_j$;
12         if (the number of remain session of the user is larger than 1) then
13             for ($i = 1; i \leq l; i++$) do
14                 $c_{j+i} = t_{j+i} + H_1\left(H_1\left(H_1\left(d_j + c_j\right) + d_{j+i}\right) + c_j\right)$;
15                 $SK_{j+i} = fk^{j+i} + c_{j+i} \times bk^{j+i}$;
16                 push $SK_{j+i}$ into the buffer $SK[i]$;
17             end for
18
19     end if
20     reset the timer;
21 else
22     if (RefreshKey timer is triggered) then
23         switch the session key from buffer $SK[1]$;
24         reset the timer;
25
26 end if
27 }
```

4.2.3. Self-healing mechanism.

When a sub-MTU especially the KDC breaks down because of attacks or natural disasters, the RTUs can keep on working for at most l sessions. When the timer of refresh key is triggered and the user does not received the rekeying message, it will switch the session key automatically from the front of the key queue Q and reset the timer of refresh key according to the Algorithm 2 (line 22–24). This procedure will proceed until the buffer is empty or the user receives the broadcast message again.

After the broken sub-MTU is repaired or replaced, a legal user u_i with lifetime from s_1 to s_2 can update all of its session keys in the buffer. Without loss of generality, we consider that a user u_i loses its connection with the sub-MTU from s_i to s_j $\left(s_1 < s_i < s_j < s_{j+l} < s_2\right)$ for some reason. u_i receives the jth broadcast message B_j successfully, and recovers c_j and the session key SK_j. Suppose the secret c_{k-1} ($j < k - 1 < k \leq j + l$) in s_{k-1} has been restored, u_i first evaluates F''_k by Equation (4). Then, the lost secret c_k can be retrieved: $c_k = t_k + F''_k$. Thus, the lost key SK_k for s_k can be computed by Equation (3). After all the keys are calculated, u_i pushes them into the key queue.

4.2.4. Adding new member nodes.

When user u_i (lifetime from s_1 to s_2) tries to join the existing group, it firstly requests the KDC's authentication. After verifying its identification, KDC encrypts the following items via secure channel between KDC and the new user and then sends them to u_i:

$$KDC \rightarrow u_i : \left\{ t \,\|\, T_{refresh} \,\|\, \mathcal{D}_i \,\|\, h(u_i, y) \,\|\, fk^{s_1} \,\|\, bk^{s_2} \right\}$$

After that, the user u_i can compute the session by the broadcast message from session s_1 to s_2.

4.2.5. Re-initialization mechanism.

If there are more than t revoked users, the broadcast polynomial may be retrieved. Our proposed LiSH+ scheme will be re-initialization in order to avoid session key leakage. In addition, the scheme will be re-started when the session keys are used out.

4.3. Dynamic revocation mechanism

In reality, the user nodes may be compromised at any time. As a result, as soon as the compromised users are detected by means of some intrusion detection systems, they must be revoked from the group immediately in order to avoid greater harm to the system. Thus, a dynamic revocation mechanism is essential. In this session, we give the details on how to dynamically eliminate the compromised users from the group. As a result, the node compromised attack can be alleviated in our LiSH+ scheme.

The rekeying procedure of the proposed scheme with dynamic revocation mechanism is shown in Figure 5. The KDC will broadcast a rekeying message to update the session key from SK_j to SK'_j whenever it detects the compromised users:

$$B_j \to all : \left\{R\_rekeying \bigcup \mathcal{R}'_j \bigcup \mathcal{L}'_j \left\| \mathcal{T}'_j \right\| b'_j\right\}$$

where *R_rekeying* indicates that it is a rekeying message in the middle of the session *j*. When a non-revoked user receives the *R_rekeying* message, it will calculate the new session key SK'_j and update the key buffer. It evaluates revocation polynomial $v'_j(u_i)$, computes the forward key fk^j and backward key bk^j, obtains $h\left(u_i, kf^j + kb^j\right)$ and then recovers

$$c'_j = \frac{b'_j(u_i) - h_j\left(u_i, kf^j + kb^j\right)}{v'_j(u_i)} \quad \text{mod } q \tag{13}$$

Finally, user u_i computes the jth session key as

$$SK'_j = fk^j + c'_j \times bk^j \tag{14}$$

Then, u_i calculates all SK_k $(j < k \leqslant j+l)$ in its legal lifetime and push them into the key buffer.

5. SECURITY ANALYSIS

In this section, we give detailed security analysis of LiSH+ scheme to verify that our scheme satisfies the requirements of self-healing key distribution with revocation capability. Further analysis proves that LiSH+ assures forward and backward security, collusion-free property.

Theorem 1. *LiSH+ is a session key distribution with privacy and achieves self-healing with time-limited t-collusion-free capability.*

Proof.

(1) A session key distribution with privacy: For a non-revoked user u_i in s_j, his or her session key is determined by forward key fk^j, backward key bk^j, and a broadcast secret c_j. The first two elements fk^j and bk^j are covertly assigned to the user beforehand. And random number c_j is encrypted in the broadcast

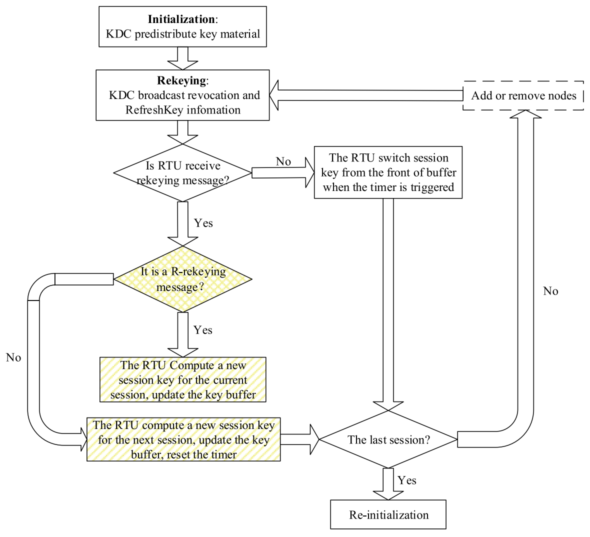

Initialization:
KDC predistribute key material
Rekeying:
KDC broadcast revocation and RefreshKey infomation
Add or remove nodes
Is RTU receive rekeying message?
No
The RTU switch session key from the front of buffer when the timer is triggered
Yes
It is a R-rekeying message?
No
Yes
No
The RTU Compute a new session key for the current session, update the key buffer
The RTU compute a new session key for the next session, update the key buffer, reset the timer
The last session?
Yes
Re-initialization

Figure 5. The rekeying procedure of the proposed scheme with dynamic revocation mechanism.

message. Thus, it is impossible for any user to evaluate the session key only by private secrets fk^j and bk^j or only by the broadcast message.

(2) Self-healing: As described in Section 4, our self-healing set $\mathcal{T}_j$ is a forward key chain. Only legal users who correctly decrypt the current session key can restore later session keys in its lifetime.

(3) Time-limited node revocation: A user u_i who is operating from s_2 to $s_3 \, (1 < s_2 < s_3 < m)$ has no access to session keys out of his or her lifetime. Due to the computationally irreversible property of one-way hash function, u_i cannot compute backward secret $bk^{s_4} \, (s_3 < s_4 \leq m)$ and forward secret $fk^{s_1} \, (1 \leq s_1 < s_2)$. Therefore, it is infeasible for u_i to evaluate the mask polynomial $h\left(u_i, kf^j + kb^j\right)$ and session key S_j, where j is not in the time window $\left[s_2, s_3\right]$.

(4) t-collusion-free capability: Suppose $\mathcal{R}$ is a set of revoked user in and before s_j, where $|\mathcal{R}| \leq t$, but it is computationally infeasible for them to collude to acquire session key SK_j. From an attacker's perspective, it is easy to crack the private secret seeds fk^j and bk^j by capturing user nodes from network. However, another session key component c_j is a random number chosen independently at each session, and it cannot be deduced from preceding sessions. By analyzing the broadcast message, the way to attack c_j is to recover the jth mask polynomial. The collusive set $\mathcal{R}$ can get at most t values of the current mask polynomial, that is, $h\left(x, kf^j + kb^j\right)$, $u_i \in \mathcal{R}$. But the mask polynomial is t-degree at least $t+1$ points are required to retrieve it. Hence, it is computationally infeasible for $\mathcal{R}$ to calculate c_j and SK_j.

□

Theorem 2. *LiSH+ achieves t-wise forward and backward secrecy.*

Proof.

(1) t-wise forward secrecy. Let R be a set of users revoked in and before $s_j, |\mathcal{R}| \leq t$. Consider a user $u_i \in \mathcal{R}$ whose lifetime is from s_1 to s_2. We can thus analyze the forward secrecy property in three scenarios:

   (a) $s_1 < j < s_2$, which signifies that u_i keeps private secrets of s_j such as d_j, fk^j, bk^j and $h\left(u_i, kf^j + kb^j\right)$. If an attacker further obtains c_j, our session key will be cracked. However, our revocation polynomial v_j contains all of the revoked users, thus for any user $u_i \in \mathcal{R}, v_j\left(u_i\right) = 0$. So it is impractical to decrypt c_j by the revoked user u_i;

   (b) $j > s_2$. In this case, u_i could only get the forward seed fk^j and have no knowledge of any other secrets of s_j. Also as pointed out in case (a), the revocation polynomial $v_j(u_i)$ still equals to zero here;

   (c) The revoked user set $\mathcal{R}$ can get at most t points of the mask polynomial. The coalition of these t users cannot recover the mask polynomial $h\left(x, kf^j + kb^j\right)$.

(2) t-wise backward secrecy. Let $\mathcal{J}$ be a set of users joined after $s_j, |\mathcal{J}| \leq t$. Consider a user $u_i \in \mathcal{J}$ whose lifetime is from s_1 to s_2. We can also analyze the backward secrecy property in three cases:

   (a) $s_1 < j < s_2$. Same to case (a) in proof (1);

   (b) $j < s_1$. u_i could only get the backward seed bk^j and have no knowledge of any other secrets of s_j. Also the revocation polynomial $v_j\left(u_i\right) = 0$ in this case;

   (c) Same to case (c) in proof (1).

In summary, it is infeasible for users revoked in and before s_j or users joined after s_j to compute the jth session key. Our scheme is t-wise forward and backward secure. □

As we mentioned in Section 2, there may be attacks against Du's self-healing chain construction. In Du's scheme, it produces and broadcasts a set

$$D_j = \left\{c_j d_1 \left(c_1 + c_2\right), c_j d_2 \left(c_2 + c_3\right), \cdots, c_j d_{j-2} \left(c_{j-2} + c_{j-1}\right), c_j d_{j-1} c_{j-1}\right\}$$

in each session, where d_j is a private secret in s_j. Legal users who correctly decrypt c_j can obtain a new set

$$D'_j = \left\{d_1 \left(c_1 + c_2\right), \cdots, d_{j-2} \left(c_{j-2} + c_{j-1}\right), d_{j-1} c_{j-1}\right\}$$

by means of dividing each element in the original set D_j by c_j. Finally, a legal user holding the private random numbers $\left\{d_i, d_{i+1}, \cdots, d_{j-1}\right\}$ from s_i to $s_j (i < j)$ can recover the lost keys in its lifetime from set D'_j.

In Du's self-healing design, there is a constant value $d_i \left(c_i + c_{i+1}\right) (1 \leq i < m)$ in every element of self-healing set and these constant values are directly exposed to users. If the user nodes are compromised, this values are directly exposed to the attacker as well. As a result, the attacker can obtain the session. Our new mechanism takes one-way function to protect secrets from being cracked. Even if an attacker knows d_i for s_i, $\{c_{i+1}, d_{i+1}\}$ for s_{i+1} and the broadcast value t_{i+1}, it still cannot decrypt c_i in s_i due to the irreversible computation property of one-way hash function.

6. PERFORMANCE EVALUATION

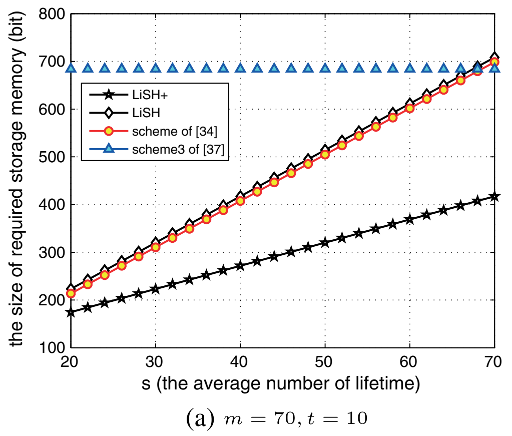
In this section, we evaluate the efficiency of the proposed LiSH+ scheme in terms of storage overhead and communication cost. Table II gives the comparison of LiSH+ with existing schemes based on hash chain in terms of storage overhead, communication cost, revocation ability, collusion resistance, and robustness. Although the scheme of [28] is better in term of storage overhead and communication cost, the revocation ability of [28] is limited. The users cannot be revoked by KDC and will exist only with their lifetimes expiring. Furthermore, schemes of [28] and [30] cannot resist collusion attack. Schemes of [34] and [37] can only partially resist collusion attack. In the following sections, we give a detailed performance evaluation of the proposed LiSH+ scheme from the storage and communication point of view.

Table II. Comparison of storage and communication overhead among schemes based on hash chain.†

Schemes	Storage overhead	Communication cost	Revocation ability	Collusion resistance	Robustness
scheme of [28]	$2\log q$	$1\log q$	Limited	×	✓
scheme3 of [30]	$(t+1)\log q$	$(t+1)\log q$	✓	×	×
scheme of [34]	$(2s_2-2s_1+4)\log q$	$(t+(1+m)/2)\log q$‡	✓	Partial	×
scheme3 of [37]	$(2m+1)\log q$	$(2t+1)\log q$	✓	Partial	×
LiSH	$(2s_2-2s_1+6)\log q$	$(t+l+1)\log q$	✓	✓	✓
LiSH+	$(t+s_2-s_1+6)\log q$	$(t+l+1)\log q$	✓Dynamic	✓	✓

†In this table, m is the number of sessions, t is the degree of polynomial, and $[s_1, s_2]$ is the lifetime of a user.

‡$(t+(1+m)/2)\log q$ is the average communication cost because the communication cost of scheme of [34] is $(t+j)\log q$ for section j.



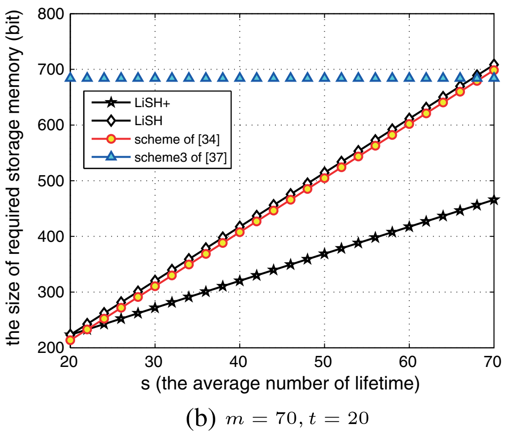
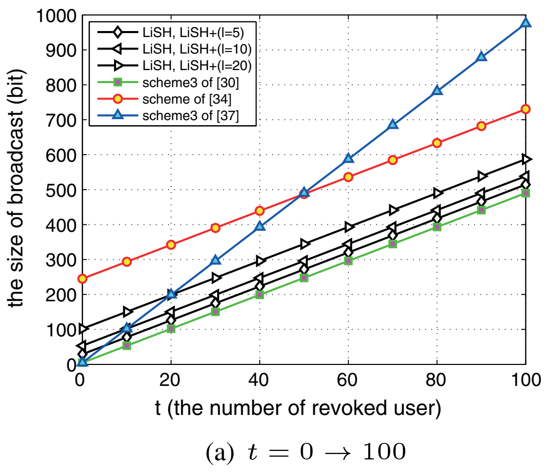


Figure 6. Comparison of the size of required storage memory. Assume that q is a 128-bit integer.



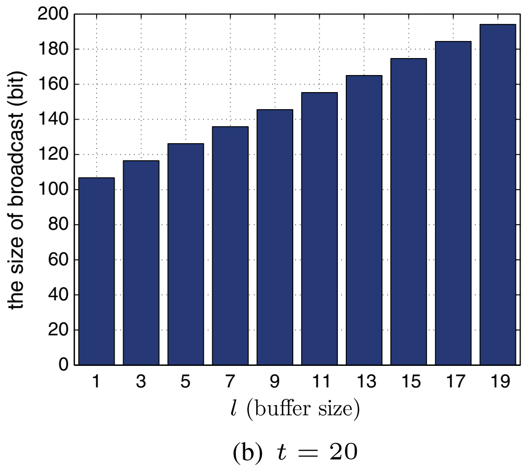


Figure 7. Comparison of the size of broadcasting packet with different buffer size l. Assume that q is a 128-bit integer.

6.1. Storage overhead

In the proposed LiSH+ scheme, a legal user in a group needs to store two initial seeds $\{fk^{s_1}, bk^{s_2}\}$, a private set $\mathcal{D}$, a t-degree mask polynomial $h\left(u_i, kf^j + kb^j\right)$, a buffer length l, and a rekeying period $T_{refresh}$ in its lifetime $\left[s_1, s_2\right]$. The storage requirement of two initial seeds is $1 \log q$ bits and $1 \log q$ bits, respectively. The storage requirement of private set $\mathcal{D}$ is $(s_2 - s_1) \log q$ bits. The storage requirement of the t-degree mask polynomial $h\left(u_i, kf^j + kb^j\right)$ is $t \log q$ bits. The storage requirement of the buffer length l and the rekeying period $T_{refresh}$ is $1 \log q$ bits and $1 \log q$ bits, respectively. Hence, the storage complexity in LiSH+ is $(t + (s_2 - s_1) + 6) \log q$ bits.

Figure 6 shows the size of required storage memory according to s (the average number of user's lifetime), where

$$s = \frac{1}{m} \sum_{foreachuser} (s_2 - s_1)$$

Without loss of generality, we assume that q is a 128-bit integer. Suppose that the maximum session number $m = 70$. We compare the required storage in term of different s and t. With the average number of user's lifetime (s) increasing, the size of required memory in scheme of [34], LiSH, and LiSH+ rise, while scheme3 of [37] needs the same size of memory because its storage overhead only relates to the number of sessions m in the group lifetime. It is easy to see that our LiSH+ scheme needs less storage overhead than the previous schemes need for the same value of m and t. In addition, when s climes to 70, the size of required storage in our LiSH+ scheme is approximately 400 bits ($t = 10$), while that of other schemes is around 700 bits. As a result, our LiSH+ scheme can save more memory for resource-constrained RTUs in the same security level. When the degree of bivariate polynomial increases, the required storage in LiSH+ scheme becomes larger and the benefit is that it is more difficult for the attacker to crack the mask polynomial.

6.2. Communication cost

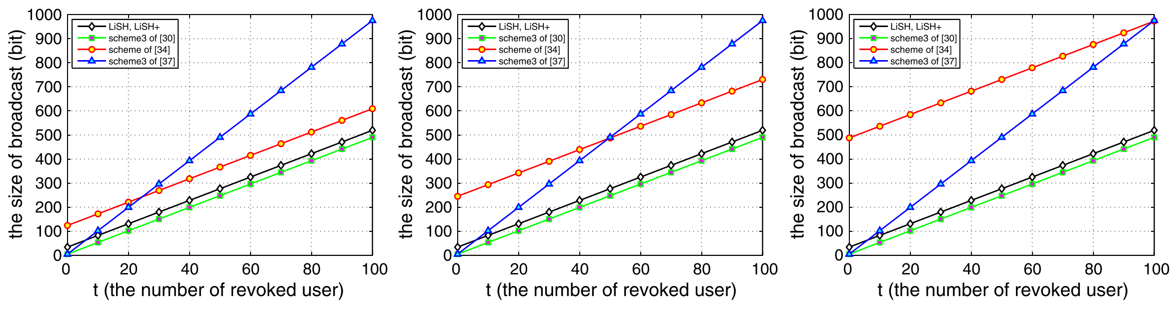
In the proposed LiSH+ scheme, the broadcast message is $B_j = \left\{\mathcal{R}_j \bigcup \mathcal{L}_j, \mathcal{T}_j, b_j\right\}$. The communication cost for the broadcast of $\mathcal{R}_j \bigcup \mathcal{L}_j$ can be neglected since user ID can be selected from a small finite field [27]. Considering practical necessity of SCADA system, we limit the self-healing set $\mathcal{T}_j$ to $l \log q$ bits in our proposed scheme. b_j is a t-degree polynomial with communication bandwidth of $(t + 1) \log q$ bits. Thus, the total communication cost of one broadcasting message is $(t + l + 1) \log q$ bits.

Without loss of generality, we assume that q is a 128-bit integer. We compare the required storage in term of different l (buffer size), m (maximum session number), and t (the degree of broadcast polynomial).

Figure 7 illustrates the size of broadcast packet with different buffer size. Figure 7(a) shows that our scheme has less communication overhead than that of [34] and scheme3 in [37] with the same value of m and t. The broadcast packet size of our scheme is a little larger than that of schemes of [30], and the reason is that our scheme can resist collusion attack at the cost of slightly increased communication overhead by introducing secure self-healing mechanism. Figure 7(b) displays the size of broadcast packet according to the buffer size when t equals to 20. It can be seen that the size of broadcast packet increases slowly with the increase of the buffer size.

Figure 8 shows the increase tendency of the size of broadcast with different values of m and t. When m varies from 50 to 200, the scheme of [34] is affected dramatically, and its communication cost increases sharply, while other schemes remain the same size of broadcast. In addition, when m equals to 200 and t equals to 100, the size of broadcast in our scheme and scheme of [30] is roughly 500 bits while that of scheme of [34] and scheme3 of [37] exceeds 950 bits.

The preceding discussions indicate that our LiSH+ scheme can achieve the security requirements such as revocation ability, collusion, and resistance robustness with moderate communication overhead, which also demonstrates LiSH+'s usability for resource-constrained SCADA system.



(a) $m = 50, l = 6$ (b) $m = 100, l = 6$ (c) $m = 200, l = 6$

Figure 8. Comparison of the size of broadcasting packet. Assume that q is a 128-bit integer.

7. CONCLUSIONS

Group communication is one of the most important communication paradigms for SCADA systems in smart grid. A robust and efficient group key management is vital for time-critical SCADA systems to make them self-healing and secure in case of system failure and attacks. In this paper, we have proposed an enhanced robust and efficient group key management, named LiSH+, to secure SCADA system in smart grid. The proposed LiSH+ scheme is characterized by adopting self-healing key to tolerant failures of the sub-MTUs and revoking compromised users dynamically. Security analysis has shown that the proposed LiSH+ is a collusion-free and self-healing key distribution scheme with *t*-wise forward and backward security. In addition, performance evaluation has also demonstrated its efficiency. In our future work, we will consider threshold based key refreshment scheme and leverage the static topology feature of SCADA systems to further detect and eject the compromised users efficiently.

ACKNOWLEDGEMENTS

This work is supported by China Scholarship Council, the National Natural Science Foundation of China under Grant Nos. 61170261, 61073189, 61272437, and 61202369, and NSERC, Canada.

REFERENCES

1. Wang W, Lu Z. Cyber security in the smart grid: survey and challenges. *Computer Networks* 2013; **7**(8): 1754–1773.
2. *Northeast blackout of 2003*. http://en.wikipedia.org/wiki/Northeast_blackout_of_2003 [Accessed on 1 March 2013].
3. India blackouts, 2012. http://en.wikipedia.org/wiki/India_blackout [Accessed on 1 March 2013].
4. Lu R, Liang X, Li X, Lin X, Shen X. PPA: an efficient and privacy-preserving aggregation scheme for secure smart grid communications. *IEEE Transactions on Parallel and Distributed Systems* 2012; **23**(9): 1621–1631.
5. Li X, Liang X, Lu R, Shen X, Lin X, Zhu H. Securing smart grid: cyber attacks, countermeasures, and challenges. *IEEE Communications Magazine* 2012; **50**(8): 38–45.
6. Yan Y, Qian Y, Sharif H, Tipper D. A survey on cyber security for smart grid communications. *IEEE Communications Surveys & Tutorials* 2012; **14**(4): 998–1010.
7. Sridhar S, Govindarasu M, Liu CC. Risk analysis of coordinated cyber attacks on power grid. *Control and Optimization Methods for Electric Smart Grids,* Springer 2012; **3**: 275–294.
8. Wang Y. sSCADA: securing SCADA infrastructure communications. *International Journal of Communication Networks and Distributed Systems* 2011; **6**(1): 59–78.
9. Choi D, Lee S, Won D, Kim S. Efficient secure group communications for SCADA. *IEEE Transactions on Power Delivery* 2010; **25**(2): 714–722.
10. Beaver C, Gallup D, Neumann W, Torgerson M. Key management for SCADA. *Technical Report*, Cryptog. Information Sys. Security Dept., Sandia Nat, 2002. Labs SAND2001-3252.
11. Igure VM, Laughter SA, Williams RD. Security issues in SCADA networks. *Computers & Security* 2006; **25**(7): 498–506.
12. Choi D, Jeong H, Won D, Kim S. Hybrid key management architecture for robust SCADA systems. *Journal of Information Science and Engineering* 2011; **27**: 197–211.
13. Dawson R, Boyd C, Dawson E, Nieto JMG. SKMA: a key management architecture for SCADA systems, *Australasian Workshops on Grid Computing and E-research*, Darlinghurst, Australia, 2006; 183–192.
14. Curtis K. *A DNP3 protocol primer*, 2005. *DNP User Group*.
15. Yan Y, Qian Y, Sharif H, Tipper D. A survey on smart grid communication infrastructures: motivations, requirements and challenges. *IEEE Communications Surveys & Tutorials* 2013; **15**(1): 5–20.
16. Rong B, Chen HH, Qian Y, Lu K, Hu RQ, Guizani S. A pyramidal security model for large-scale group-oriented computing in mobile ad hoc networks: the key management study. *IEEE Transactions on Vehicular Technology* 2009; **58**(1): 398–408.
17. Wong CK, Gouda M, Lam SS. Secure group communications using key graphs. *IEEE/ACM Transactions on Networking* 2000; **8**(1): 16–30.
18. Poovendran R, Baras JS. An information-theoretic approach for design and analysis of rooted-tree-based multicast key management schemes. *IEEE Transactions on Information Theory* 2001; **47**(7): 2824–2834.
19. Qian Y, Lu K, Tipper D. A design for secure and survivable wireless sensor networks. *IEEE Wireless Communications* 2007; **14**(5): 30–37.
20. Lu K, Qian Y, Guizani M, Chen HH. A framework for a distributed key management scheme in heterogeneous wireless sensor networks. *IEEE Transactions on Wireless Communications* 2008; **7**(2): 639–647.
21. Jiang R, Luo J, Wang X. HRKT: hierarchical route key tree based group key management for wireless sensor networks. *KSII Transaction on Internet and Information Systems* 2013; **7**(8): 1754–1773.

22. Qian Y, Moayeri N. Design of secure and application-oriented vanets, *IEEE Vehicular Technology Conference(VTC Spring 2008)*, Singapore, 2008; 2794–2799.
23. Choi D, Kim H, Won D, Kim S. Advanced key-management architecture for secure SCADA communications. *IEEE Transactions on Power Delivery* 2009; **24**(3): 1154–1163.
24. Jiang R, Lu R, Lai C, Luo J, Shen X. Robust group key management with revocation and collusion resistance for SCADA in smart grid, *IEEE Globe Communication Conference*, Atlanta, GA, 2013; 824–829.
25. Staddon J, Miner S, Franklin M, Balfanz D, Malkin M, Dean D. Self-healing key distribution with revocation, *IEEE Symposium on Security and Privacy*, Berkeley, California, 2002; 241–257.
26. Blundo C, Darco P, De Santis A, Listo M. Design of self-healing key distribution schemes. *Designs, Codes and Cryptography* 2004; **32**(1): 15–44.
27. Hong D, Kang JS. An efficient key distribution scheme with self-healing property. *IEEE Communications Letters* 2005; **9**(8): 759–761.
28. Jiang Y, Lin C, Shi M, Shen X. Self-healing group key distribution with time-limited node revocation for wireless sensor networks. *Ad Hoc Networks* 2007; **5**(1): 14–23.
29. Dutta R, Mukhopadhyay S. Improved self-healing key distribution with revocation in wireless sensor network, *IEEE WCNC*, Kowloon, 2007; 2963–2968.
30. Dutta R, Mukhopadhyay S, Collier M. Computationally secure self-healing key distribution with revocation in wireless ad hoc networks. *Ad Hoc Networks* 2010; **8**(6): 597–613.
31. Du W, He M. Self-healing key distribution with revocation and resistance to the collusion attack in wireless sensor networks, *Provable Security*, Shanghai, China, 2008; 345–359.
32. Xu Q, He M. Improved constant storage self-healing key distribution with revocation in wireless sensor network, *Information Security Applications*, Jeju Island, Korea, 2009; 41–55.
33. Yang Y, Zhou J, Deng R, Bao F. Computationally secure hierarchical self-healing key distribution for heterogeneous wireless sensor networks, *Information and Communications Security*, Beijing, China, 2009; 135–149.
34. Du C, Hu M, Zhang H, Zhang W. Anti-collusive self-healing key distribution scheme with revocation capability. *Information Technology Journal* 2009; **8**(4): 619–624.
35. Wen M, Liu X, Li J, Wang Y. A group key distribution scheme with verification and revocation. *Journal of Shanghai Jiaotong University* 2012; **46**(2): 12–17.
36. Kang D, Lee J, Kim B, Hur D. Proposal strategies of key management for data encryption in SCADA network of electric power systems. *International Journal of Electrical Power & Energy Systems* 2011; **33**(9): 1521–1526.
37. Tian B, Han S, Dillon TS, Das S. A self-healing key distribution scheme based on vector space secret sharing and one way hash chains, *IEEE WoWMoM*, Newport Beach, CA, 2008; 1–6.